

# Lecture 7: Application Layer

## Email and SMTP

COMP 332, Fall 2018

Victoria Manfredi

WESLEYAN  
UNIVERSITY



**Acknowledgements:** materials adapted from Computer Networking: A Top Down Approach 7<sup>th</sup> edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University and some material from Computer Networks by Tannenbaum and Wetherall.

# Today

## Announcements

- homework 3 due Wednesday by 11:59p
  - parsing HTTP requests and responses
    - put any decoding in a try block, send raw bytes even if can't decode
  - client needs to generate HTTP request
    - “Connection: close\r\n” in header will close socket after each response

## Electronic mail

- overview
- SMTP
- mail access protocols

## Domain names

# Electronic Mail

## **OVERVIEW**

## Ray Tomlinson at Raytheon BBN Technologies

### THE FATHER OF EMAIL

REMEMBERING RAYTHEON ENGINEER RAY TOMLINSON 1941-2016



Engineer Ray Tomlinson sent the first network email in 1971, choosing the '@' symbol to separate the name of the sender from the address of the host computer.

Share

*In 1971, in a windowless room in Cambridge, Massachusetts, a bearded computer scientist named Ray Tomlinson was hunched before two massive computers, struggling to send the world's first email.*

He had been programming and debugging for hours, trying fruitlessly to get a message from one cabinet-sized computer to another.

Now he tried again, banging out his name on a teletype keyboard: TOMLINSON. He followed that with an @ symbol – a little-used key he had chosen as a separator – and then the name of the other computer.

Tomlinson rolled his chair over to the second computer's teletype and banged out TYPE MAILBOX on the keyboard.

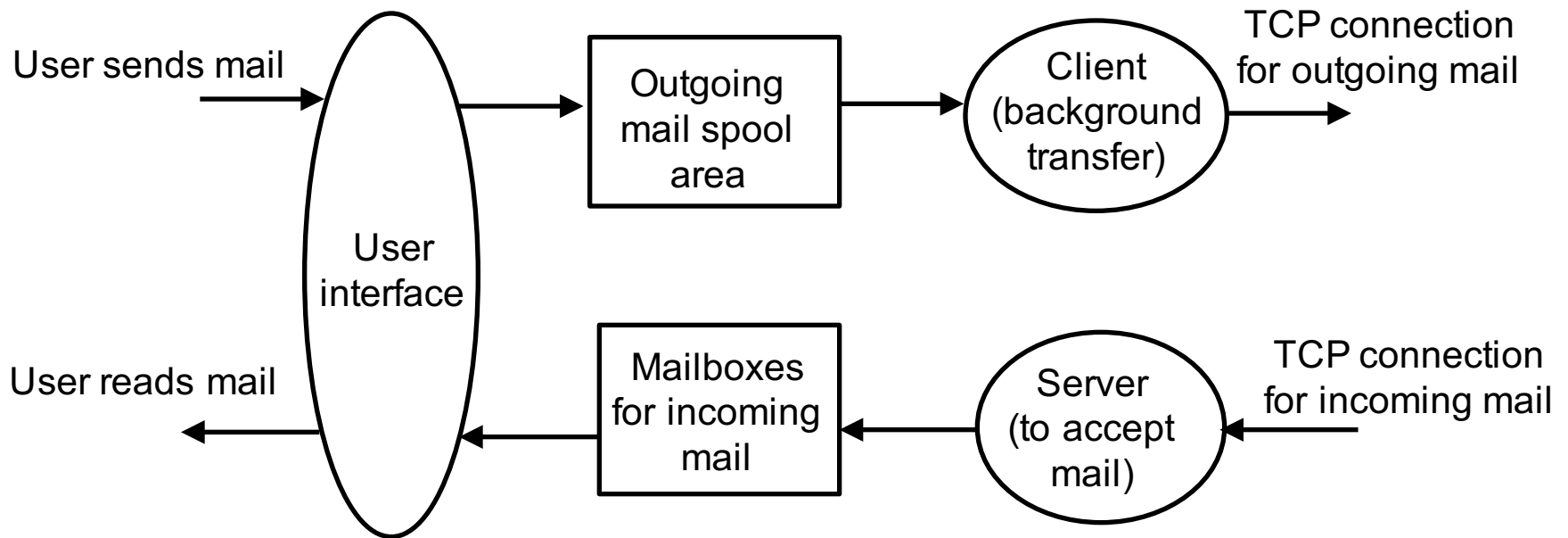
For a moment there was silence. And then with a rattle, the teletype came alive. History's first email had arrived.

"The mail was sitting there just like it is today when you check your inbox," Tomlinson said.

Tomlinson, a principal engineer at Raytheon BBN Technologies, passed away on March 5, 2016. He was 74 years old.

Inducted into the Internet Hall of Fame in 2012 for his invention of modern email, Tomlinson made the historic choice to separate the name of his message's recipient from the name of the host computer using the "@" symbol, creating one of the most universally recognized digital icons on the planet. In 2011, he was ranked No. 4 on the list of the top 150 MIT-

# Overview



## Uses client-server communication

- **not interactive**: transfer of msgs occurs in background (“spooling”)

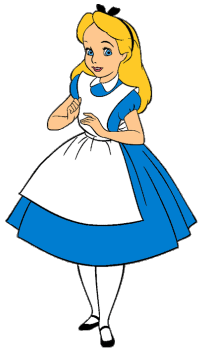
## Reliable service

- **uses TCP**

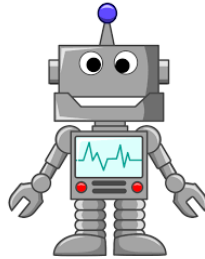
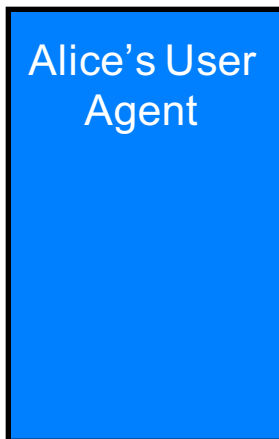
# Major components of electronic mail (email)

## User-agents aka mail reader (what you use)

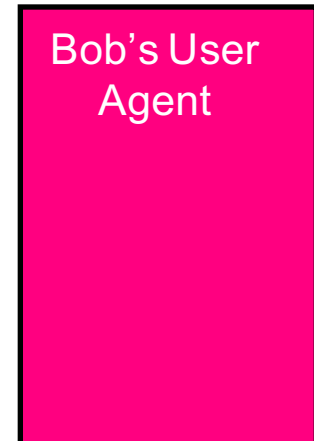
- composing, editing, reading mail messages
- e.g., Outlook, Thunderbird, iPhone mail client, Gmail
- incoming/outgoing messages stored on mail server
- client-server communication with mail server



Alice



Bob



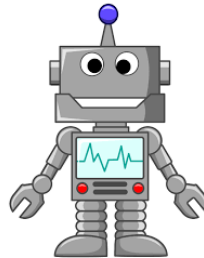
# Major components of electronic mail (email)

## Mail servers

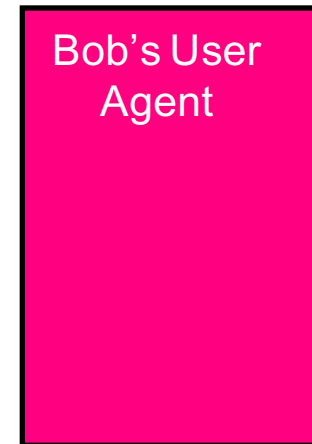
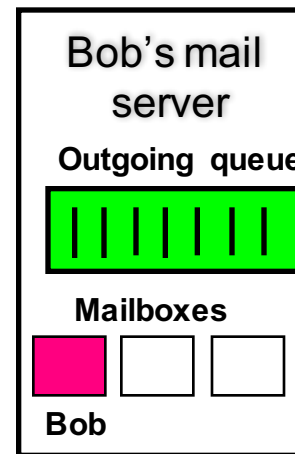
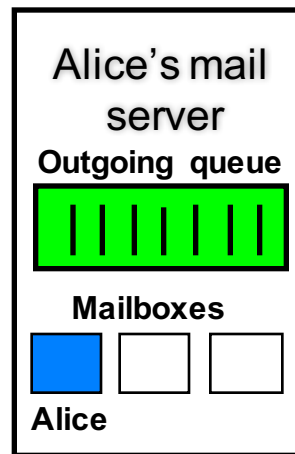
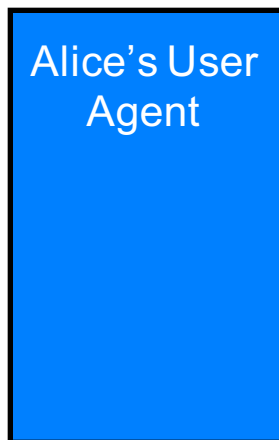
- mailbox for each user: holds user's incoming messages
- **outgoing message queue**: holds messages to be sent
  - messages held in queue until successfully delivered
  - reattempts done every 30 min or so. If undeliverable, user notified



Alice



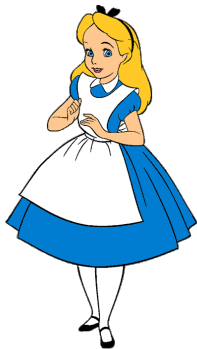
Bob



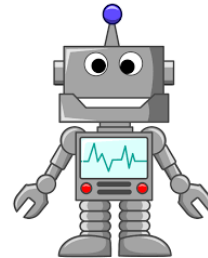
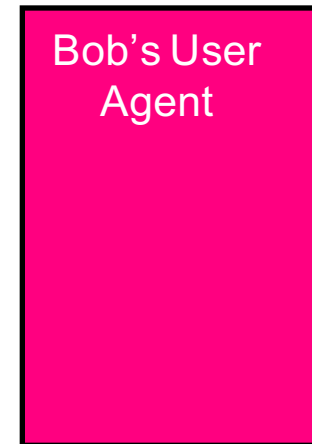
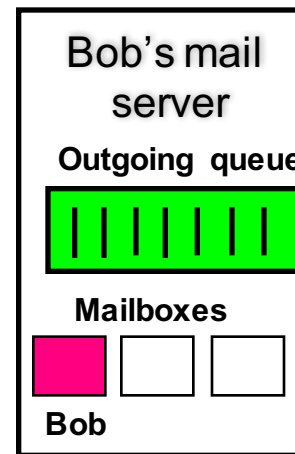
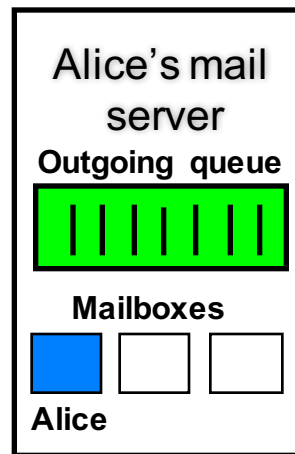
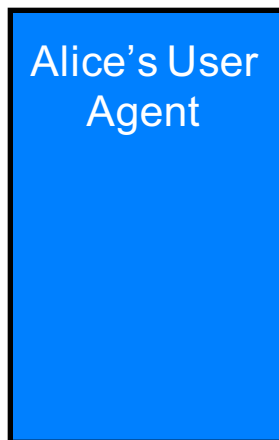
# Major components of electronic mail (email)

## SMTP (simple mail transfer protocol)

- **transfers msgs:** from user agent to mail server and between mail servers
- persistent connection, TCP port 25, SSL encrypted uses port 465
- p2p comm among mail servers, client-server with user-agents
  - user agent does not run server side of SMTP (would need to always be on)
  - mail server runs both client and server sides



Alice



Bob



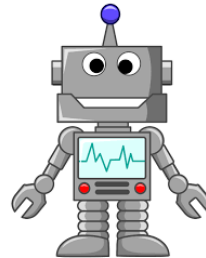
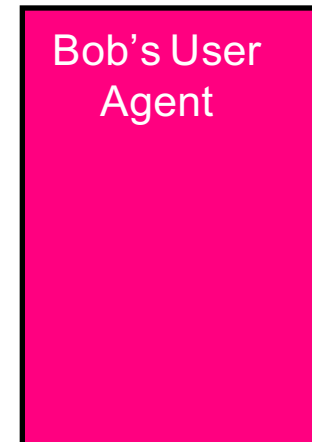
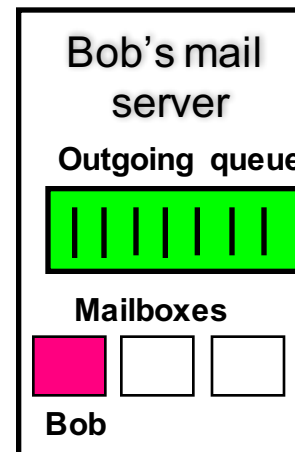
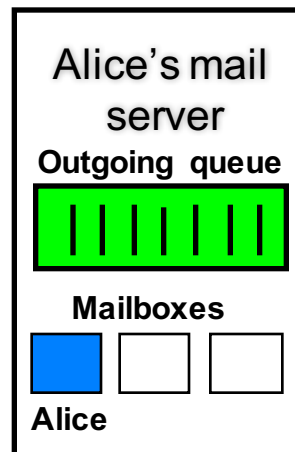
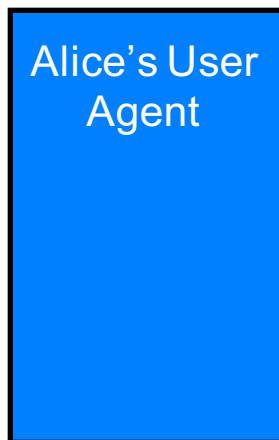
# Major components of electronic mail (email)

## Mail access protocols for user agent to retrieve mail

- POP3: Post Office Protocol
  - basic: downloads email, deletes from server, emails stored on computer
- IMAP: Internet Mail Access Protocol
  - more complex, recommended over POP3
    - manipulate msgs stored on server, email stored on server, use multiple computers
- HTTP: used by gmail, yahoo, etc ...



Alice

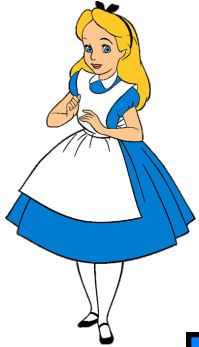


Bob

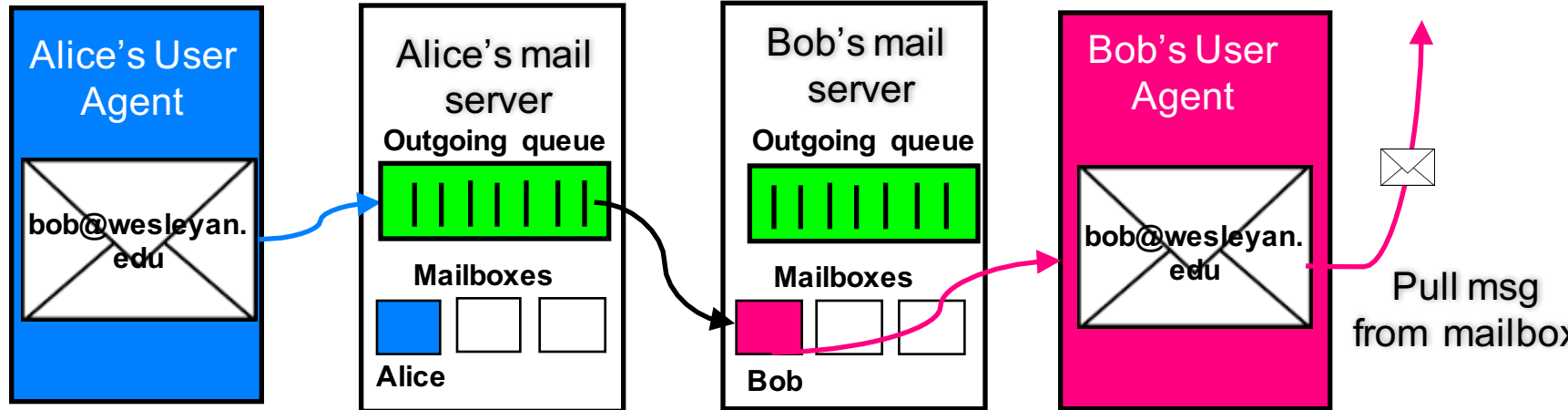
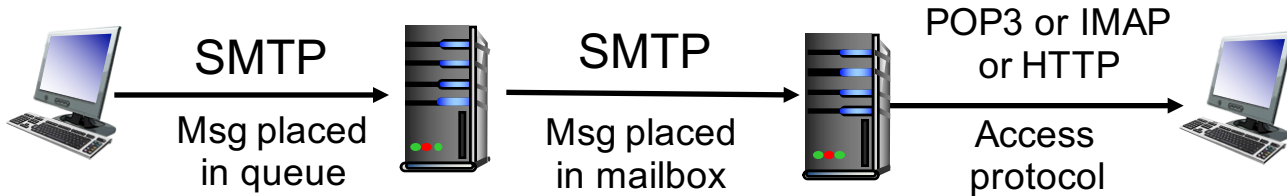
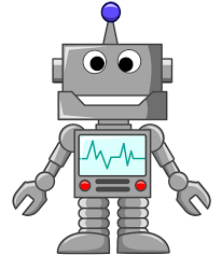


# What happens when Alice sends email to Bob?

Alice

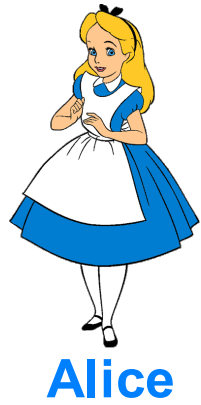


Bob

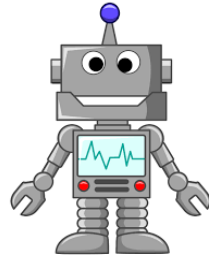


Q: What happens before any mail protocol communication?  
TCP handshake

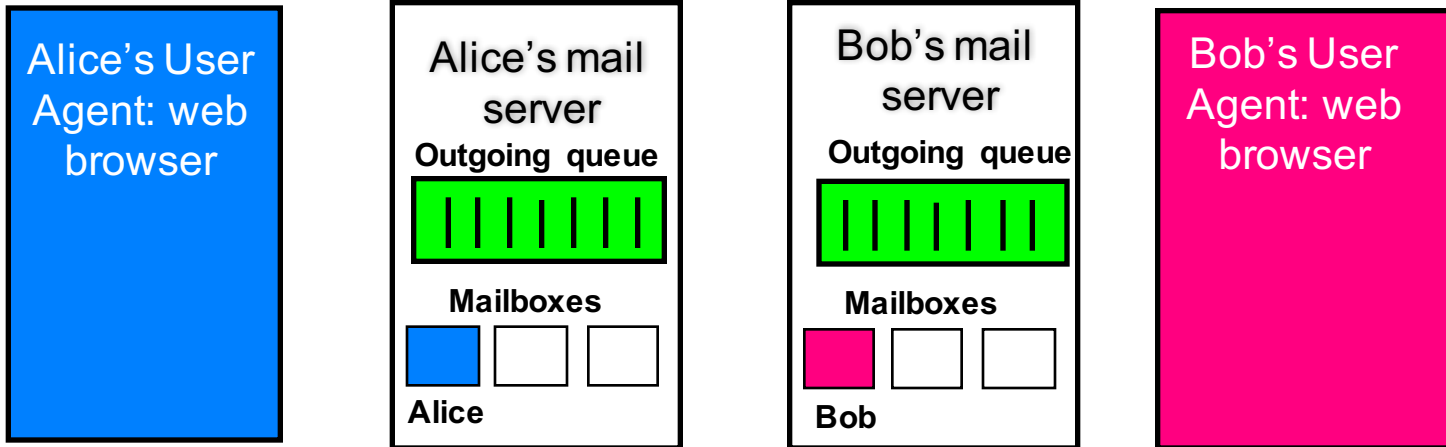
# Webmail



Alice



Bob



HTTP is used for communication between Client and mail server  
SMTP is used for communication between mail servers

# Electronic Mail

# **SIMPLE MAIL TRANSFER PROTOCOL**

# SMTP [RFC 2821]

## Simple Mail Transfer Protocol

- defines exchange of mail from client to server and between servers
- **uses TCP**: to reliably transfer email message from client to server

## Direct transfer

- **sending server to receiving server**
- **3 phases of transfer**
  - handshaking (greeting)
  - transfer of messages
  - closure

## Command/response interaction (like HTTP)

- **commands**: ASCII text
- **response**: status code and phrase

# Testing out SMTP

## Logon to an SMTP server

- use nc or telnet to open insecure connection (probably won't work...)
  - `nc exchange2010.wesleyan.edu 25`
- use openssl to open secure connection
  - `openssl s_client -crlf -connect exchange2010.wesleyan.edu:465`
- can use openssl to connect to https sites as well
  - `openssl s_client -crlf -connect www.bankofamerica.com:443`


## See 220 reply from server

- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands
- above lets you send email without using email client
  - you're directly logged onto mail server

# Sample SMTP interaction once logged on

```
C: nc hamburger.edu 25
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 server uses CRLF.CRLF to determine end of message

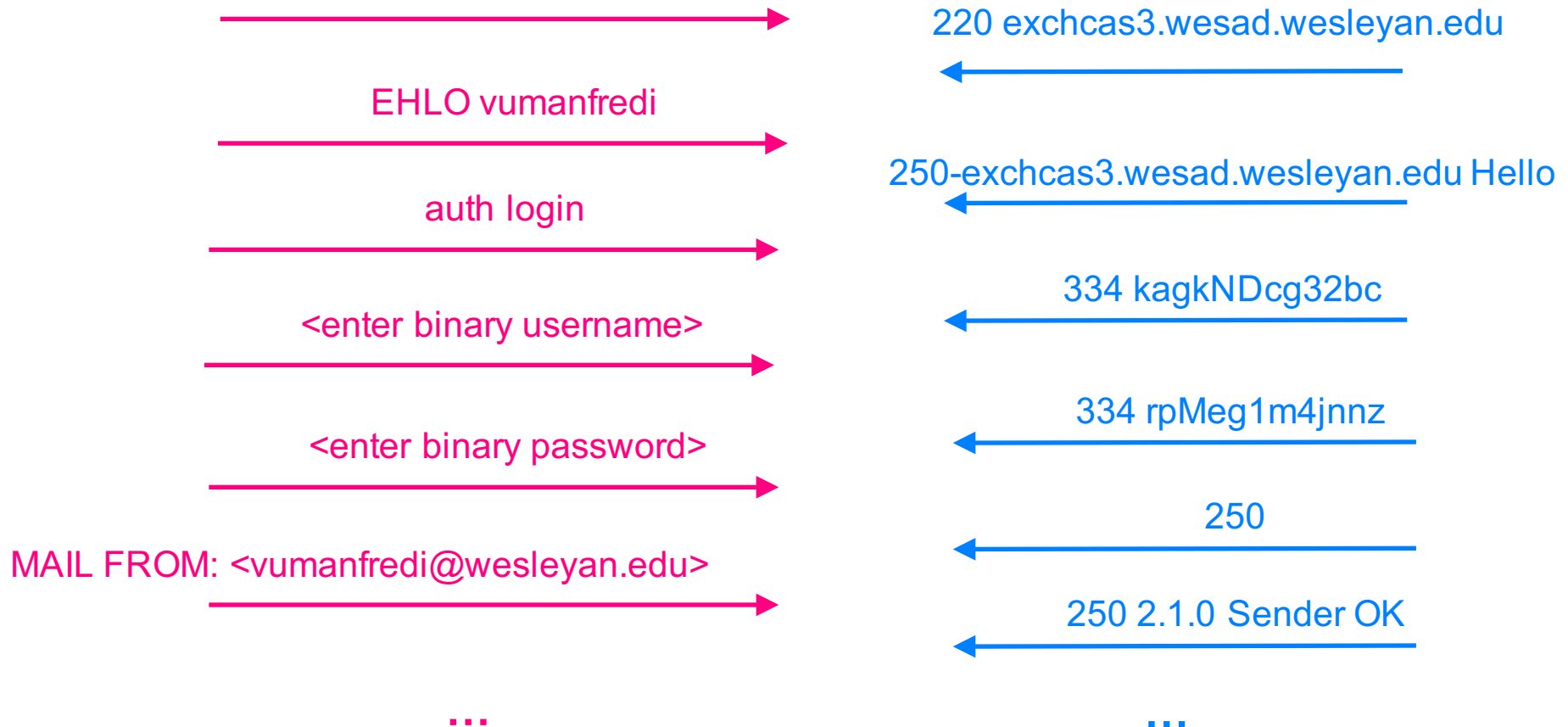


To really try this in practice, we need to encrypt...

# SMTP client-server commands

**Client** establishes SSL/TCP connection to **Server**

openssl s\_client -crlf -connect exchange2010.wesleyan.edu:465



See smtp.txt on schedule for full example and try yourself



# Look at smtp.txt handout

Walkthrough how to logon to mail server and send email

# HTTP vs. SMTP

## HTTP

- pull
- each object encapsulated in its own response message

## SMTP

- push
- multiple objects sent in multipart message

## Both

- ASCII command/response interaction
- status codes

# SMTP message format

## RFC 822

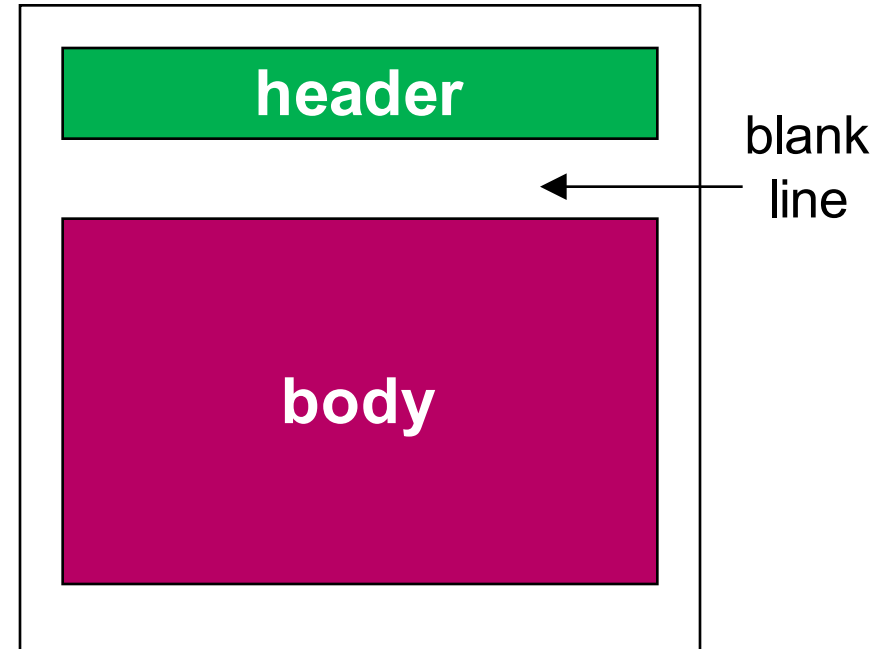
- specifies format of e-mail message

## Header lines

- To:
- From:
- Subject:
- **different from SMTP MAIL FROM, RCPT TO!**

## Body

- the “message”
- ASCII characters only



## Q: How to send images?

MIME (Multipurpose Internet Mail Extensions) encodes arbitrary data (e.g. binary image) in plain ASCII text. SMTP supports only ASCII messages

# MIME extension for images

## Multipurpose Internet Mail Extensions, RFC 2045, 2056

- additional lines in message header declare MIME content type
- message can have multiple parts, e.g., text, image, etc.

**MIME version**

**Method used  
to encode data**

**Multimedia data  
type, subtype,  
parameter declaration**

**Encoded data**

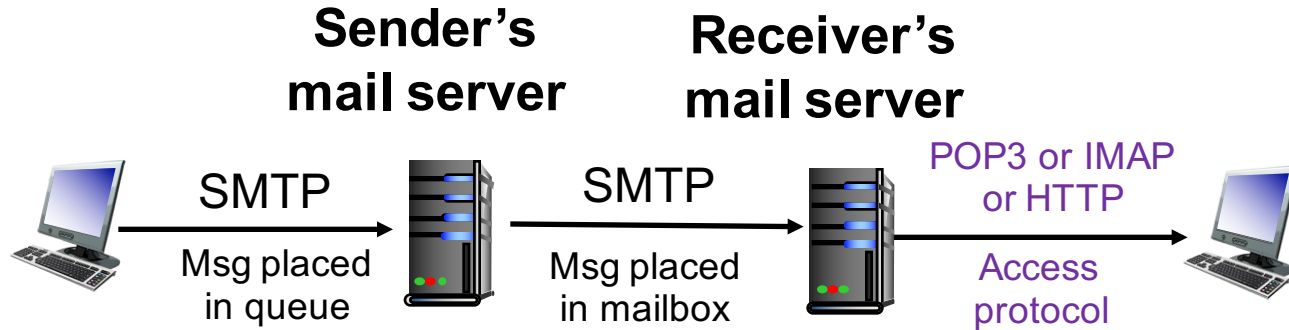
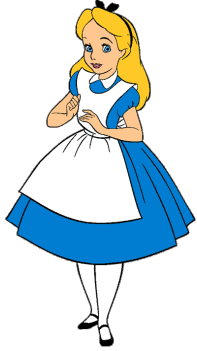
```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
base64 encoded data .....
.....
.....base64 encoded data
```

# Electronic Mail

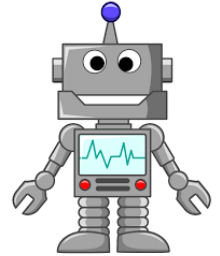
# MAIL ACCESS PROTOCOLS

# Mail access protocols

Alice



Bob



## Mail retrieval from server

- **POP3**: Post Office Protocol [RFC 1939]
  - authorization (agent <-> server) and download
- **IMAP**: Internet Mail Access Protocol [RFC 1730]
  - more features
  - 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 vs. IMAP

## POP3

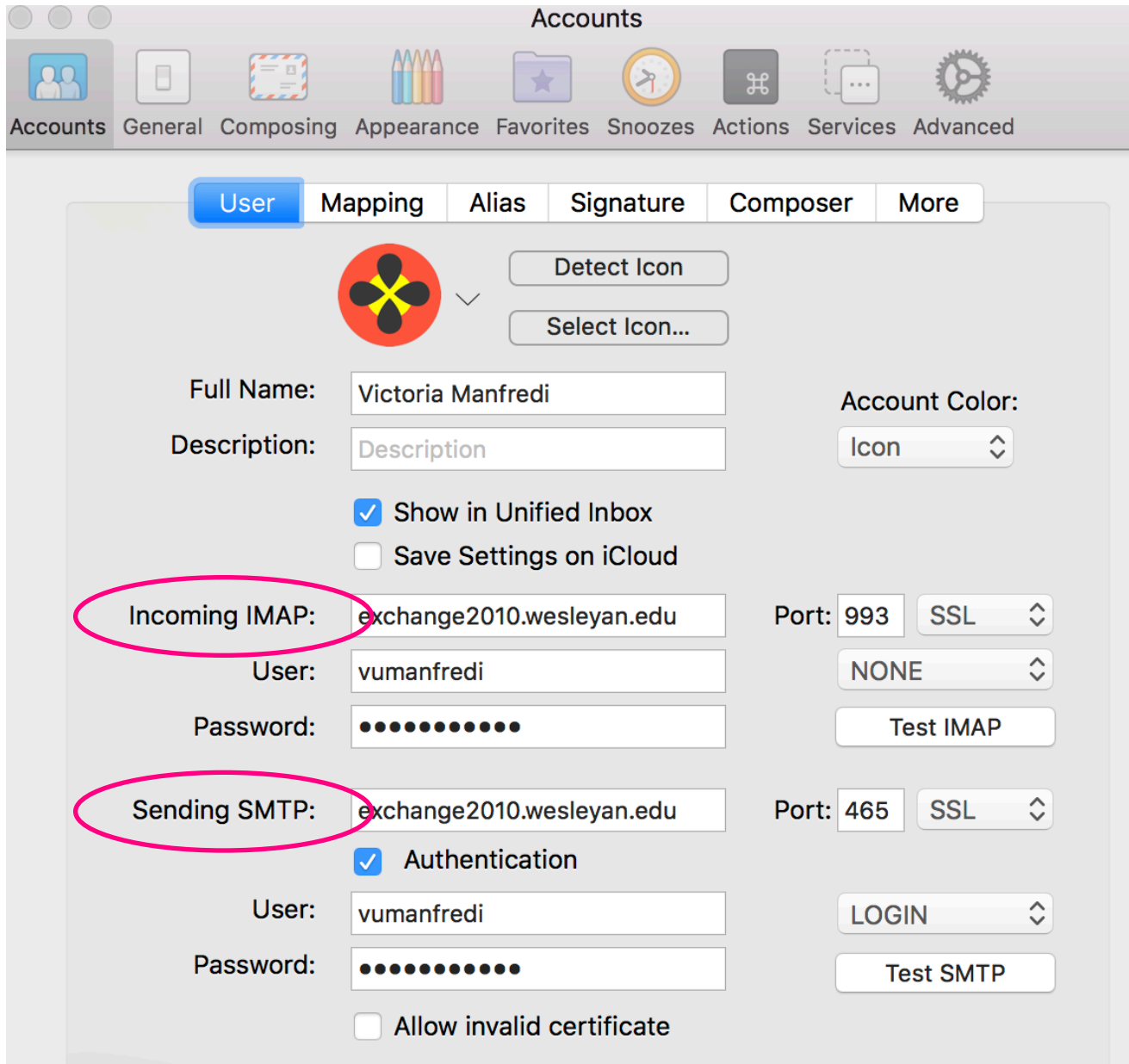
- “download and delete” mode
  - previous example: Bob cannot re-read e-mail if he changes client
- “download-and-keep” mode
  - copies of messages on different clients
- **stateless** across sessions

## IMAP

- **keeps all messages 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



# Setting up your user agent



# Mail server ip address

```
> dig exchange2010.wesleyan.edu

; <◇> DiG 9.8.3-P1 <◇> exchange2010.wesleyan.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22981
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;exchange2010.wesleyan.edu.      IN      A

;; ANSWER SECTION:
exchange2010.wesleyan.edu. 283 IN      A      129.133.7.96
```

```
> dig wesleyan.edu

; <◇> DiG 9.8.3-P1 <◇> wesleyan.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38320
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;wesleyan.edu.                  IN      A

;; ANSWER SECTION:
wesleyan.edu.                 21593  IN      A      129.133.7.68
```

# Look at complete email header

Show raw source in gmail or wesleyan email

# Domain Names

## **OVERVIEW**

# Problem

## People have multiple identifiers

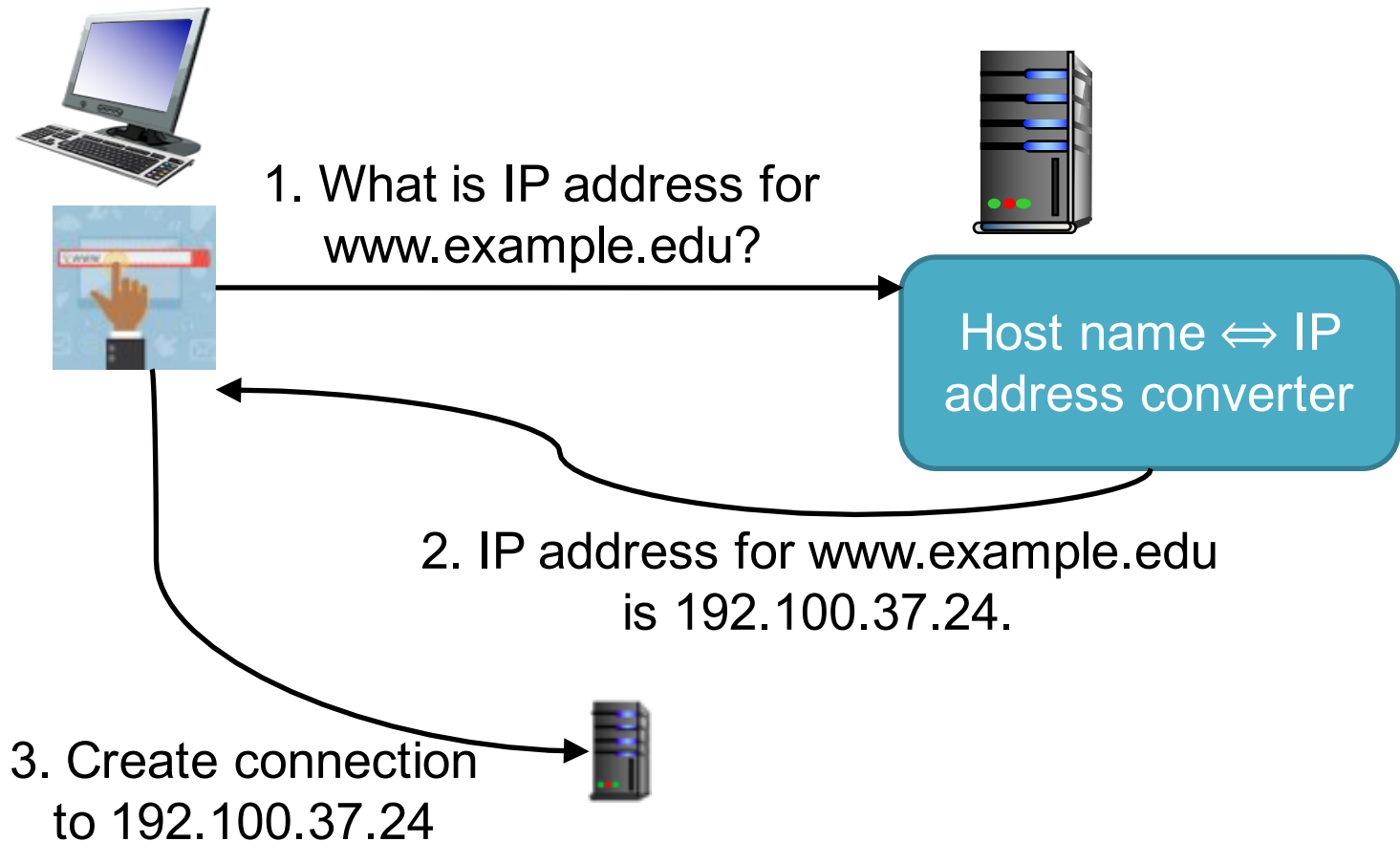
- SSNs, name/nickname, passport #, driver's license #, phone #, ...
- directories, phone books, registries

## Internet hosts, routers have multiple identifiers too

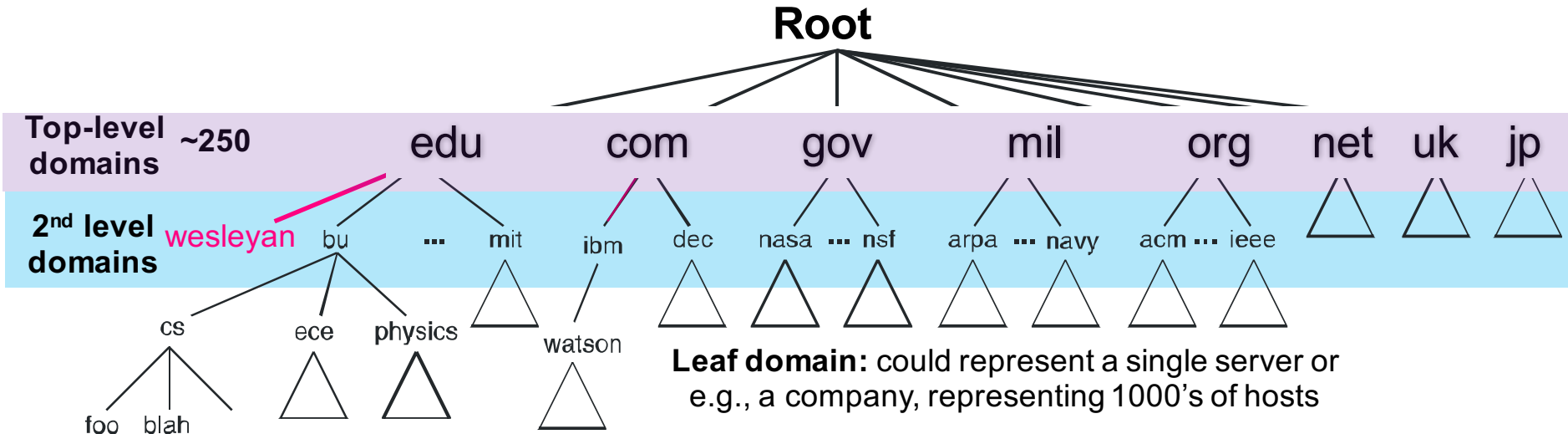
- **IPv4 address** (32 bits written as a 'dotted quad' nnn.nnn.nnn.nnn)
  - used to address packets
  - processed by routers
- **“name”**, e.g., [www.google.com](http://www.google.com)
  - used by humans (who are really bad remembering strings of numbers)
  - canonical “true” name vs. aliases which may point to same host

Q: how to map between IP address and name, and vice versa? Why is this needed?

# When hostname is typed into browser...



# Internet domain name space is hierarchical



Each domain name is a leaf/node in a subtree, e.g.

- .edu → .bu.edu → .cs.bu.edu → [www.cs.bu.edu](http://www.cs.bu.edu)

Why subtrees? Prevents name collisions

- [www.bu.edu](http://www.bu.edu) vs [www.bu.com](http://www.bu.com) vs [www.bu.org](http://www.bu.org)

# dig wesleyan.edu to get ip address

```
> dig wesleyan.edu

; <<> DiG 9.8.3-P1 <<> wesleyan.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11633
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;wesleyan.edu.                IN      A

;; ANSWER SECTION:
wesleyan.edu.                21600  IN      A      129.133.7.68

;; Query time: 3877 msec
;; SERVER: 129.133.52.12#53(129.133.52.12)
;; WHEN: Sun Sep 23 19:20:04 2018
;; MSG SIZE rcvd: 46
```



# Hostname may not map to unique IP address

```
> dig vumanfredi.web.wesleyan.edu

; <<> DiG 9.8.3-P1 <<> vumanfredi.web.wesleyan.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58367
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;vumanfredi.web.wesleyan.edu.      IN      A

;; ANSWER SECTION:
vumanfredi.web.wesleyan.edu. 3589 IN     CNAME  wesfiles.wesleyan.edu.
wesfiles.wesleyan.edu.      533    IN      A      129.133.6.79
```

```
> dig dlicata.web.wesleyan.edu

; <<> DiG 9.8.3-P1 <<> dlicata.web.wesleyan.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45830
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;dlicata.web.wesleyan.edu.        IN      A

;; ANSWER SECTION:
dlicata.web.wesleyan.edu. 3600 IN     CNAME  wesfiles.wesleyan.edu.
wesfiles.wesleyan.edu.      524    IN      A      129.133.6.79
```