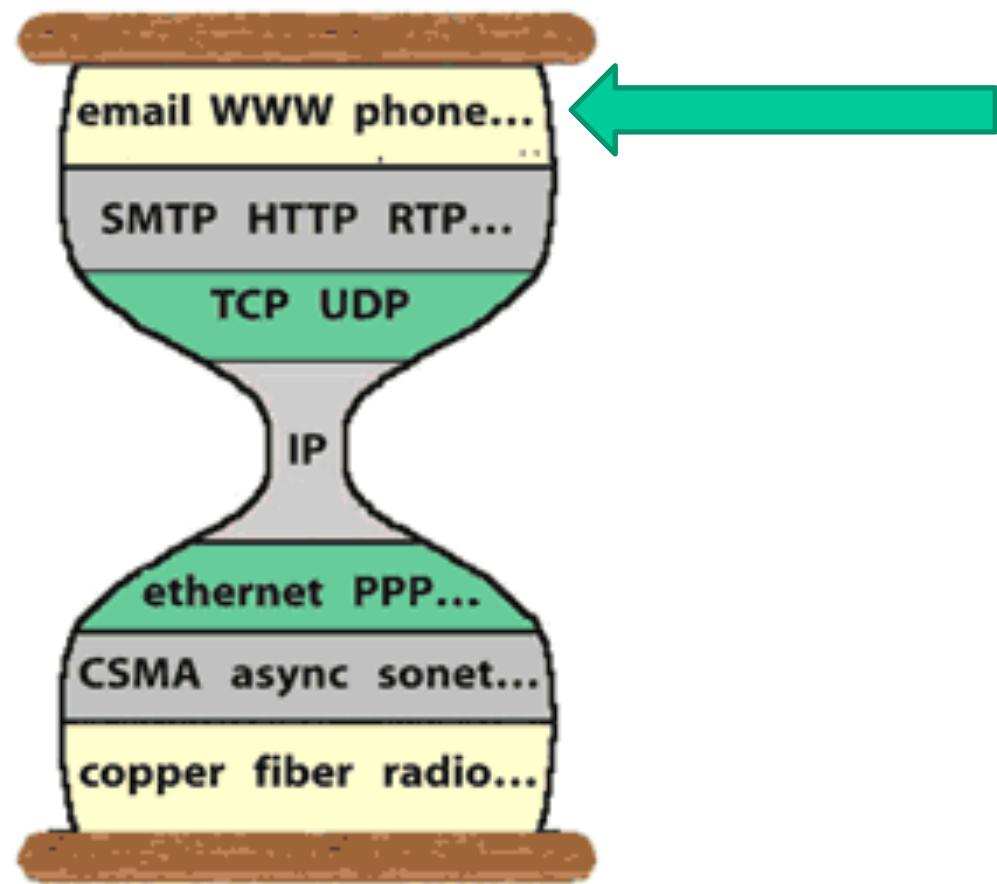


Plan for today: More on Application Layer

- ◆ Finish with HTTPS
- ◆ Electronic Mail
 - How many parts in email delivery system
 - Simple Mail Transfer Protocol (SMTP)
 - Mail retrieval protocols:
 - POP3
 - IMAP
 - Basics of email security
- ◆ P2P

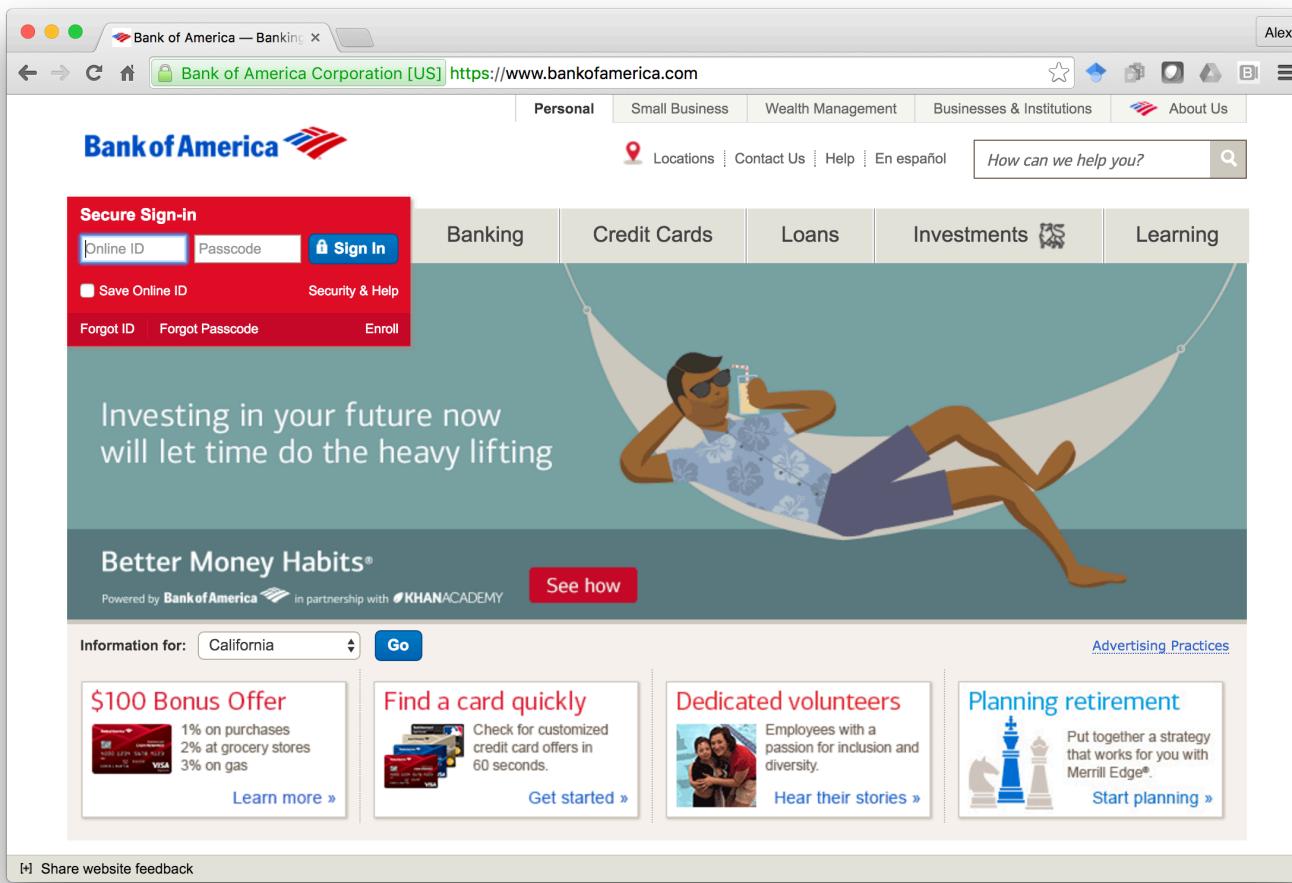


Storage-Less Cookies (last lecture question)

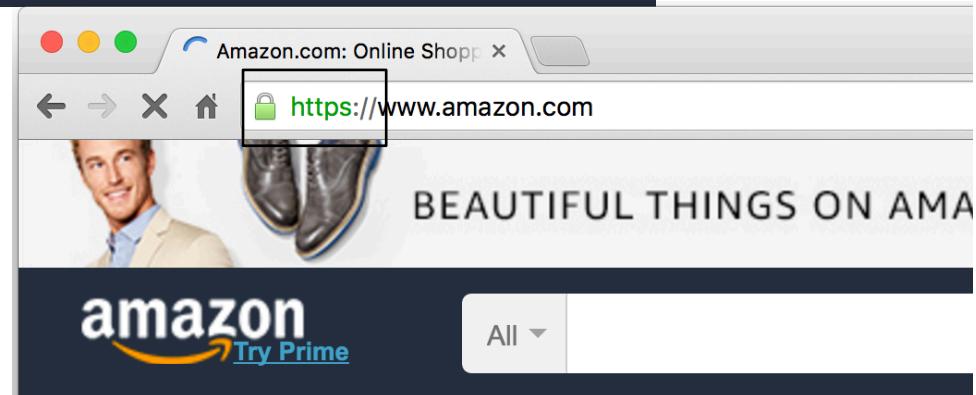
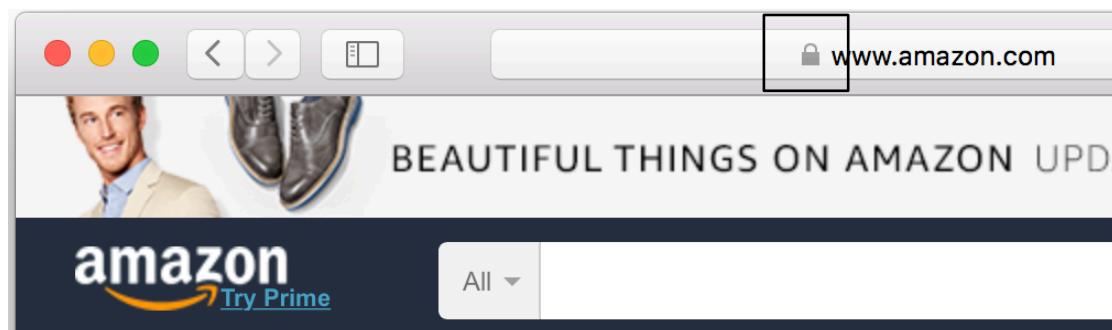
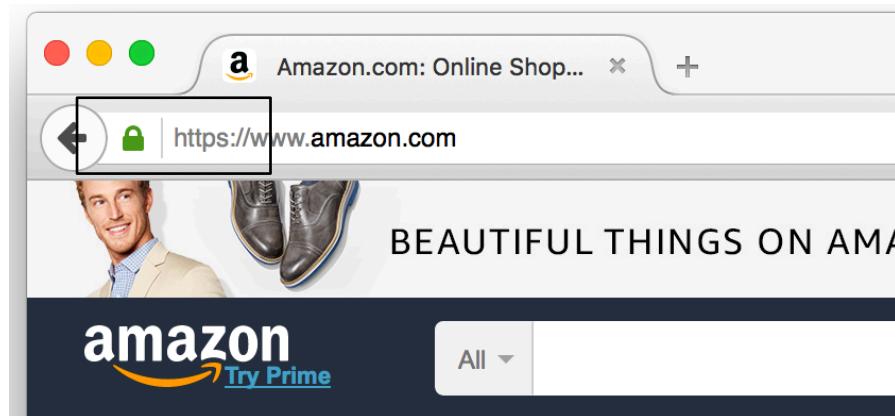
- ◆ Say, server has databases of users
- ◆ Whenever users wants to login, users sends login/password and server wants to remember that user authenticated for the future requests
- ◆ "Standard" solution
 - Server generates a long unique ID, creates a state file (state record) in the database, and set unique ID as a cookie for the client to repeat
 - "Bad guys" cannot easily guess such ID
- ◆ "Storage-Less" solution
 - Server encrypts/signs statement (user=x has been authenticated) and sets the whole encrypted blob as a cookie for the client
 - "Bad guys" cannot guess what's inside nor pretend that user=y is authenticated.

HTTPS: Secure HTTP

- ◆ Ensure that you're receiving data from place you think you do
- ◆ Ensure secrecy of information exchanges

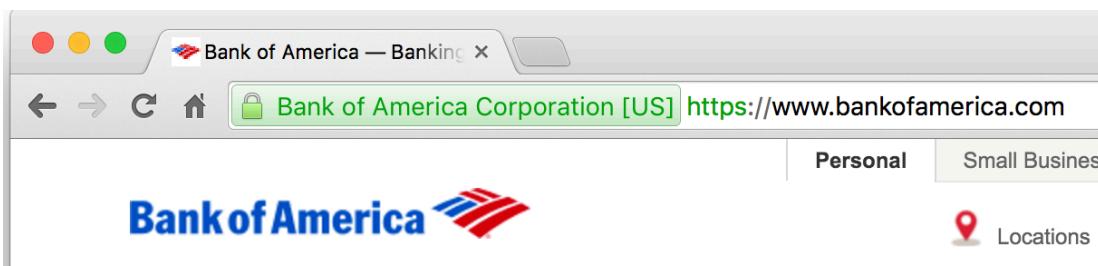


HTTPS: Sites with Domain Validation (DV) Certificates

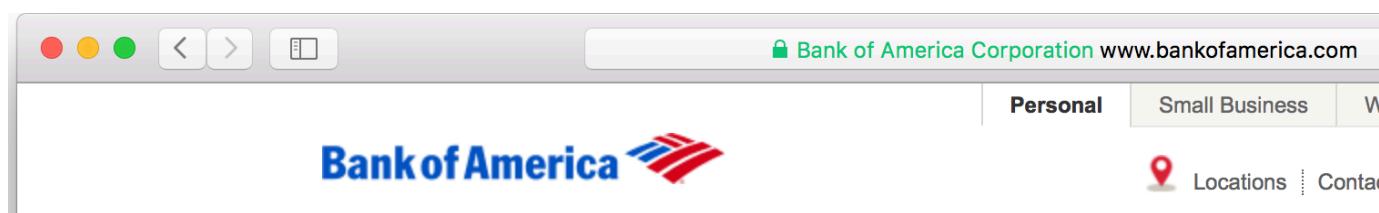


HTTPS: Sites with Extended Validation (EV) Certificates

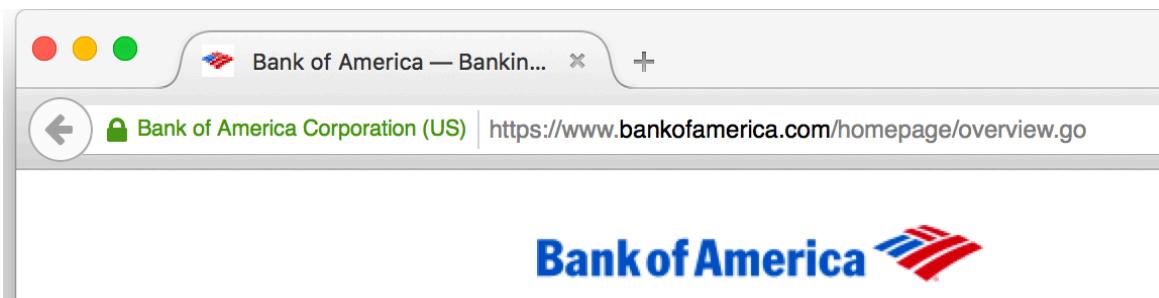
- the Certificate Authority (CA) checks the right of the applicant to use a specific domain name PLUS it conducts a THOROUGH vetting of the organization



Chrome

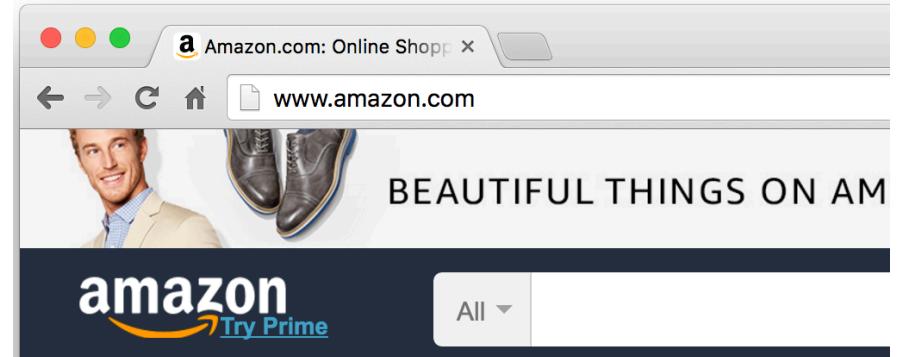
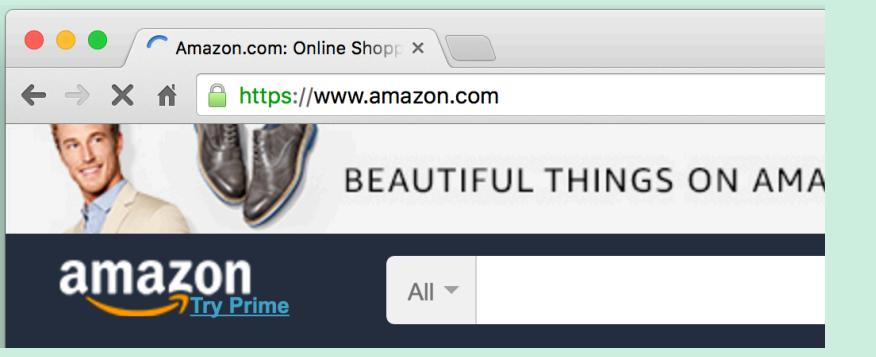
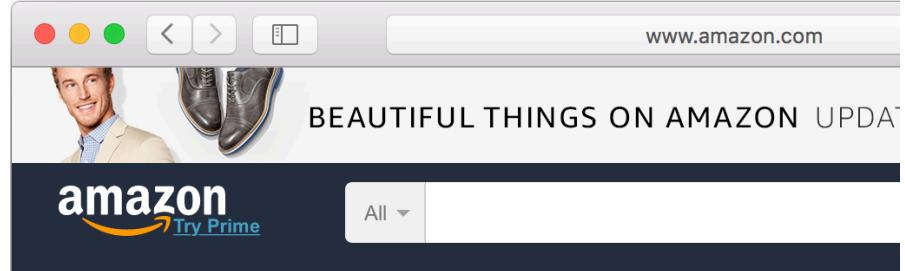
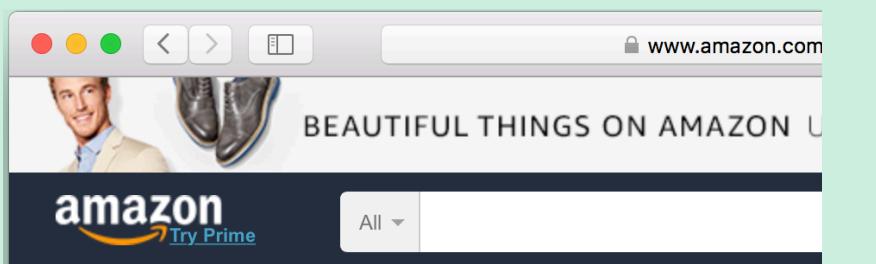
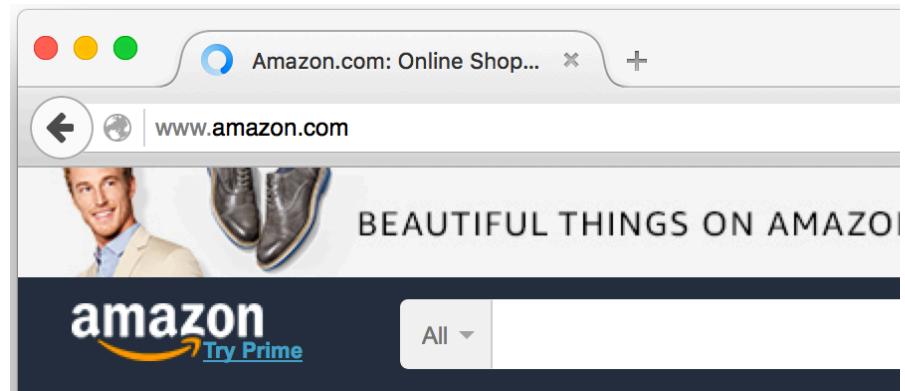
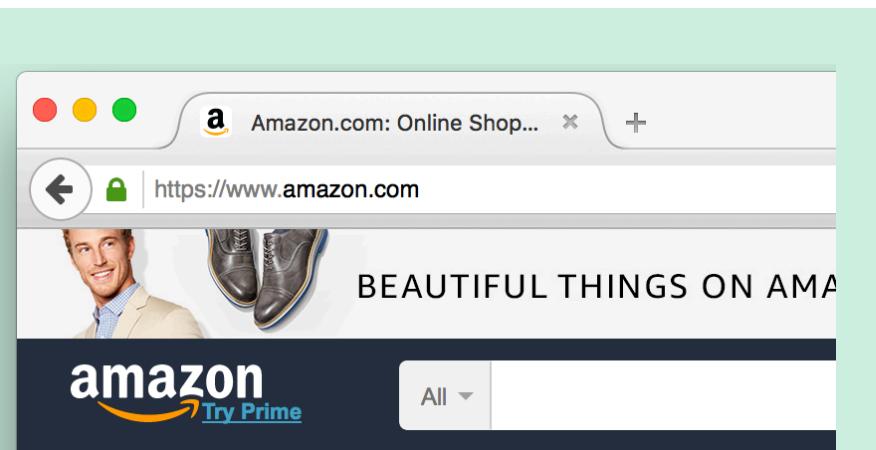


Safari



Firefox

Beware: Some Sites use HTTPS Optionally



HTTPS: Checking Certificate Manually

A screenshot of a web browser window displaying the SSL certificate information for www.amazon.com. The browser's address bar shows the URL <https://www.amazon.com>. A certificate dialog box is overlaid on the page, providing detailed information about the SSL certificate.

Certificate Details:

- Subject Name:** www.amazon.com
- Issued by:** Symantec Class 3 Secure Server CA - G4
- Expires:** Saturday, December 31, 2016 at 1:59:59 AM Eastern European Standard Time
- Status:** This certificate is valid

Issuer Details:

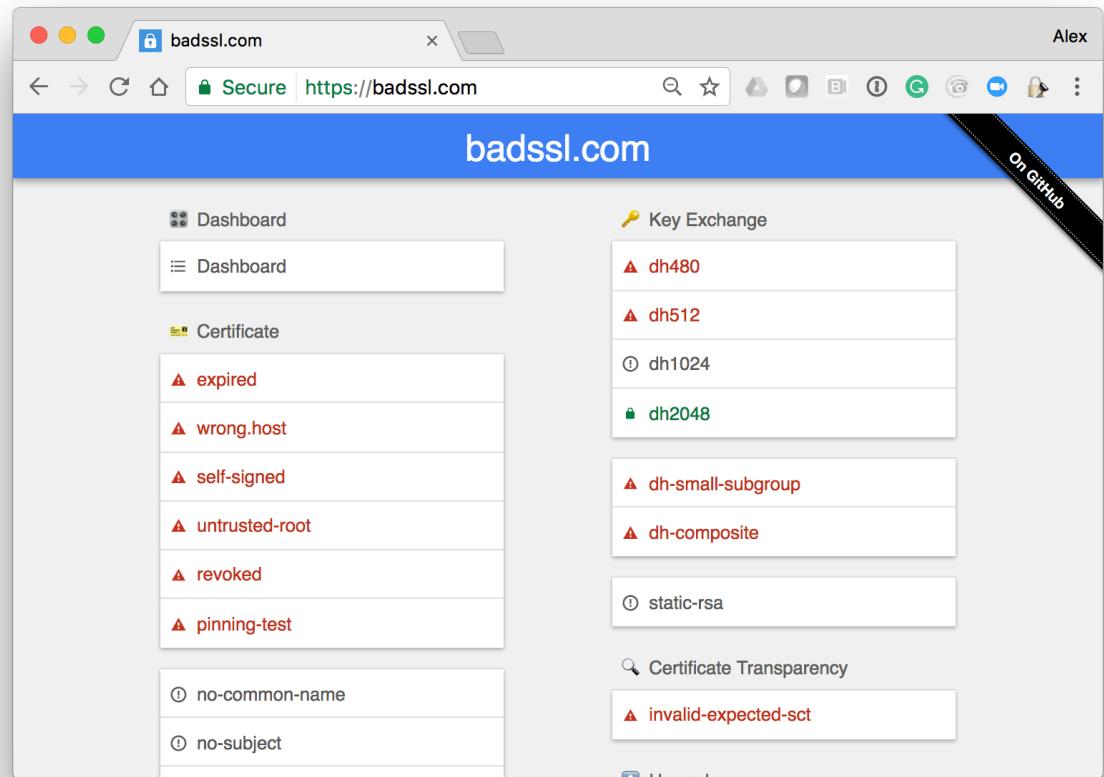
- Country:** US
- Organization:** Symantec Corporation
- Organizational Unit:** Symantec Trust Network
- Common Name:** Symantec Class 3 Secure Server CA - G4

OK

The browser's status bar at the bottom right indicates "SSL 256-bit" and "Secure". The title bar of the browser window says "Alex".

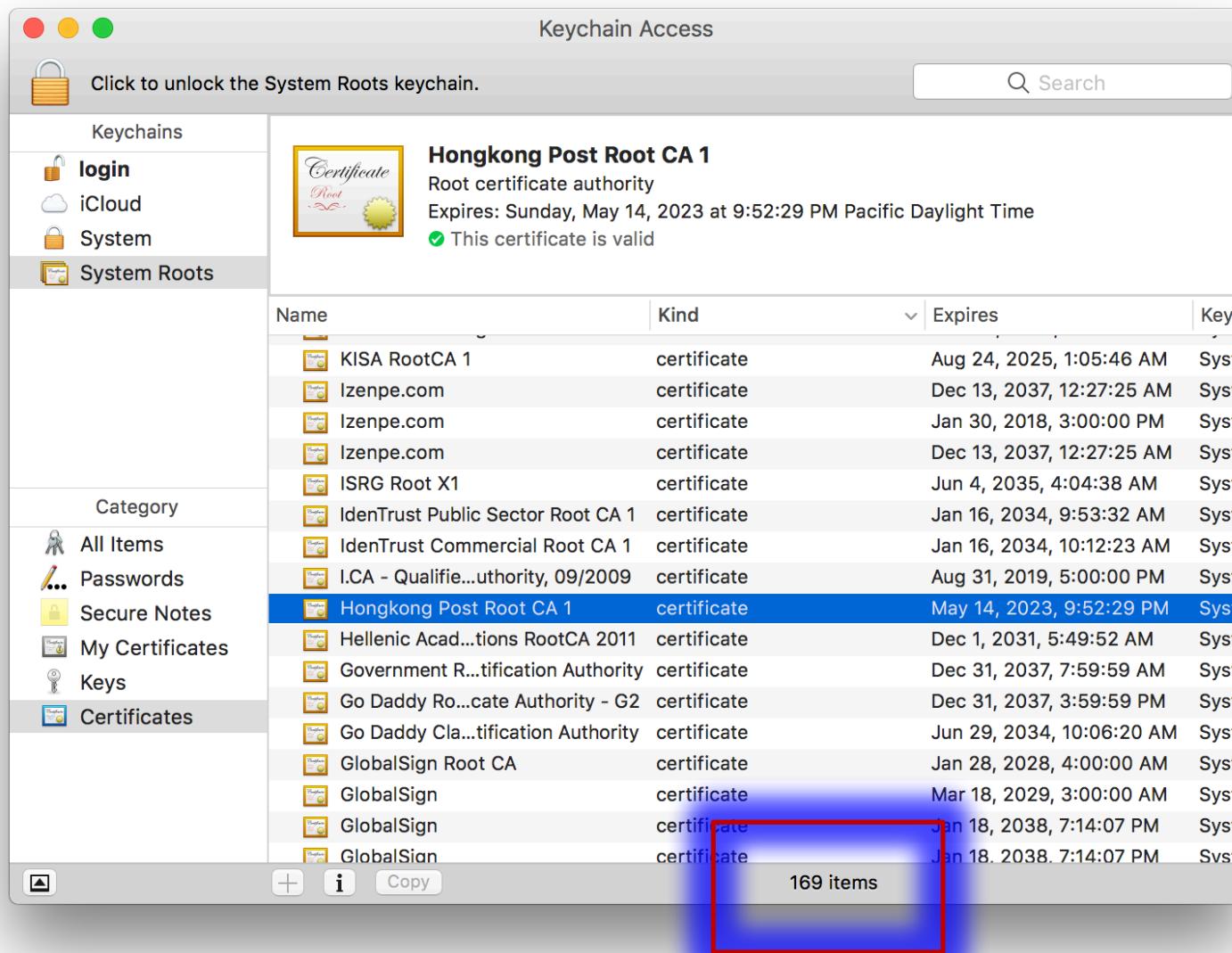
HTTPS (TLS) Certificate States

- ◆ Valid
- ◆ Expired
- ◆ Wrong host
- ◆ Untrusted root
- ◆ Revoked
- ◆ Pinned
- ◆ + more



Is your browser safe?

Whom You Trust?



E-Mail

Not so simple protocol(s)

E-Mail: 4 major components

- ◆ **User Agents:** a.k.a. “mail app”: composing, editing, reading mail messages

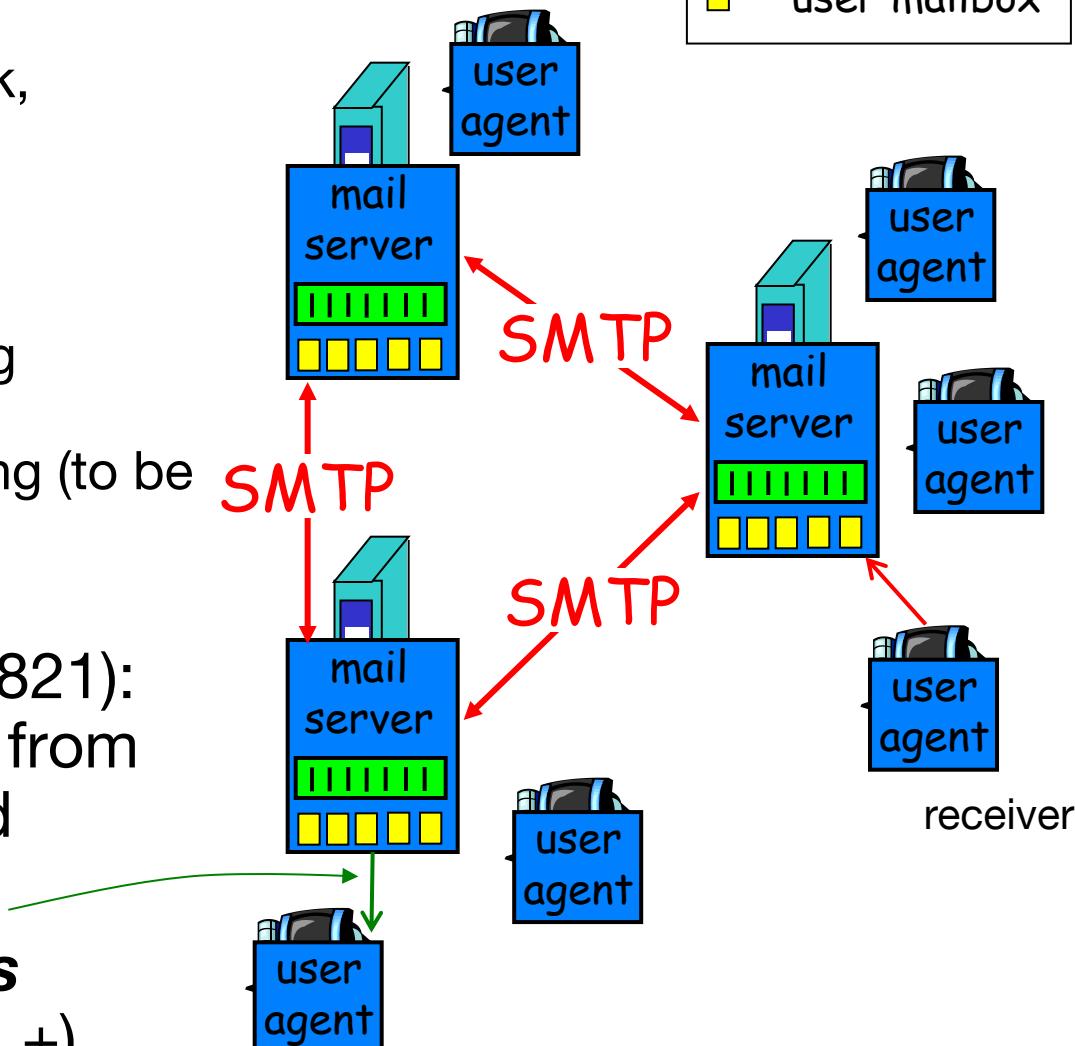
- Gmail, Apple Mail, Outlook, Thunderbird, etc.

- ◆ **Mail Servers:**

- (sendmail, postfix, etc.)
 - **mailbox** contains incoming messages for user
 - **message queue** of outgoing (to be sent) mail messages

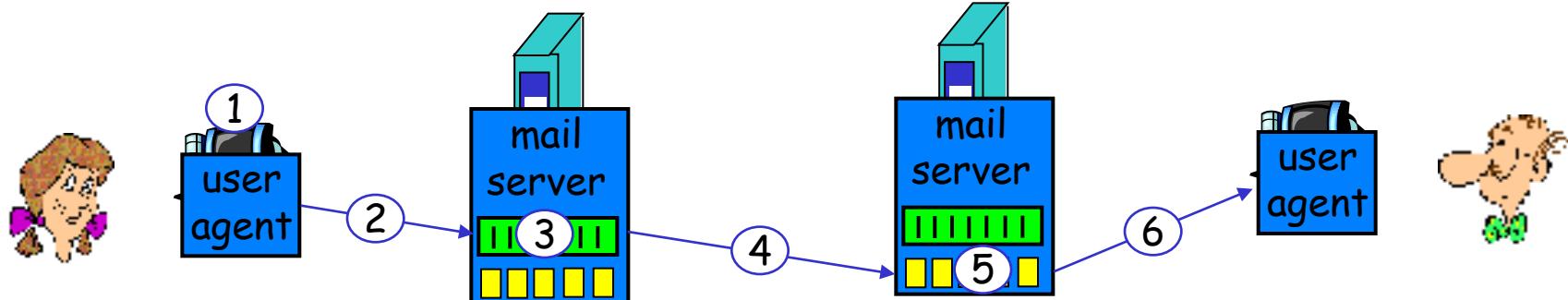
- ◆ **Simple Mail Transfer Protocol** (SMTP, RFC 2821): transfer email messages from user client to servers and *between mail servers*

- ◆ **Mail Retrieval Protocols** (IMAP, POP3, Exchange, +)

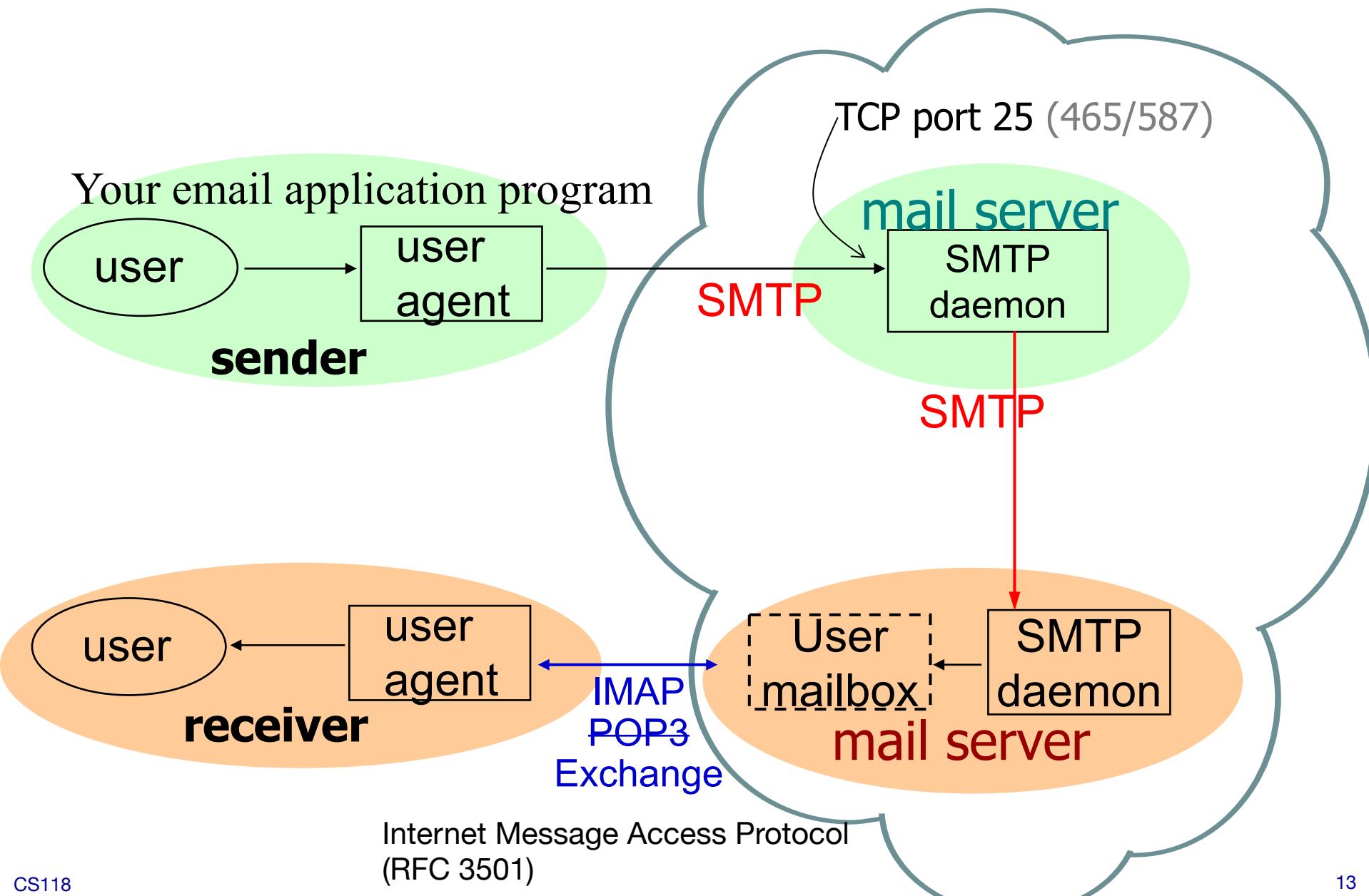


Example: Alice sends a message to Bob

- 1) Alice writes a msg “to” bob@someschool.edu
- 2) Alice’s UA sends message to her mail server; message placed in message queue
- 3) The mail server opens TCP connection with Bob’s mail server
- 4) Alice’s server sends the msg over the TCP connection
- 5) Bob’s mail server places the msg in Bob’s mailbox
- 6) Bob invokes his user agent to fetch message



Email delivery



Simple Mail Transfer Protocol [RFC 2821]

- ◆ Three phases of transfer
 - handshaking (greeting)
 - (starting secure session / STARTTLS)
 - transfer of messages
 - closing
 - Note this is *not* TCP connection close (to be covered in next lectures)
- ◆ Each phase: command/response interaction
 - commands: ASCII text
 - response: status code and phrase
- ◆ messages must be encoded in 7-bit ASCII

Example of Sending Email over SMTP

```
x 16:52 ~ $ telnet
zimbra.cs.ucla.edu 25
Trying 131.179.128.68...
Connected to zimbra.cs.ucla.edu.
Escape character is '^]'.
220 zimbra.cs.ucla.edu ESMTP
Postfix

ehlo cs.ucla.edu
250-zimbra.cs.ucla.edu
250-PIPELINING
250-SIZE 40000000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN

rcpt to: aa@cs.ucla.edu
250 2.1.5 0k

mail from: aa@cs.ucla.edu
250 2.1.0 0k

data
354 End data with <CR><LF>. <CR><LF>
Subject: Hello, world!

Hello, world! Duh.

.
250 2.0.0 0k: queued as EC81F160544

quit
221 2.0.0 Bye
Connection closed by foreign host.
```

A typical SMTP message exchange (after the TCP connection setup)

sender SMTP process

receiver SMTP process

220 zimbra.cs.ucla.edu ESMTP

EHLO cs.ucla.edu

250-zimbra.cs.ucla.edu

RCPT TO: aa@cs.ucla.edu

250 2.1.5 Ok

MAIL FROM: aa@cs.ucla.edu

250 2.1.0 Ok

DATA

354 End data with <CR><LF>.<CR><LF>

Subject: Hello, world!

...

can start
sending
another msg
at this point

<CRLF> • <CRLF>

CRLF: Carriage-Return,
LineFeed

250 2.0.0 Ok: queued as EC81F160544

QUIT

221 2.0.0 Bye

Reply Status Codes: Look Familiar?

Code meaning

220 service ready

221 I'm closing too

250 requested action OK

500 error, command not recognized

550 no such mbox, no action taken

Common practices

1st digit: whether response is good/bad/incomplete

e.g. 2= positive completion, 5=negative completion

2nd digit: encodes responses in specific categories

e.g. 2=connections, 5=mail system (status of the receiver mail system)

3rd digit: a finer gradation of meaning in each category specified by the 2nd digit.

SMTP: A Few Highlights

FYI

- ◆ SMTP server uses CRLF.CRLF to determine end of message
- ◆ SMTP msg body: use an empty line to separate header and body
- ◆ Message (header & body) must be encoded in 7-bit ASCII
- ◆ Multipurpose Internet Mail Extensions (MIME) to support
 - Header information or text in non-ASCII char. set
 - Non-text attachments
 - Message bodies with multiple parts

Find similarities with
HTTP

Comparison between SMTP and HTTP

- ◆ HTTP: pull
- ◆ SMTP: push
- ◆ both use ASCII command/response interaction, responses with status codes
 - Both also use empty line for header-body separation
- ◆ HTTP: each object encapsulated in its own response message
- ◆ SMTP: multiple objects can be encoded in one multipart message

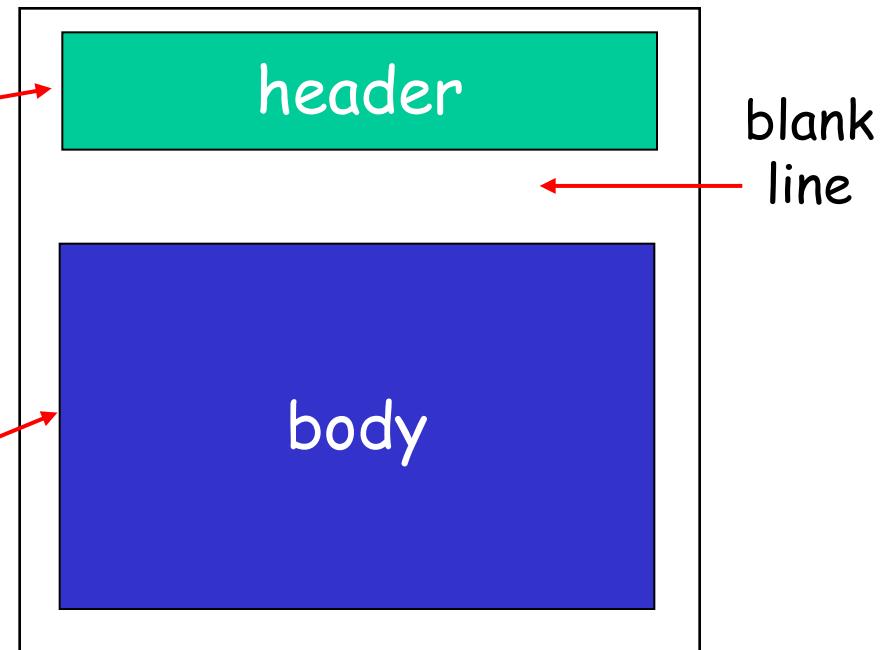
Mail message format

- ◆ RFC 822: standard for text message format:

- ◆ header lines, e.g.,

- To:
- From: } These are independent from what's used in SMTP
- Subject:

- ◆ body
 - the “message”, ASCII characters only



Multipurpose Internet Mail Extensions (MIME)

FYI

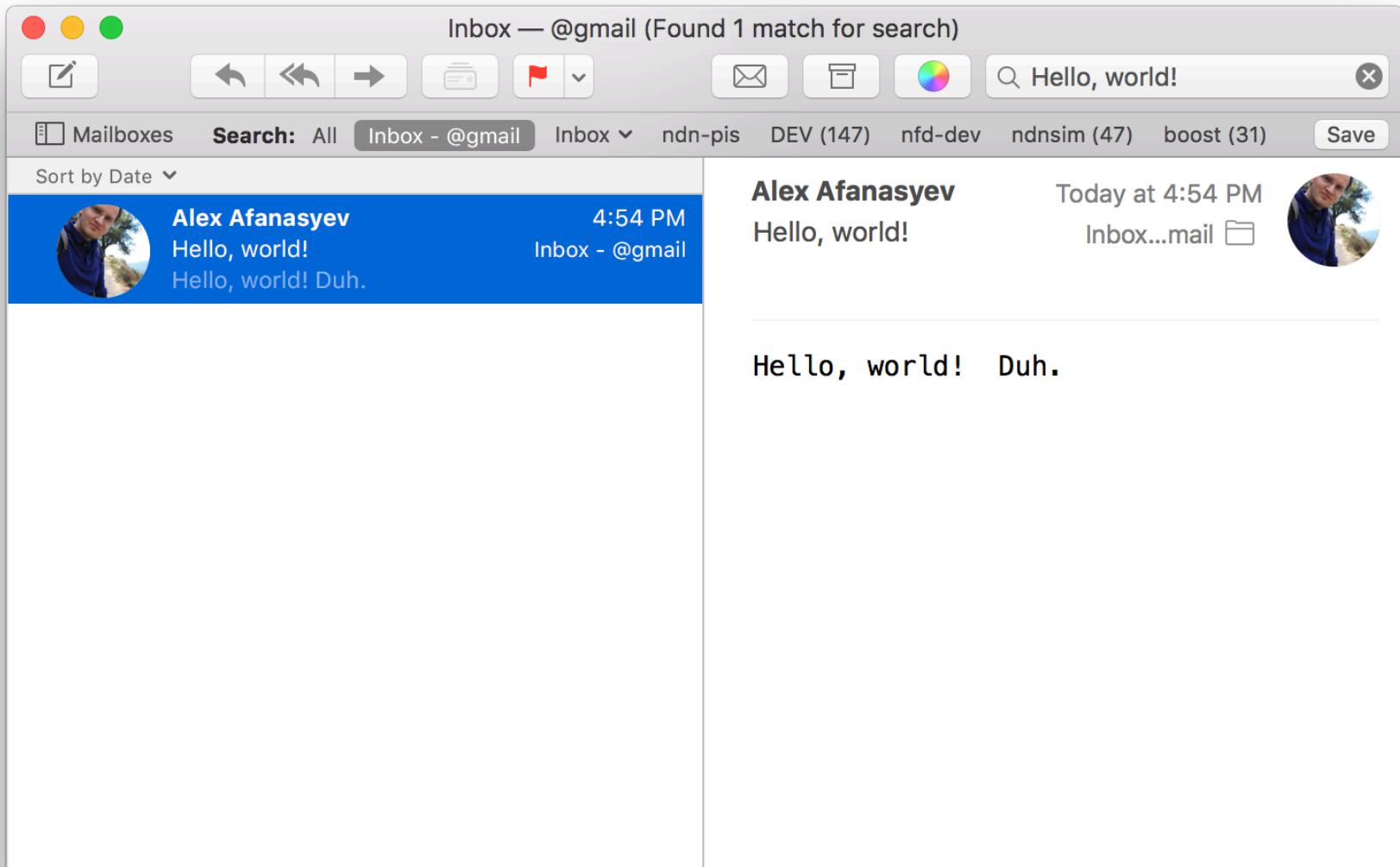
- ◆ To support
 - Header information or text in non-ASCII char. set
 - Non-text attachments
 - Message bodies with multiple parts
- ◆ additional lines in msg header for MIME content type

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
```

Here is my email



Received message (Cmd-Opt-U in Apple Mail)

Delivered-To: cawka1@gmail.com
Received: by 10.157.5.104 with SMTP id 95csp742023otw;
 Wed, 6 Apr 2016 07:54:12 -0700 (PDT)
X-Received: by 10.98.79.205 with SMTP id f74mr38989594pfj.68.1459954452037;
 Wed, 06 Apr 2016 07:54:12 -0700 (PDT)
Return-Path: <aa@cs.ucla.edu>
Received: from zimbra.cs.ucla.edu (zimbra.cs.ucla.edu. [131.179.128.68])
 by mx.google.com with ESMTPS id f90si5095225pff.83.2016.04.06.07.54.11
 for <cawka1@gmail.com>
 (version=TLS1_2 cipher=ECDHE-RSA-AES128-GCM-SHA256 bits=128/128);
 Wed, 06 Apr 2016 07:54:11 -0700 (PDT)
Received-SPF: pass (google.com: domain of aa@cs.ucla.edu designates 131.179.128.68 as permitted
sender) client-ip=131.179.128.68;
Authentication-Results: mx.google.com;
 spf=pass (google.com: domain of aa@cs.ucla.edu designates 131.179.128.68 as permitted
sender) smtp.mailfrom=aa@cs.ucla.edu
Received: from localhost (localhost [127.0.0.1])
 by zimbra.cs.ucla.edu (Postfix) with ESMTP id A445B16127A;
 Wed, 6 Apr 2016 07:54:11 -0700 (PDT)
Received: from zimbra.cs.ucla.edu ([127.0.0.1])
 by localhost (zimbra.cs.ucla.edu [127.0.0.1]) (amavisd-new, port 10032)
 with ESMTP id w90NfWSmRntW; Wed, 6 Apr 2016 07:54:11 -0700 (PDT)
Received: from localhost (localhost [127.0.0.1])
 by zimbra.cs.ucla.edu (Postfix) with ESMTP id 103E516126F;
 Wed, 6 Apr 2016 07:54:11 -0700 (PDT)

X-Virus-Scanned: amavisd-new at zimbra.cs.ucla.edu
Received: from zimbra.cs.ucla.edu ([127.0.0.1])
 by localhost (zimbra.cs.ucla.edu [127.0.0.1]) (amavisd-new, port 10026)
 with ESMTP id b3lcNgHG14vP; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
Received: from mailman.cs.ucla.edu (Mailman.CS.UCLA.EDU [131.179.128.30])
 by zimbra.cs.ucla.edu (Postfix) with ESMTP id EF118160544
 for <afanasev@zimbra.cs.ucla.edu>; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
Received: by mailman.cs.ucla.edu (Postfix)
 id EDA0D7D6F4; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
Delivered-To: aa@cs.ucla.edu
Received: from zimbra.cs.ucla.edu (zimbra.CS.UCLA.EDU [131.179.128.68])
 by mailman.cs.ucla.edu (Postfix) with ESMTP id E3BE27D6F3
 for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
Received: from localhost (localhost [127.0.0.1])
 by zimbra.cs.ucla.edu (Postfix) with ESMTP id 6100F16126F
 for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:54:09 -0700 (PDT)
X-Virus-Scanned: amavisd-new at zimbra.cs.ucla.edu
Received: from zimbra.cs.ucla.edu ([127.0.0.1])
 by localhost (zimbra.cs.ucla.edu [127.0.0.1]) (amavisd-new, port 10024)
 with ESMTP id dqevBkdxYCo9 for <aa@cs.ucla.edu>;
 Wed, 6 Apr 2016 07:54:08 -0700 (PDT)
Received: from cs.ucla.edu (wlan-141-23-116-69.tubit.tu-berlin.de [141.23.116.69])
 by zimbra.cs.ucla.edu (Postfix) with ESMTP id EC81F160544
 for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:53:23 -0700 (PDT)

Subject: Hello, world!

Message-Id: <20160406145331.EC81F160544@zimbra.cs.ucla.edu>
Date: Wed, 6 Apr 2016 07:53:23 -0700 (PDT)
From: aa@cs.ucla.edu

Hello, world! Duh.

What Happened?

- Me → zimbra.cs.ucla.edu

```
Received: from cs.ucla.edu (wlan-141-23-116-69.tubit.tu-berlin.de [141.23.116.69])
          by zimbra.cs.ucla.edu (Postfix) with ESMTP id EC81F160544
          for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:53:23 -0700 (PDT)
```

- zimbra → localhost (virus scan)

```
Received: from zimbra.cs.ucla.edu ([127.0.0.1])
          by localhost (zimbra.cs.ucla.edu [127.0.0.1]) (amavisd-new, port 10024)
          with ESMTP id dgevBkdxYCo9 for <aa@cs.ucla.edu>;
          Wed, 6 Apr 2016 07:54:08 -0700 (PDT)
```

- localhost → zimbra.cs.ucla.edu (?)

```
Received: from localhost (localhost [127.0.0.1])
          by zimbra.cs.ucla.edu (Postfix) with ESMTP id 6100F16126F
          for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:54:09 -0700 (PDT)
```

- zimbra → mailman.cs.ucla.edu (local delivery)

```
Received: from zimbra.cs.ucla.edu (zimbra.CS.UCLA.EDU [131.179.128.68])
          by mailman.cs.ucla.edu (Postfix) with ESMTP id E3BE27D6F3
          for <aa@cs.ucla.edu>; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
```

- mailman → zimbra (redirecting)

```
Received: from mailman.cs.ucla.edu (Mailman.CS.UCLA.EDU [131.179.128.30])
          by zimbra.cs.ucla.edu (Postfix) with ESMTP id EF118160544
          for <afanasev@zimbra.cs.ucla.edu>; Wed, 6 Apr 2016 07:54:10 -0700 (PDT)
```

What Happened? (cont.)

- ◆ zimbra → localhost (virus scan)
- ◆ localhost → zimbra
- ◆ zimbra → localhost
- ◆ localhost → zimbra
- ◆ zimbra → mx.google.com
- ◆ → 10.98.79.205
- ◆ → 10.157.5.104

How to send emails?

- ◆ SMTP servers for the domain normally allow emails only for this domain
- ◆ Use “provider’s” SMTP service
 - Allow customers to send email to any domain name
- ◆ Use your email provider SMTP service with authentication
 - Google’s SMTP
 - Yahoo SMTP

Other way to send email?

- ◆ Contact domain's SMTP directly
- ◆ Drawbacks?

Securing channel to SMTP server

- ◆ Remember HTTPS
- ◆ Similar can/should be used for SMTP
 - Don't allow others to see your login password

```
openssl s_client -starttls smtp -crlf -connect  
smtp.gmail.com:25
```

EHLO testing

AUTH LOGIN

```
base64('username')  
base64('password')
```

Does secure channel between SMTP makes email secure?

Additional Security Layers in SMTP

- ◆ DKIM
 - https://en.wikipedia.org/wiki/DomainKeys_Identified_Mail
 - Allows the receiver to check that an email claimed to have come from a specific domain was indeed authorized by the owner of that domain
- ◆ SPF
 - https://en.wikipedia.org/wiki/Sender_Policy_Framework
 - A mechanism to check that incoming mail from a domain comes from a host authorized by that domain's administrators

Is your email secure?

- ◆ No
- ◆ Why?

Is there away to secure your email?

- ♦ How to ensure that others know your email is indeed from you?
 - Solve challenges I posted on Piazza yesterday.

Securing Email

- ◆ Sign with a valid certificate
 - issued by some certification authority
 - can check and vouch that it is indeed you who sent the email
 - can simply check you own the email
 - ▲ <https://startssl.com/> (free one)
 - ▲ many others
 - Web-of-Trust certificate (PGP, GPG)
 - others can vouch for you
- ◆ Encrypt with somebody else's public key
 - How can you get this public key?

Getting started with PGP

- ◆ OSX
 - <https://gpgtools.org/>
- ◆ Windows
 - <https://www.gpg4win.org/>
- ◆ Linux
 - <https://help.ubuntu.com/community/GnuPrivacyGuardHowto>

GPG Keychain (OSX)

GPG Keychain

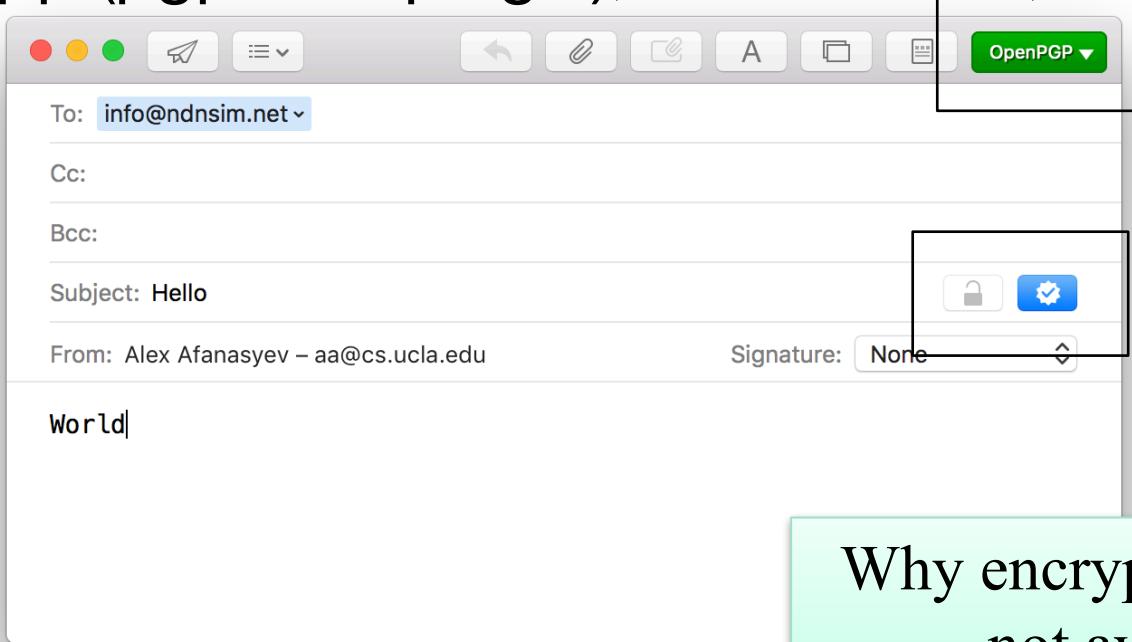
The screenshot shows the GPG Keychain application window. At the top, there are icons for New, Import, Export, Lookup Key, and Delete. To the right of the title bar are an information icon, a search bar with the placeholder "Search", and buttons for "Details" and "Filter". The main area is a table displaying 19 public keys. The columns are: Type, Name, Email, Created, Key ID, and Validity (represented by three colored squares). The table rows are as follows:

Type	Name	Email	Created	Key ID	Validity
pub	Alexander Afanasyev	alexander.afanasyev@ucla.edu	4/4/11	0A95AC3E	[Green, Green, Green]
sec/pub	Alexander Afanasyev	alexander.afanasyev@ucla.edu	11/5/13	5E8570C9	[Green, Green, Green]
pub	Audrey Tang	cpan@audreyt.org	4/10/01	3C3501A0	[Yellow, Yellow, Grey]
pub	Christopher Hunt	chunt@reachone.com	1/28/11	DD862A15	[Red, Grey, Grey]
pub	David R. Oran	oran@cisco.com	9/9/03	54DD2AE6	[Yellow, Yellow, Grey]
pub	Dmitry Moskalchuk	dm@crystax.net	1/26/15	9CFB90C3	[Yellow, Yellow, Grey]
pub	Flávio de Queiroz Guimarães	flavioqueiroz2004@hotmail.com	4/10/13	7293742E	[Yellow, Yellow, Grey]
pub	GPGTools Team	team@gpgtools.org	8/19/10	00D026C4	[Green, Green, Green]
pub	Jairo Eduardo López	jelfn@sislacom.com	11/16/15	C9ACC8C2	[Yellow, Yellow, Grey]
pub	Jamie Strandboge	jamie@strandboge.com	9/30/10	CC559573	[Yellow, Yellow, Grey]
pub	John G. Kemp	kemp@network-services.uoregon.edu	3/3/04	A3CA7130	[Yellow, Yellow, Grey]
pub	Kees Cook	kees@outflux.net	9/27/10	DC6DC026	[Yellow, Yellow, Grey]
pub	Lennart Schulte	lennart.schulte@rwth-aachen.de	3/12/11	4534937E	[Red, Grey, Grey]
pub	Lucas Wang	lucas@cs.ucla.edu	2/24/11	98170B0B	[Yellow, Yellow, Grey]
pub	PAUSE Batch Signing Key 2015	pause@pause.perl.org	2/3/03	450F89EC	[Red, Grey, Grey]
pub	Richard James Laager	rlaager@wiktel.com	12/1/00	5E1F1BCE	[Yellow, Yellow, Grey]
pub	Stefan Winter	stefan.winter@restena.lu	9/6/13	8A39DC66	[Yellow, Yellow, Grey]
pub	Steve Rieger	riegersteve@gmail.com	12/3/14	191CE12A	[Red, Grey, Grey]
pub	Yingdi Yu	yingdi@cs.ucla.edu	7/31/13	F4A45863	[Green, Green, Green]

19 of 19 keys listed Show secret keys only

Signing outgoing emails

- ◆ Manually
 - Gmail and most others web-based systems ☹
 - You can sign up with <https://Keybase.io>
- ◆ Automatically
 - Mail.app (pgp with plugin), Thunderbird, Outlook, etc.



Why encryption option is
not available?

Manual signing of things

✓ 23:11 ~ \$ **gpg --clearsign**

You need a passphrase to unlock the secret key for
user: "Alexander Afanasyev <alexander.afanasyev@ucla.edu>"
4096-bit RSA key, ID 5E8570C9, created 2013-11-05

Hello class,

This is definitely me who wrote this text.

-----BEGIN PGP SIGNED MESSAGE-----

Hash: SHA256

Hello class,

This is definitely me who wrote this text.

-----BEGIN PGP SIGNATURE-----

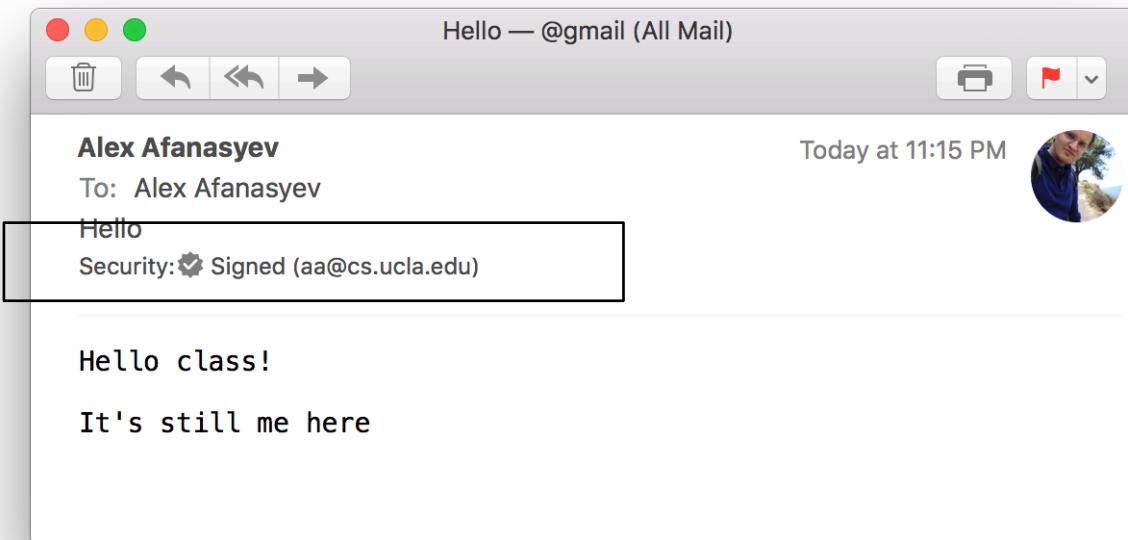
Comment: GPGTools - <http://gpgtools.org>

iQIcBAEBCAAGBQJXBXusAAoJEJFqoM1ehXDJ02wQAKby89g0282X0PDdVKR00yAu
BR/9Ztd1x+fWNJPZKbDVJ7e6oN4S6ggk/ZUJ1XXJxg5PFdGLGinTHp/vgH4oU0pUG+YNRUYLb3olrkvaJlCKKLfA+0dVGhhoCiv0Z1zH9wBi+zVv70bPoXFwbLf0ezQ8
yrsgB/e0etTI3A2hAP17+RvSlXE+IWbryiJh/6nsU/u2CsjE/yTjdjrJXDYcIKEN
/a2fhxqcbvwoT1kNgBa5Z9fntd+/1+qCsJcRqeF4ok6b2w8CB+TApec6JXRkuzsR
fMvNbLvKbV3DTpdYWTPAD0yP6tEfzd5oG0Mah5gZzwhVAfyqf9ETUBx8cVQ+QmDU
KJ7oFlWExKYiSoPuB3pep1XjHYWQkejdyhRp4w0U0GbR0JkPYlu1b3s+X0TNu3Ng
JZIYP0FJ0E1L8PM3zTMuMKfIIbHVP0tjJlncNSJrt6IVHjkJrDloSN08bBXr29R3
X+BoRid51+j3mQ43yK9wub1bi81tPX2fo++HLMg7AQBBW6pv3eZR5fomaN4xfH0e
SRyBwy5EANDiS2JWNvjPBlFzrhcNsSsp0qAh2+gZW9ZrPo4yZ3KK/SVrW0SrNq/5
j9Y/Nh6aJr0kk03dE/pyDFmuLJ3m9FDt/DhiIbcx6RJiKQISQx3pt0ERXEiRu4qy
FBiIFKZAQ4GwrPyF8Fod
=tlZn

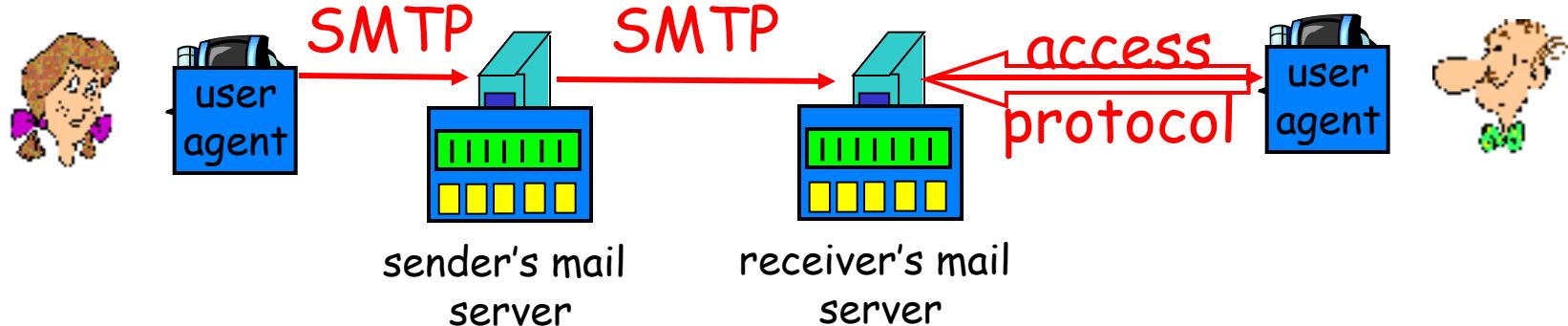
-----END PGP SIGNATURE-----

Verification

- ◆ Manually
 - gpg + paste text and signature
- ◆ Automatically



Mail access protocols



- ◆ SMTP: deliver mail to receiver's server
- ◆ Mail access protocol: retrieval from server
 - POP: Post Office Protocol [RFC 1939]
 - authorization (agent <→ server) and download
 - IMAP: Internet Mail Access Protocol [RFC 1730]
 - more features (e.g. multiple folders)
 - manipulation of stored messages on server
 - HTTP: gmail, Hotmail , Yahoo! Mail

POP3 protocol

FYI

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 and IMAP

More about POP3

- ◆ Previous example uses “download and delete” mode.
- ◆ Bob cannot re-read e-mail if he changes client
- ◆ “Download-and-keep”: copies of messages on different clients
- ◆ POP3 is stateless across sessions

IMAP

- ◆ Keep all messages in one place: the server
- ◆ Allows user to organize messages in folders
- ◆ IMAP keeps user state across sessions:
 - names of folders and mappings between message IDs and folder name

What You Learned Today

- ◆ Email
 - Even more complicated
 - Need to use signatures/encryption to make sure your communication is safe
 - **! be cautious of any emails, even if it is from your friends unless they are signed or you expect those emails !**
 - And I haven't mentioned all the security measures added to try to limit spam and fraud