Networks Lecture 4

Paolo Ciancarini

Innopolis University

January 28, 2022

Source of the material

- This lecture is based on the following resources
  - Chapter 2 of Computer Networking: A Top Down Approach by J Kurose and K Ross
  - https://www.diffen.com/difference/TCP_vs_UDP
  - The material is aligned and add/deleted according to the need of the students.

Topic of the lecture

More Internet applications

- Electronic Mail
- SMTP
- POP3
- IMAP
- FTP

Topic of the tutorial

- Email Protocol Suit
- TCP Socket Programming

Topic of the lab

- TCP Socket Programming

An application-layer protocol defines:

- types of messages exchanged,
  - e.g., request, response
- message syntax:
  - what fields in messages & how fields are delineated
- message semantics
  - meaning of information in fields
- rules for when and how processes send & respond to messages

open protocols:

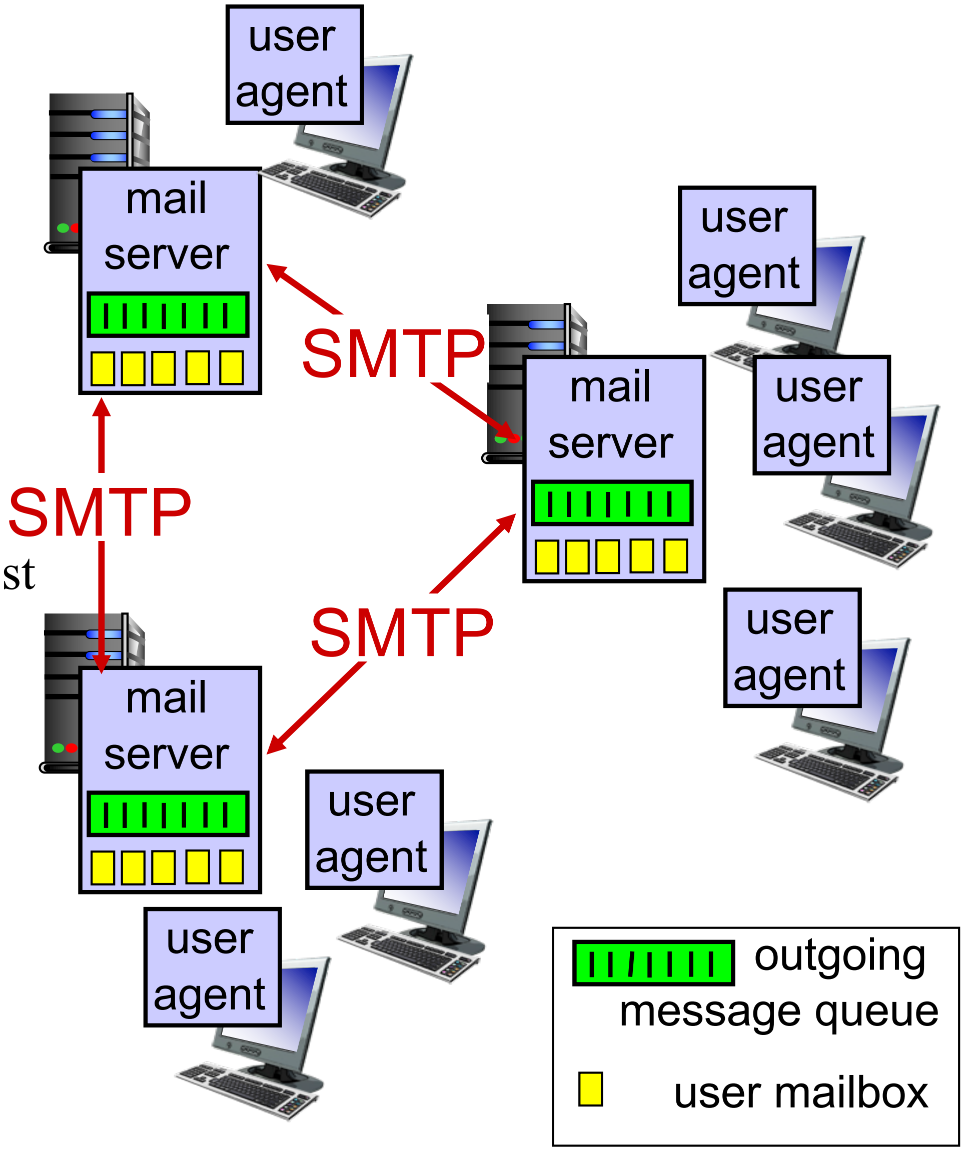
- defined in RFCs, everyone has access to protocol definition
- allows for interoperability
- e.g., HTTP, SMTP

proprietary protocols:

- e.g., Skype, Zoom

Electronic mail

- Three major components:
  - User agents
  - Mail servers
  - SMTP (Simple mail transfer protocol)

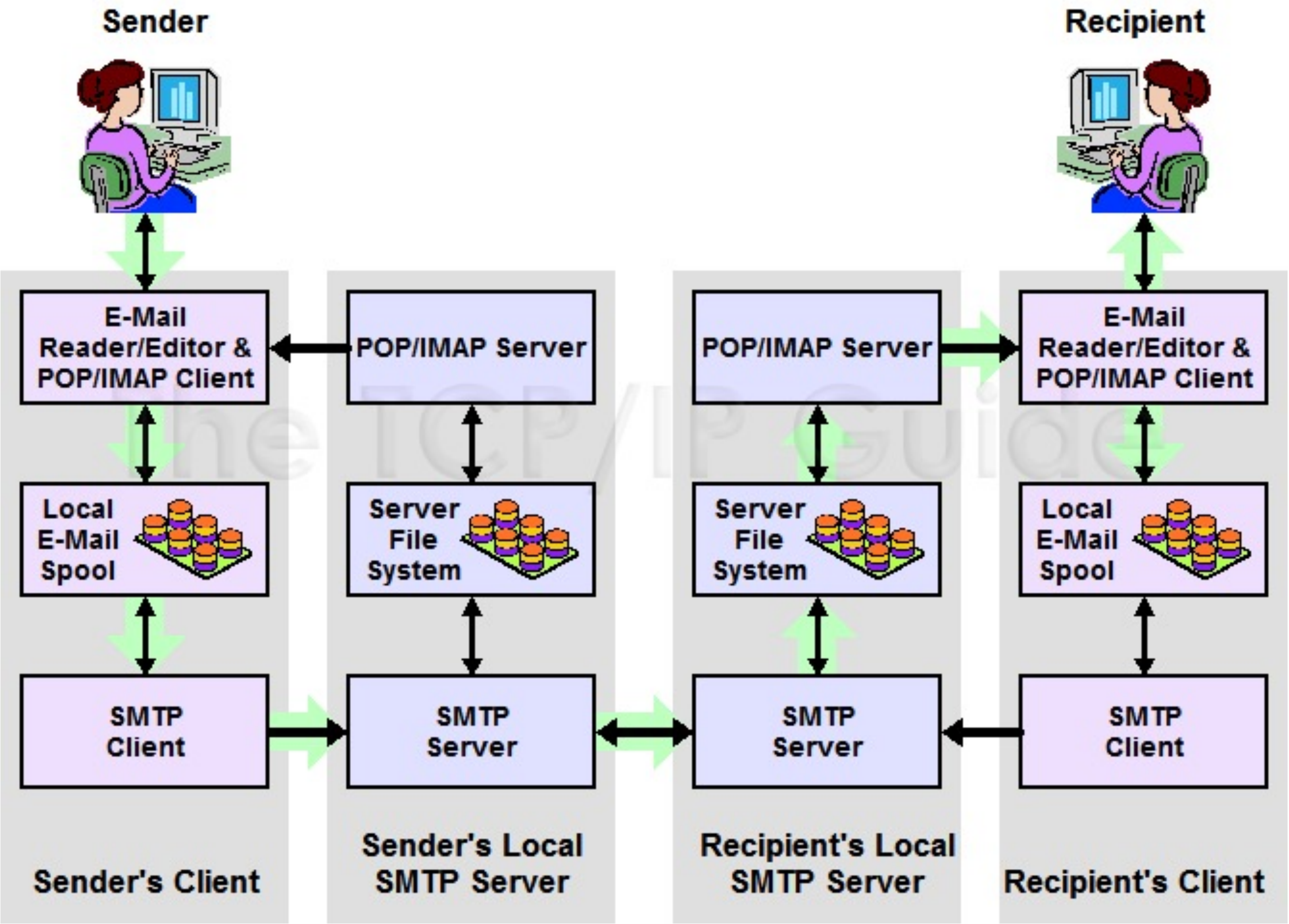


NOTE: We can send email to any mail server.

- Mail server is not obliged to accept every request
  - Security issue

https://www.fastmail.help/hc/en-us/articles/1500000278382-Email-standards

Sending and receiveing e-mail



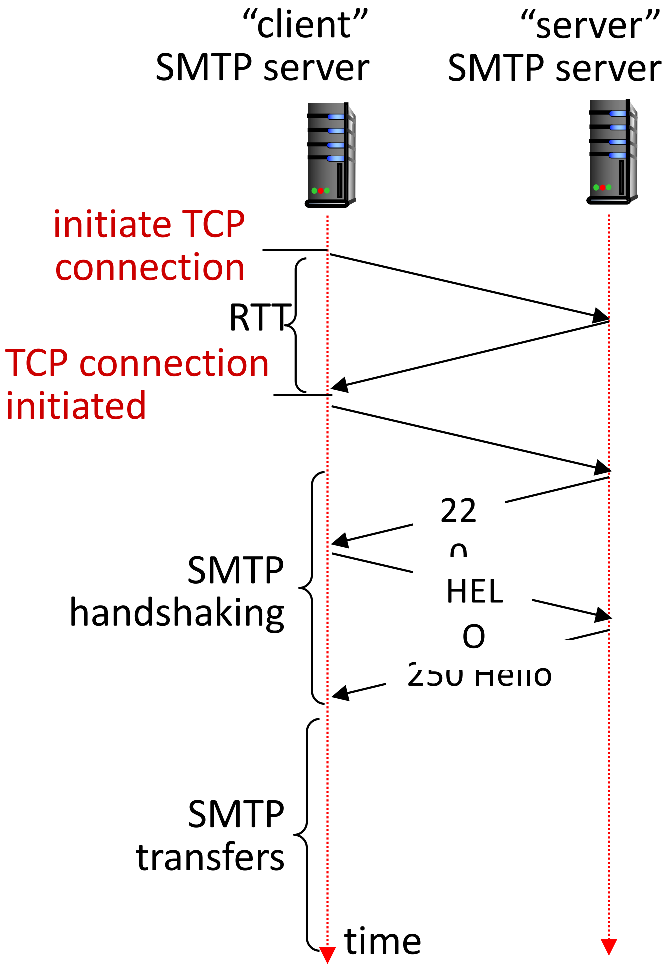
Electronic mail

- User Agents
  - Also known as "mail reader"
  - User agent can compose, edit, and read mail messages
    - For Example: Outlook, Thunderbird, iPhone mail client
  - Outgoing, incoming messages stored on server

- 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

SMTP RFC (5321)

- uses TCP to reliably transfer email message from client (mail server initiating connection) to server, port 25
  - direct transfer: sending server (acting like client) to receiving server
- three phases of transfer
  - SMTP handshaking (greeting)
  - SMTP transfer of messages
  - SMTP closure
- command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase



“client” SMTP server

“server” SMTP server

initiate TCP connection

RTT

TCP connection initiated

SMTP handshaking

22
0

HEL
O

250 Hello

SMTP transfers

time

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

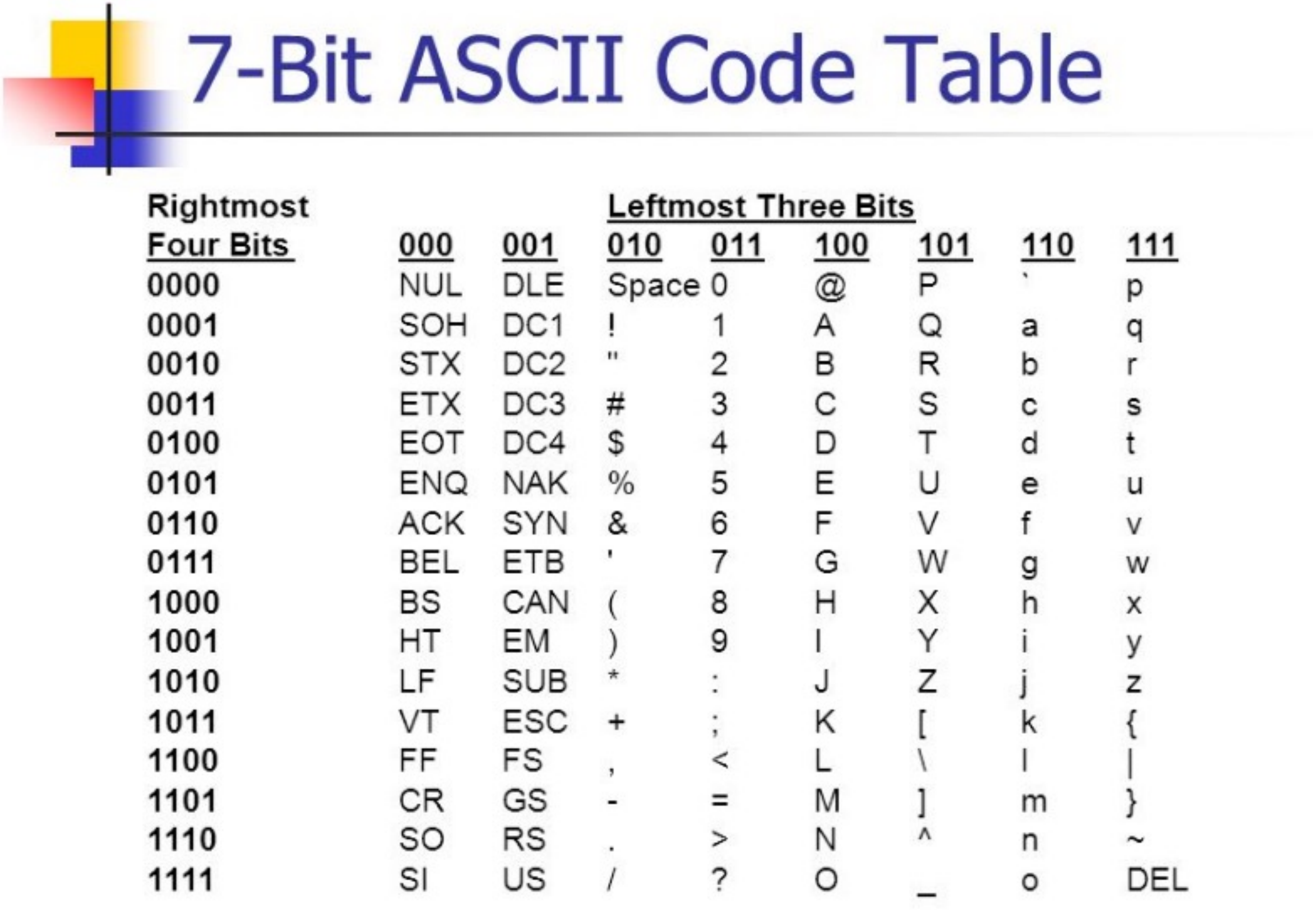
- Three phases of transfer
  - Handshaking (greeting)
  - Transfer of messages
  - Closure

- Command/response interaction (like HTTP, FTP)
  - **Commands:** ASCII text
  - **Response:** status code and phrase

- Messages must be in 7-bit ASCII

ASCII = American Standard Code for Information Interchange

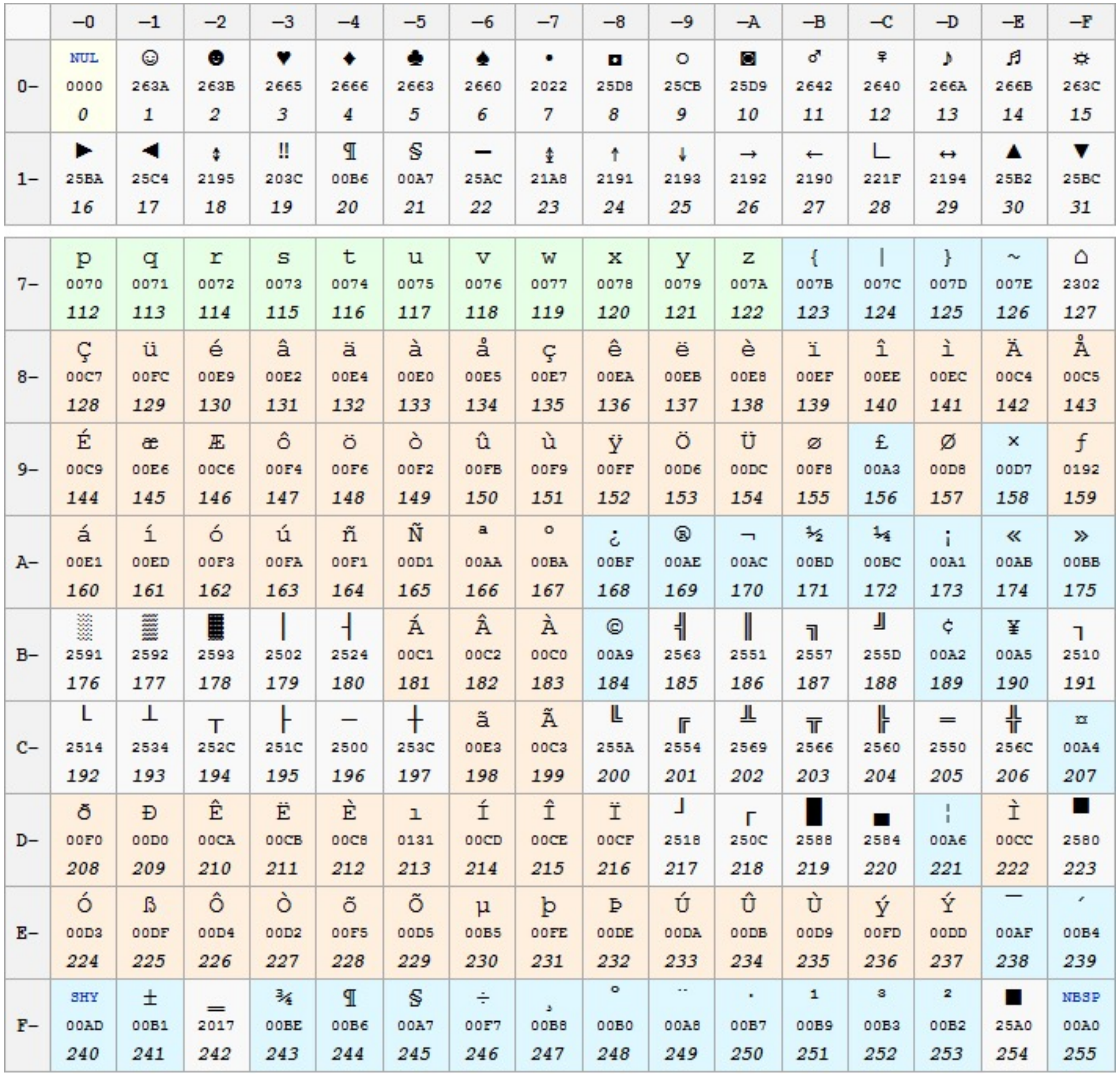
7bit ASCII



7-Bit ASCII Code Table

Rightmost			Leftmost Three Bits					
Four Bits	000	001	010	011	100	101	110	111
0000	NUL	DLE	Space	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

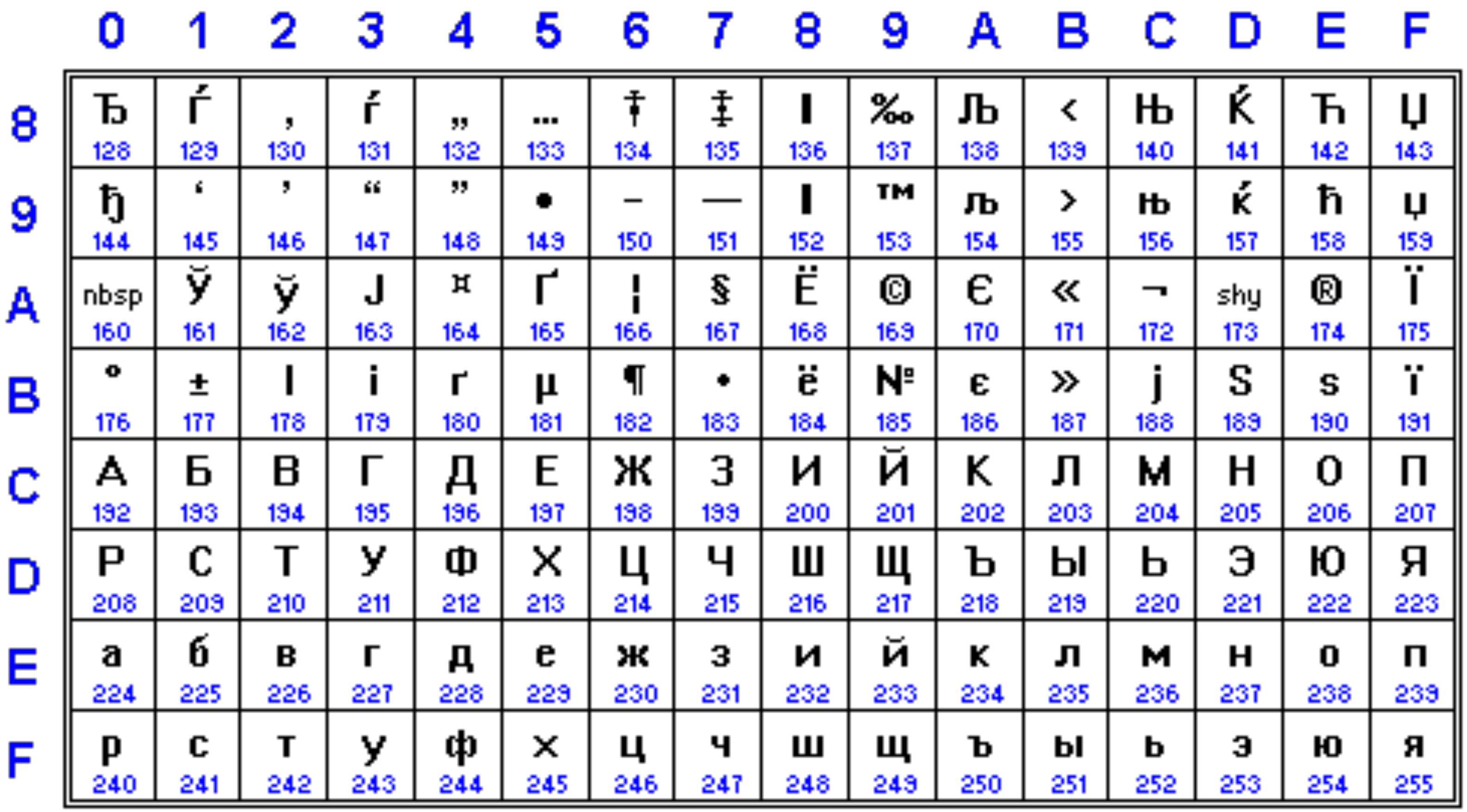
8 bit ASCII Italian



	–0	–1	–2	–3	–4	–5	–6	–7	–8	–9	–A	–B	–C	–D	–E	–F
0–	NUL 0000 0	☺ 263A 1	☻ 263B 2	♥ 2665 3	♦ 2666 4	♣ 2663 5	♠ 2660 6	• 2022 7	◘ 25D8 8	○ 25CB 9	◙ 25D9 10	♂ 2642 11	♀ 2640 12	♪ 266A 13	♫ 266B 14	☼ 263C 15
1–	► 25BA 16	◄ 25C4 17	↕ 2195 18	‼ 203C 19	¶ 00B6 20	§ 00A7 21	▬ 25AC 22	↨ 21A8 23	↑ 2191 24	↓ 2193 25	→ 2192 26	← 2190 27	∟ 221F 28	↔ 2194 29	▲ 25B2 30	▼ 25BC 31
7–	p 0070 112	q 0071 113	r 0072 114	s 0073 115	t 0074 116	u 0075 117	v 0076 118	w 0077 119	x 0078 120	y 0079 121	z 007A 122	{ 007B 123	| 007C 124	} 007D 125	~ 007E 126	⌂ 2302 127
8–	Ç 00C7 128	ü 00FC 129	é 00E9 130	â 00E2 131	ä 00E4 132	à 00E0 133	å 00E5 134	ç 00E7 135	ê 00EA 136	ë 00EB 137	è 00E8 138	ï 00EF 139	î 00EE 140	ì 00EC 141	Ä 00C4 142	Å 00C5 143
9–	É 00C9 144	æ 00E6 145	Æ 00C6 146	ô 00F4 147	ö 00F6 148	ò 00F2 149	û 00FB 150	ù 00F9 151	ÿ 00FF 152	Ö 00D6 153	Ü 00DC 154	ø 00F8 155	£ 00A3 156	Ø 00D8 157	× 00D7 158	ƒ 0192 159
A–	á 00E1 160	í 00ED 161	ó 00F3 162	ú 00FA 163	ñ 00F1 164	Ñ 00D1 165	ª 00AA 166	º 00BA 167	¿ 00BF 168	® 00AE 169	¬ 00AC 170	½ 00BD 171	¼ 00BC 172	¡ 00A1 173	« 00AB 174	» 00BB 175
B–	░ 2591 176	▒ 2592 177	▓ 2593 178	│ 2502 179	┤ 2524 180	Á 00C1 181	Â 00C2 182	À 00C0 183	© 00A9 184	╣ 2563 185	║ 2551 186	╗ 2557 187	╝ 255D 188	¢ 00A2 189	¥ 00A5 190	┐ 2510 191
C–	└ 2514 192	┴ 2534 193	┬ 252C 194	├ 251C 195	─ 2500 196	┼ 253C 197	ã 00E3 198	Ã 00C3 199	╚ 255A 200	╔ 2554 201	╩ 2569 202	╦ 2566 203	╠ 2560 204	═ 2550 205	╬ 256C 206	¤ 00A4 207
D–	ð 00F0 208	Ð 00D0 209	Ê 00CA 210	Ë 00CB 211	È 00C8 212	ı 0131 213	Í 00CD 214	Î 00CE 215	Ï 00CF 216	┘ 2518 217	┌ 250C 218	█ 2588 219	▄ 2584 220	¦ 00A6 221	Ì 00CC 222	▀ 2580 223
E–	Ó 00D3 224	ß 00DF 225	Ô 00D4 226	Ò 00D2 227	õ 00F5 228	Õ 00D5 229	µ 00B5 230	þ 00FE 231	Þ 00DE 232	Ú 00DA 233	Û 00DB 234	Ù 00D9 235	ý 00FD 236	Ý 00DD 237	¯ 00AF 238	´ 00B4 239
F–	SHY 00AD 240	± 00B1 241	‗ 2017 242	¾ 00BE 243	¶ 00B6 244	§ 00A7 245	÷ 00F7 246	¸ 00B8 247	° 00B0 248	¨ 00A8 249	· 00B7 250	¹ 00B9 251	³ 00B3 252	² 00B2 253	■ 25A0 254	NBSP 00A0 255

]

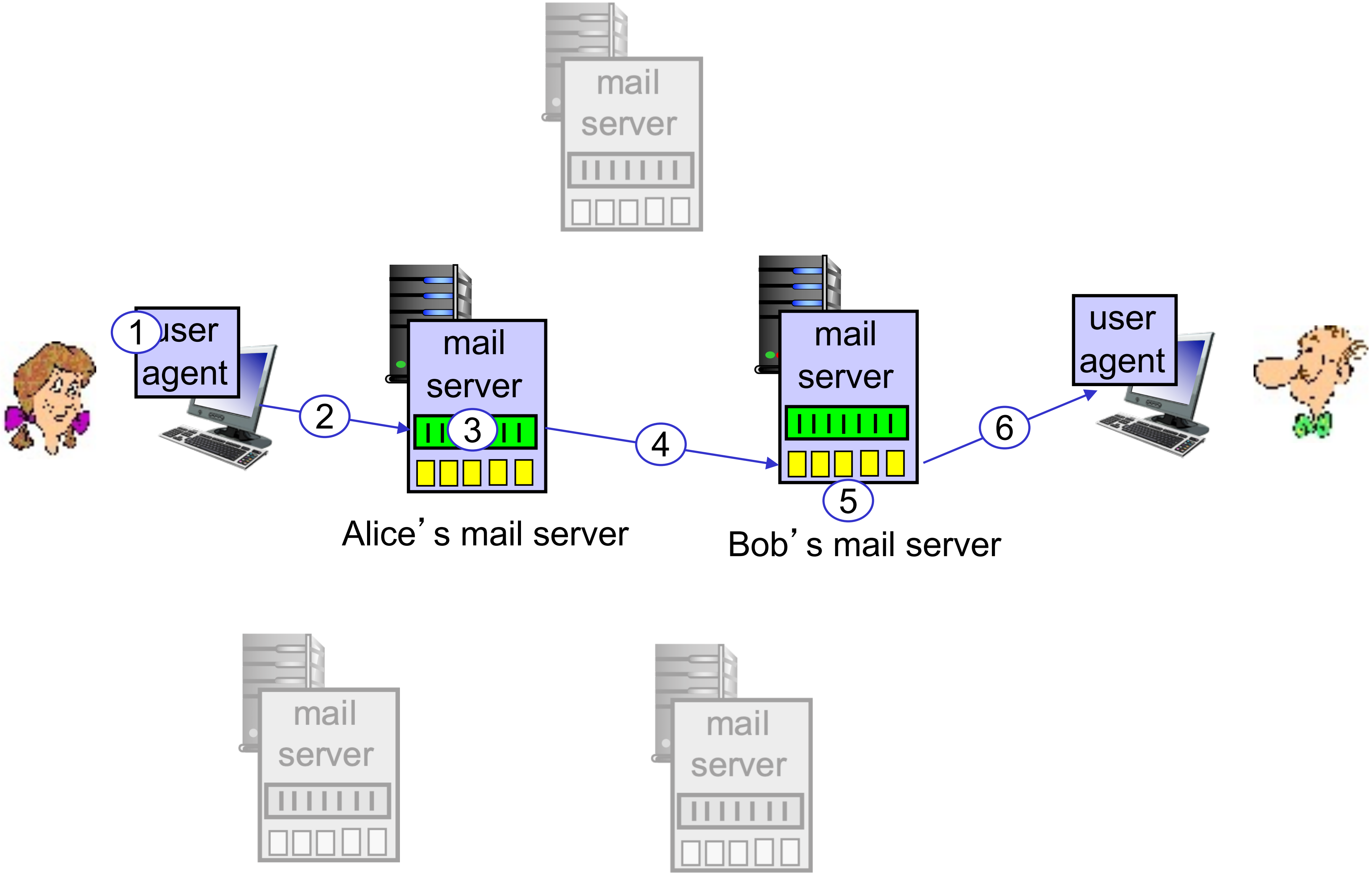
8 bit ASCII Cyrillic (CP1251)



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ђ 128	Ѓ 129	‚ 130	ѓ 131	„ 132	… 133	† 134	‡ 135	▮ 136	‰ 137	Љ 138	‹ 139	Њ 140	Ќ 141	Ћ 142	Џ 143
9	ђ 144	‘ 145	’ 146	“ 147	” 148	• 149	– 150	— 151	▮ 152	™ 153	љ 154	› 155	њ 156	ќ 157	ћ 158	џ 159
A	nbsp 160	Ў 161	ў 162	Ј 163	¤ 164	Ґ 165	¦ 166	§ 167	Ё 168	© 169	Є 170	« 171	¬ 172	shy 173	® 174	Ї 175
B	° 176	± 177	І 178	і 179	ґ 180	µ 181	¶ 182	· 183	ё 184	№ 185	є 186	» 187	ј 188	Ѕ 189	ѕ 190	ї 191
C	А 192	Б 193	В 194	Г 195	Д 196	Е 197	Ж 198	З 199	И 200	Й 201	К 202	Л 203	М 204	Н 205	О 206	П 207
D	Р 208	С 209	Т 210	У 211	Ф 212	Х 213	Ц 214	Ч 215	Ш 216	Щ 217	Ъ 218	Ы 219	Ь 220	Э 221	Ю 222	Я 223
E	а 224	б 225	в 226	г 227	д 228	е 229	ж 230	з 231	и 232	й 233	к 234	л 235	м 236	н 237	о 238	п 239
F	р 240	с 241	т 242	у 243	ф 244	х 245	ц 246	ч 247	ш 248	щ 249	ъ 250	ы 251	ь 252	э 253	ю 254	я 255

https://segfault.kiev.ua/cyrillic-encodings/

Scenario: Alice sends message to Bob



SMTP: comparison with HTTP

- ***SMTP***
  - Uses persistent connections
  - Requires message – header & body (7-bit ASCII with exceptions)
  - Server uses “CRLF.CRLF” to determine end of message

- ***Comparison with HTTP***
  - HTTP: pull (the client pulls the page from the server)
  - SMTP: push (the sender’s server pushes the message to the receiver’s server)
  - Both have ASCII command/response interaction, status codes
  - HTTP: each object encapsulated in its own response msg
  - SMTP: multiple objects sent in multipart msg

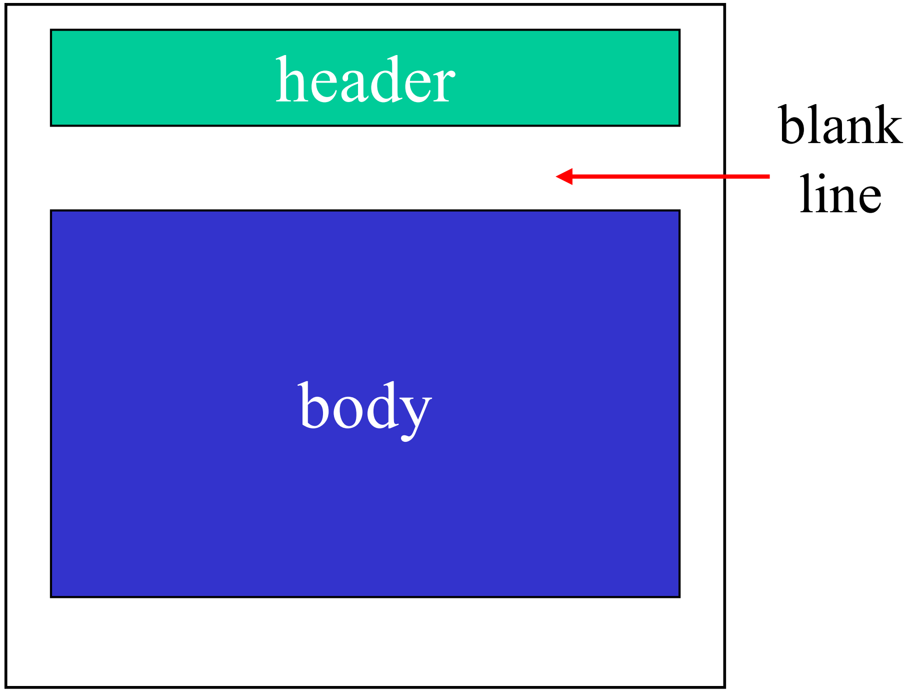
Mail Message Format

SMTP: protocol for exchanging email msgs

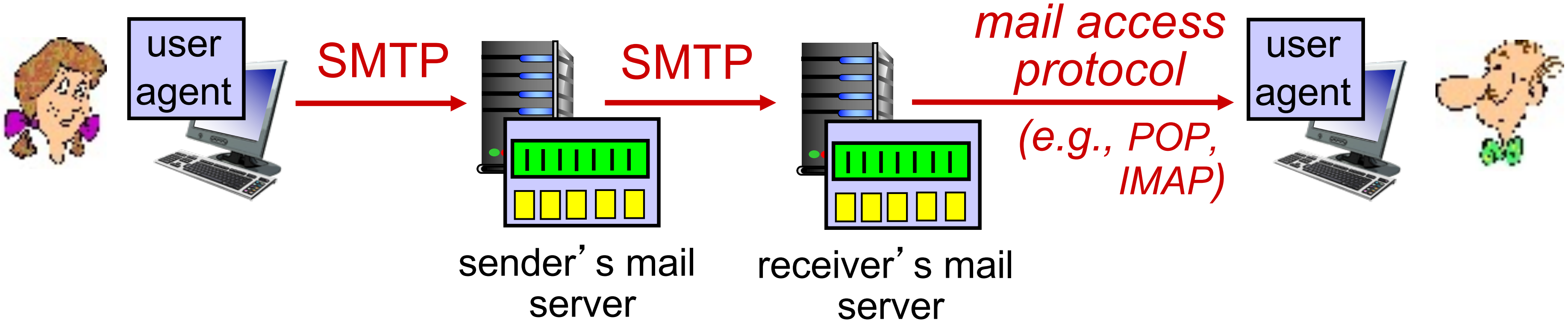
RFC 822: standard for text message format:

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

- 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 msgs on server
  - HTTP: it is not a protocol dedicated for email communications, but it can be used for accessing your mailbox. Also called web based email, this protocol can be used to compose or retrieve emails from your account. Hotmail is a good example of using HTTP as an email protocol.

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 (more) and IMAP4

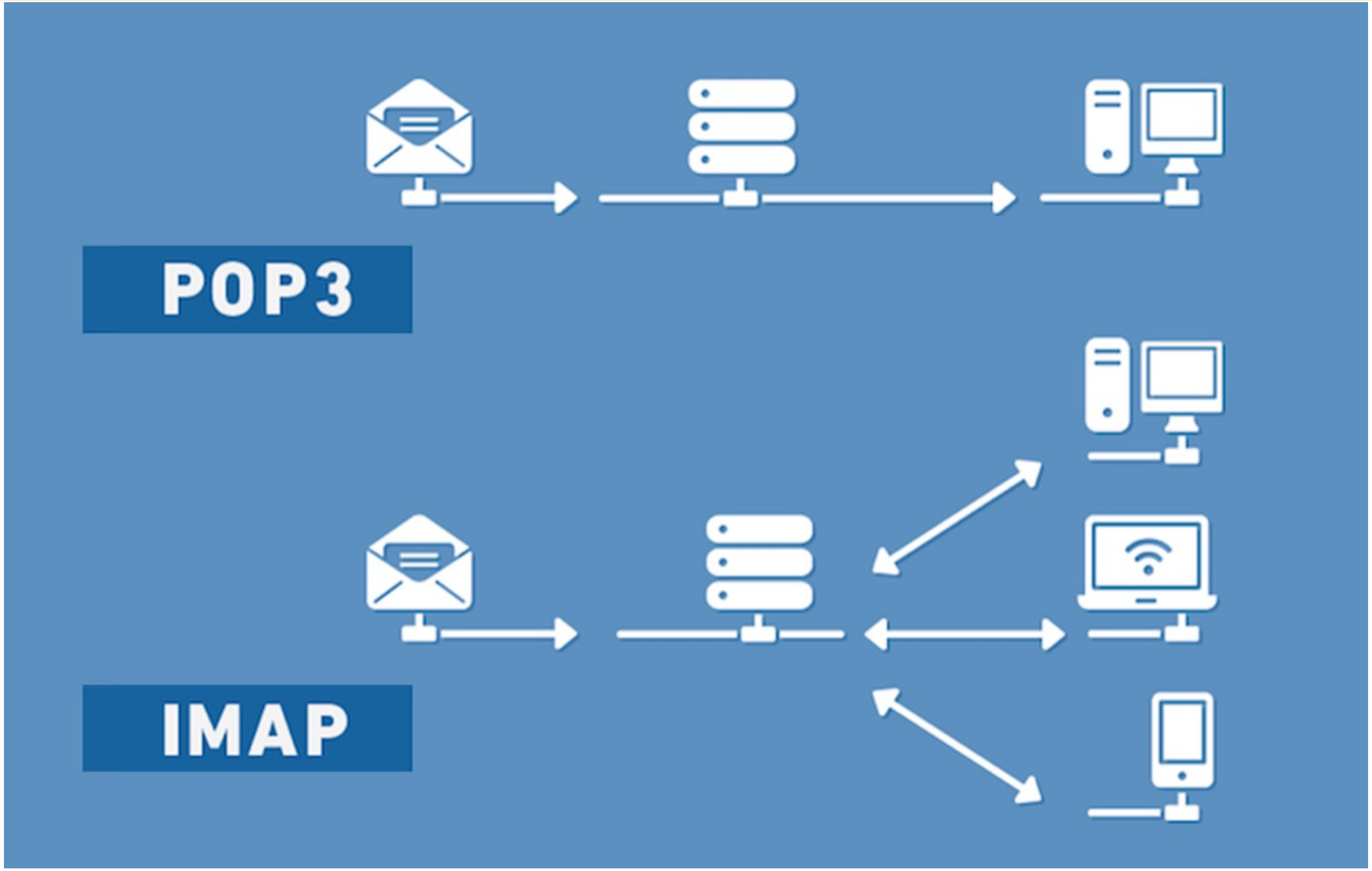
more about POP3

- 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

IMAP4

- 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

POP3 vs IMAP



IMAP does not delete the message from the main service

Base64 Encoding

- The term Base64 originates from a specific MIME (Multipurpose Internet Mail Extension) content transfer encoding.
- Each Base64 digit represents exactly 6 bits of data.
- Base 64 is a way to representing binary data – like images – into ASCII text.
- We can use Base-64 encoding to easily send binary data through
  - HTML Mail, e-mail attachments, JSON requests and HTML forms.

Base64 motivation

- When we transmit bits, we cannot be sure that the data would be interpreted in the same format as we intended it to be.
- So, we send over data coded in some format (like Base64) that both parties understand.
- That way even if sender and receiver interpret same things differently, but because they agree on the coded format, the data will not get interpreted wrongly

ASCII and Base64 are used for different purposes.

- When you encode text in ASCII, you start with a text string and convert it to a sequence of bytes.
- When you encode data in Base64, you start with a sequence of bytes and convert it to a text string.

Base 64 example

If I want to send

```
Hello
world!
```

- If I send it as ASCII (or UTF-8) it will look like this:

```
72 101 108 108 111 10 119 111 114 108 100 33
```

- The 6th byte `10` is corrupted in some systems so we can base 64 encode these bytes as a Base64 string:

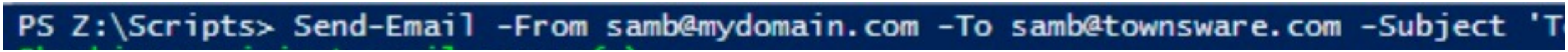
```
SGVsbG8Kd29ybGQh
```

which when encoded using ASCII looks like this:

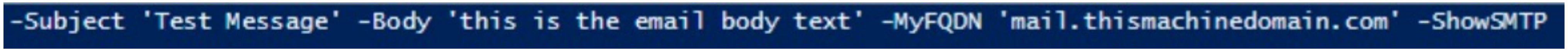
```
83 71 86 115 98 71 56 75 100 50 57 121 98 71 81 104
```

- All the bytes here are known safe bytes, so there is very little chance that any system will corrupt this message.

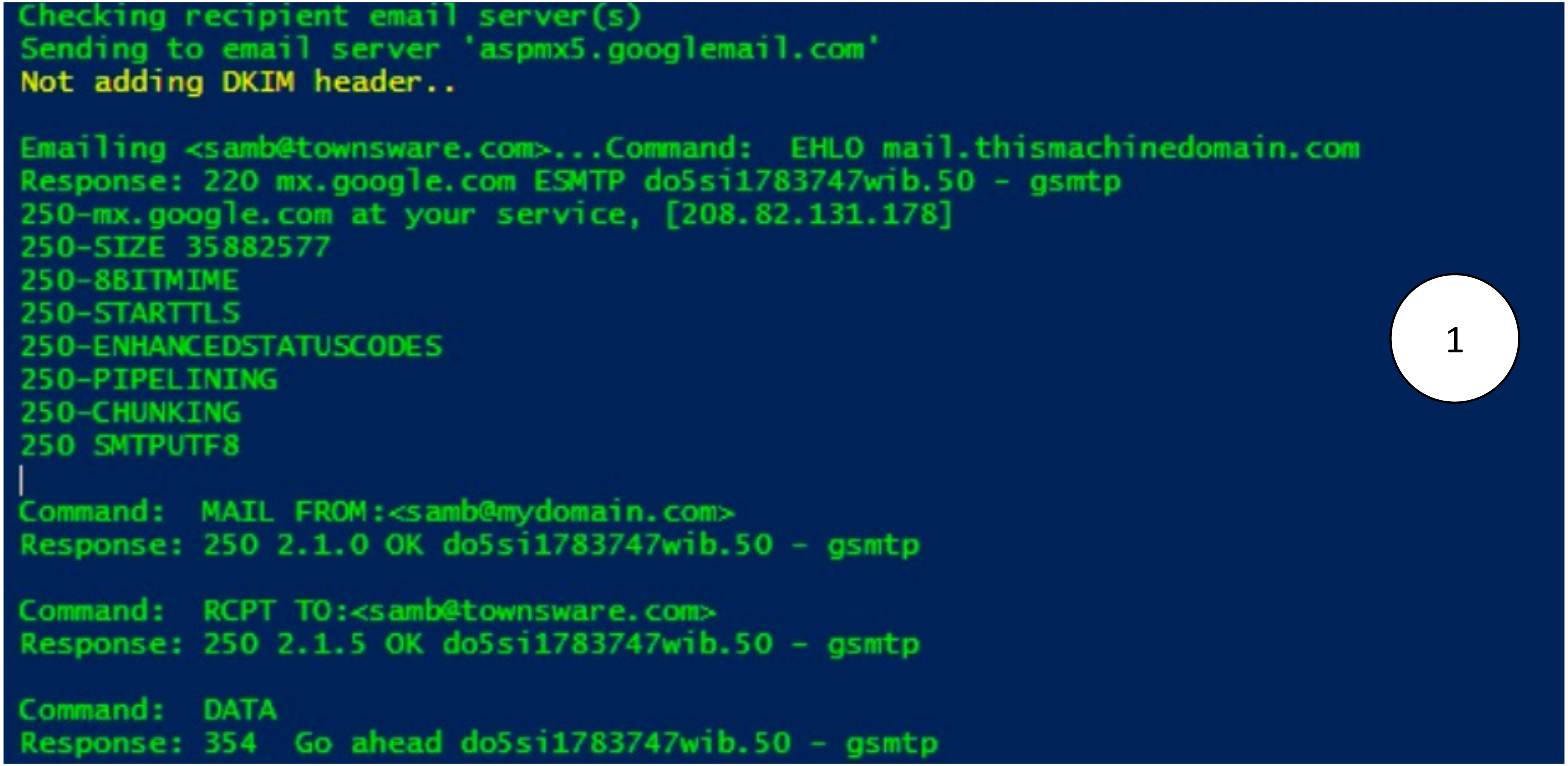
Send email messages from Powershell (1/2)



```
PS Z:\Scripts> Send-Email -From samb@mydomain.com -To samb@townsware.com -Subject 'T
```



```
-Subject 'Test Message' -Body 'this is the email body text' -MyFQDN 'mail.thismachinedomain.com' -ShowSMTP
```

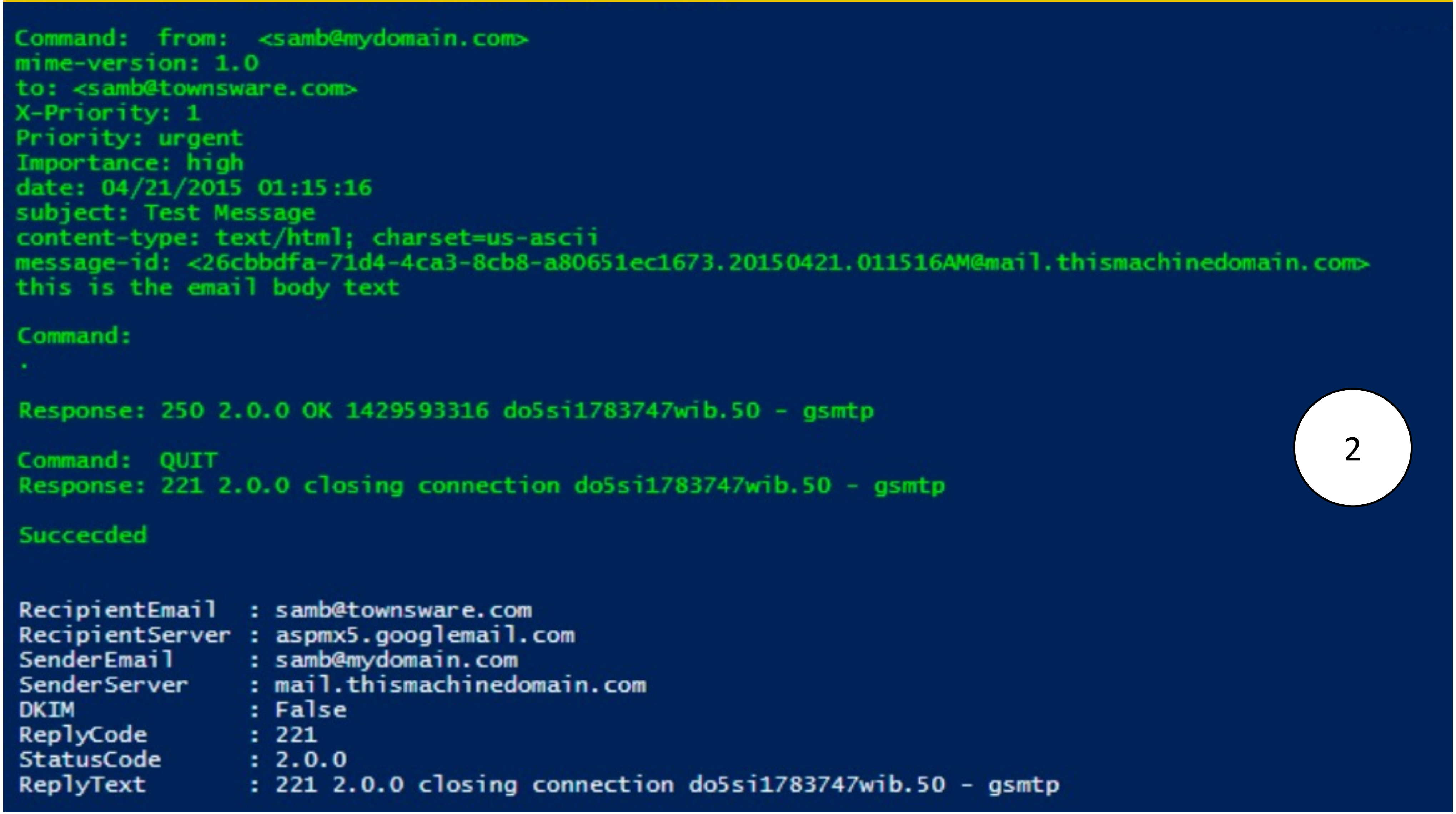


```
Checking recipient email server(s)
Sending to email server 'aspmx5.googlemail.com'
Not adding DKIM header..

Emailing <samb@townsware.com>...Command:  EHLO mail.thismachinedomain.com
Response: 220 mx.google.com ESMTP do5si1783747wib.50 - gsmtp
250-mx.google.com at your service, [208.82.131.178]
250-SIZE 35882577
250-8BITMIME
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8
|
Command:  MAIL FROM:<samb@mydomain.com>
Response: 250 2.1.0 OK do5si1783747wib.50 - gsmtp

Command:  RCPT TO:<samb@townsware.com>
Response: 250 2.1.5 OK do5si1783747wib.50 - gsmtp

Command:  DATA
Response: 354  Go ahead do5si1783747wib.50 - gsmtp
```

1

Source: https://superwidgets.wordpress.com/2015/04/21/powershell-module-to-send-email-without-need-for-smtp-relay-server/

Send email messages from Powershell (2/2)

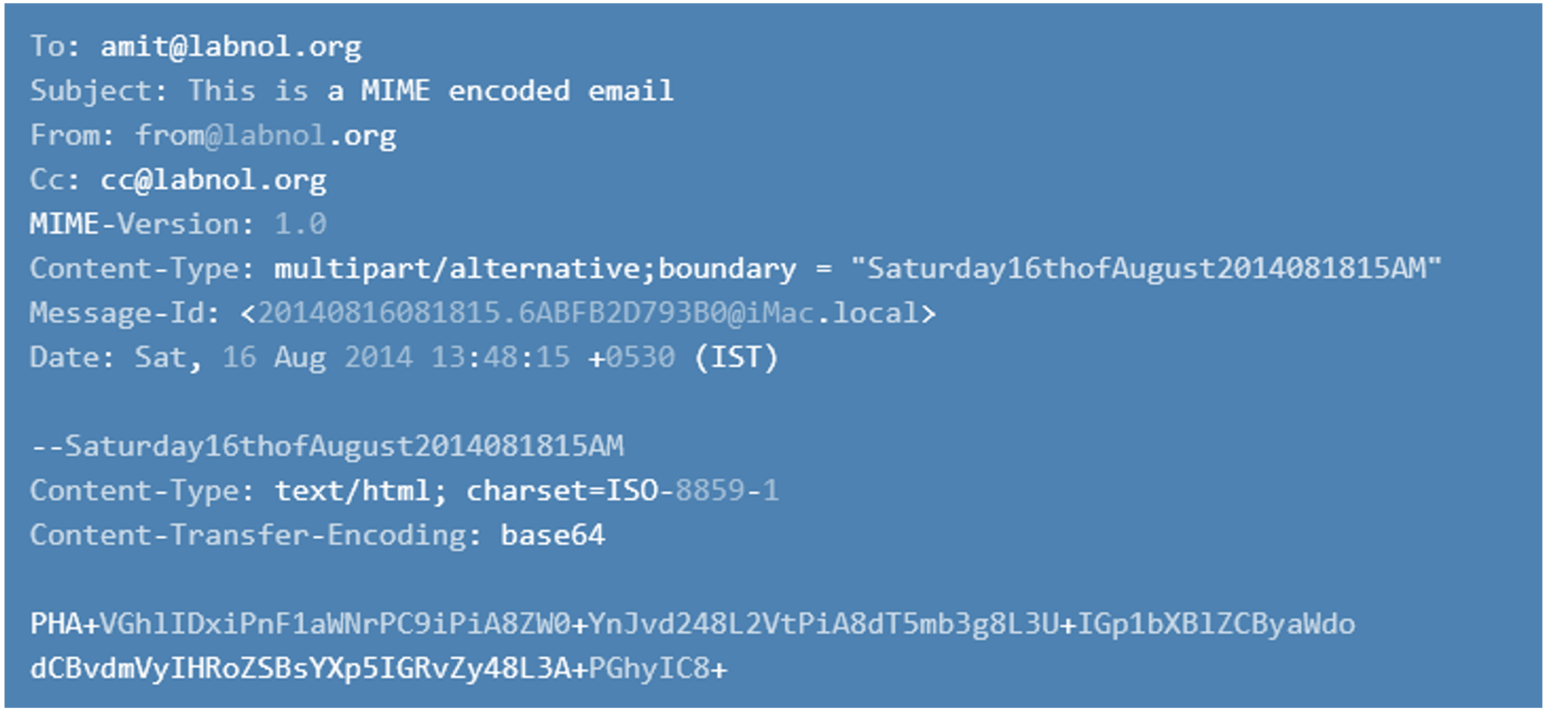


```
Command:  from:  <samb@mydomain.com>
mime-version: 1.0
to: <samb@townsware.com>
X-Priority: 1
Priority: urgent
Importance: high
date: 04/21/2015 01:15:16
subject: Test Message
content-type: text/html; charset=us-ascii
message-id: <26cbbdfa-71d4-4ca3-8cb8-a80651ec1673.20150421.011516AM@mail.thismachinedomain.com>
this is the email body text

Command:
.

Response: 250 2.0.0 OK 1429593316 do5si1783747wib.50 - gsmtp

Command:  QUIT
Response: 221 2.0.0 closing connection do5si1783747wib.50 - gsmtp

Succecded


RecipientEmail  : samb@townsware.com
RecipientServer : aspmx5.googlemail.com
SenderEmail     : samb@mydomain.com
SenderServer    : mail.thismachinedomain.com
DKIM            : False
ReplyCode       : 221
StatusCode      : 2.0.0
ReplyText       : 221 2.0.0 closing connection do5si1783747wib.50 - gsmtp
```

2

Source: https://superwidgets.wordpress.com/2015/04/21/powershell-module-to-send-email-without-need-for-smtp-relay-server/

Raw source of a MIME encoded HTML Mail



```
To: amit@labnol.org
Subject: This is a MIME encoded email
From: from@labnol.org
Cc: cc@labnol.org
MIME-Version: 1.0
Content-Type: multipart/alternative;boundary = "Saturday16thofAugust2014081815AM"
Message-Id: <20140816081815.6ABFB2D793B0@iMac.local>
Date: Sat, 16 Aug 2014 13:48:15 +0530 (IST)

--Saturday16thofAugust2014081815AM
Content-Type: text/html; charset=ISO-8859-1
Content-Transfer-Encoding: base64

PHA+VGhlIDxiPnF1aWNrPC9iPiA8ZW0+YnJvd248L2VtPiA8dT5mb3g8L3U+IGp1bXBlZCByaWdo
dCBvdmVyIHRoZSBsYXp5IGRvZy48L3A+PGhyIC8+
```

Source: https://ctrlq.org/code/19840-base64-encoded-email

DNS: Domain Name System

Top-Level Domain (TLD) servers

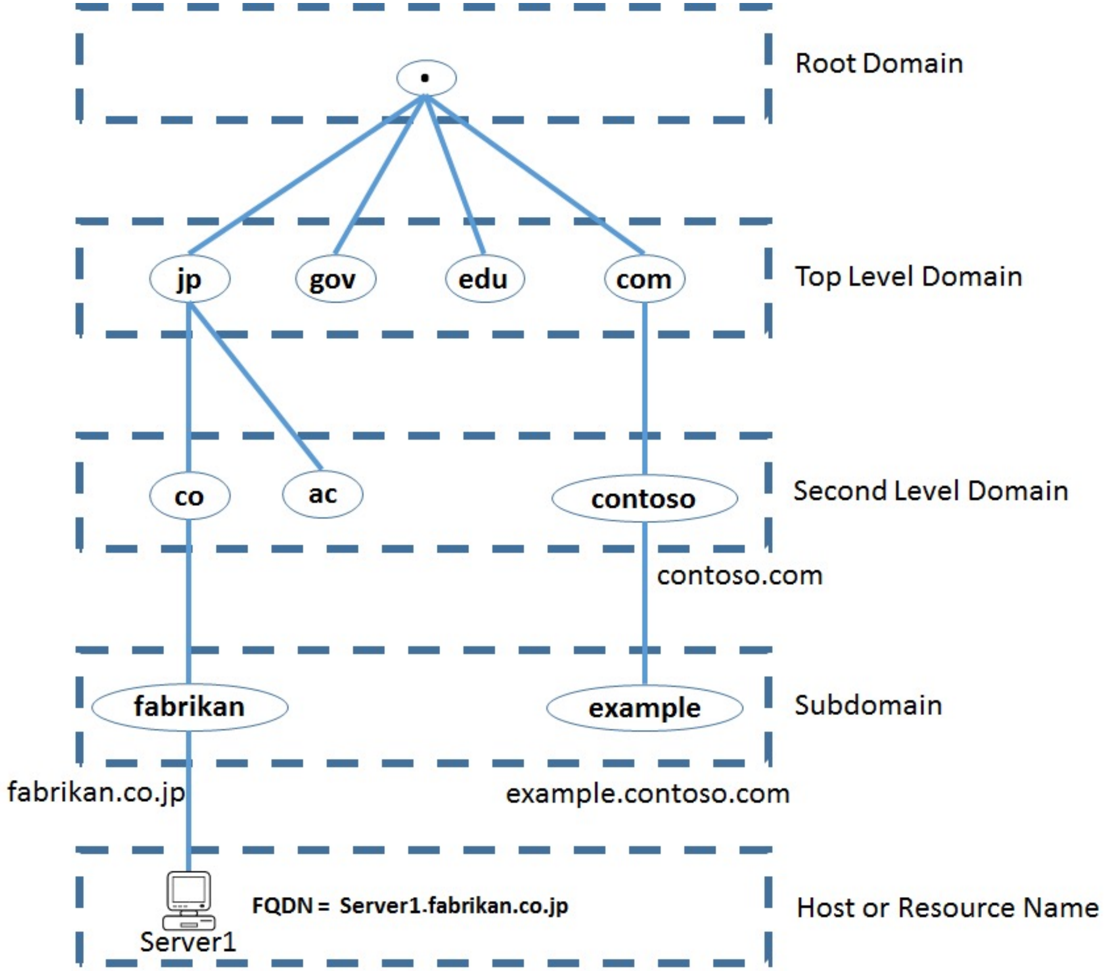
Responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp

www.microsoft.com

www.contoso.com

www.example.contoso.com

Note: All countries have their own domain names.



- **Different perspective:** Historical, Geographical, activity or even cultural

Source Image: https://gitlearning.wordpress.com/2015/06/23/dns-server/

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

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"

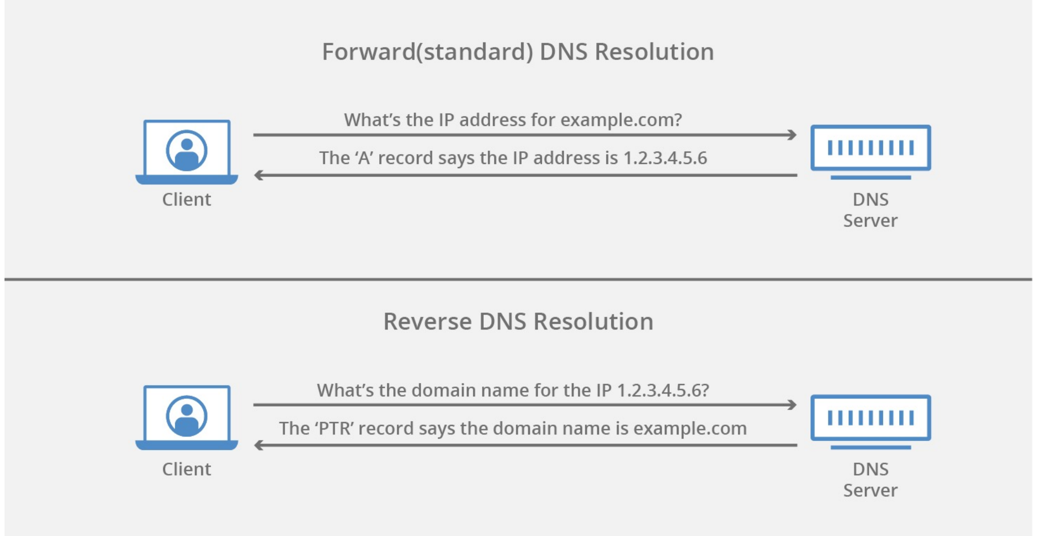
DNS: Services, Structure

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?

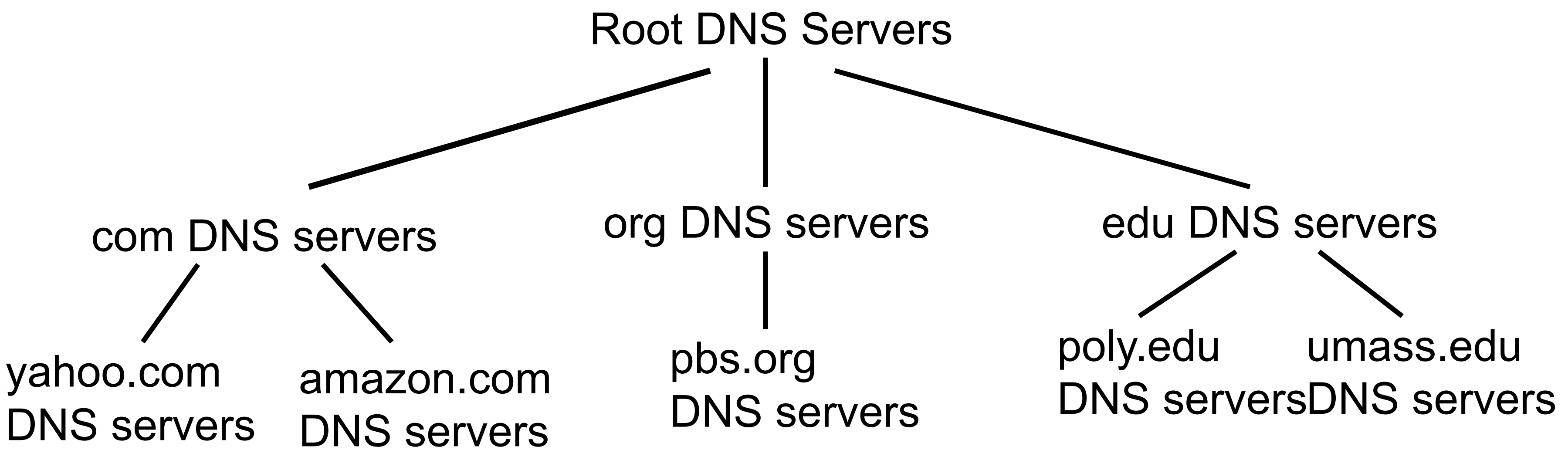
- Single point of failure
- Traffic volume
- Distant centralized database
- Maintenance
- Security



Forward(standard) DNS Resolution
What's the IP address for example.com?
The 'A' record says the IP address is 1.2.3.4.5.6
Client
DNS Server
Reverse DNS Resolution
What's the domain name for the IP 1.2.3.4.5.6?
The 'PTR' record says the domain name is example.com
Client
DNS Server

Image Source: https://www.cloudflare.com/learning/dns/glossary/reverse-dns/

DNS: A Distributed, Hierarchical Database



Client wants IP for www.amazon.com; 1st approx:

- 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

Authoritative DNS servers:

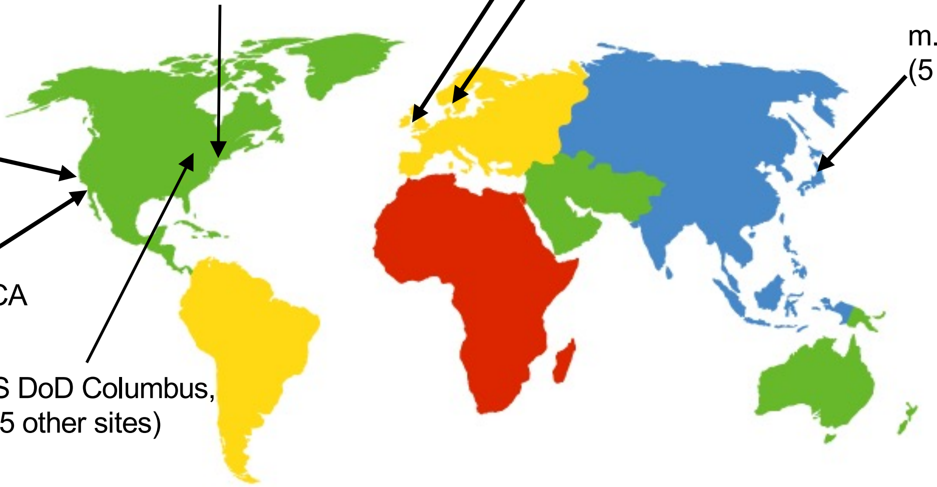
- Organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- Can be maintained by organization or service provider

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

ICANN

The Internet Corporation for Assigned Names and Numbers is a nonprofit organization responsible for coordinating the maintenance and procedures of several databases related to the namespaces and numerical spaces of the Internet,



c. Cogent, Herndon, VA (5 other sites)
d. U Maryland College Park, MD
h. ARL Aberdeen, MD
j. Verisign, Dulles VA (69 other sites)

k. RIPE London (17 other sites)

i. Netnod, Stockholm (37 other sites)

m. WIDE Tokyo
(5 other sites)

e. NASA Mt View, CA
f. Internet Software C.
Palo Alto, CA (and 48 other sites)

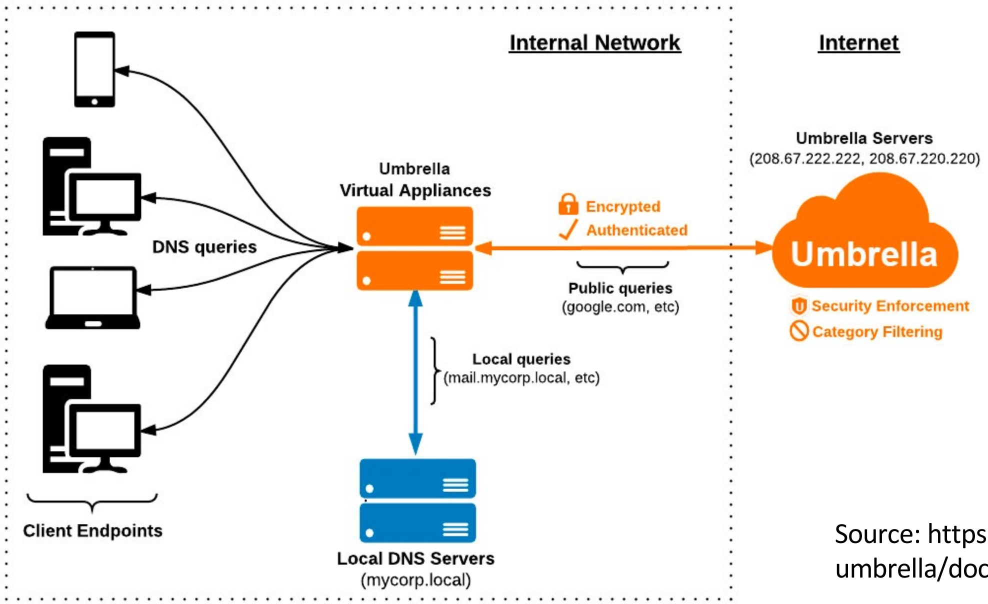
a. Verisign, Los Angeles CA
(5 other sites)
b. USC-ISI Marina del Rey, CA
l. ICANN Los Angeles, CA
(41 other sites)

g. US DoD Columbus, OH (5 other sites)

13 root name “servers” worldwide

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



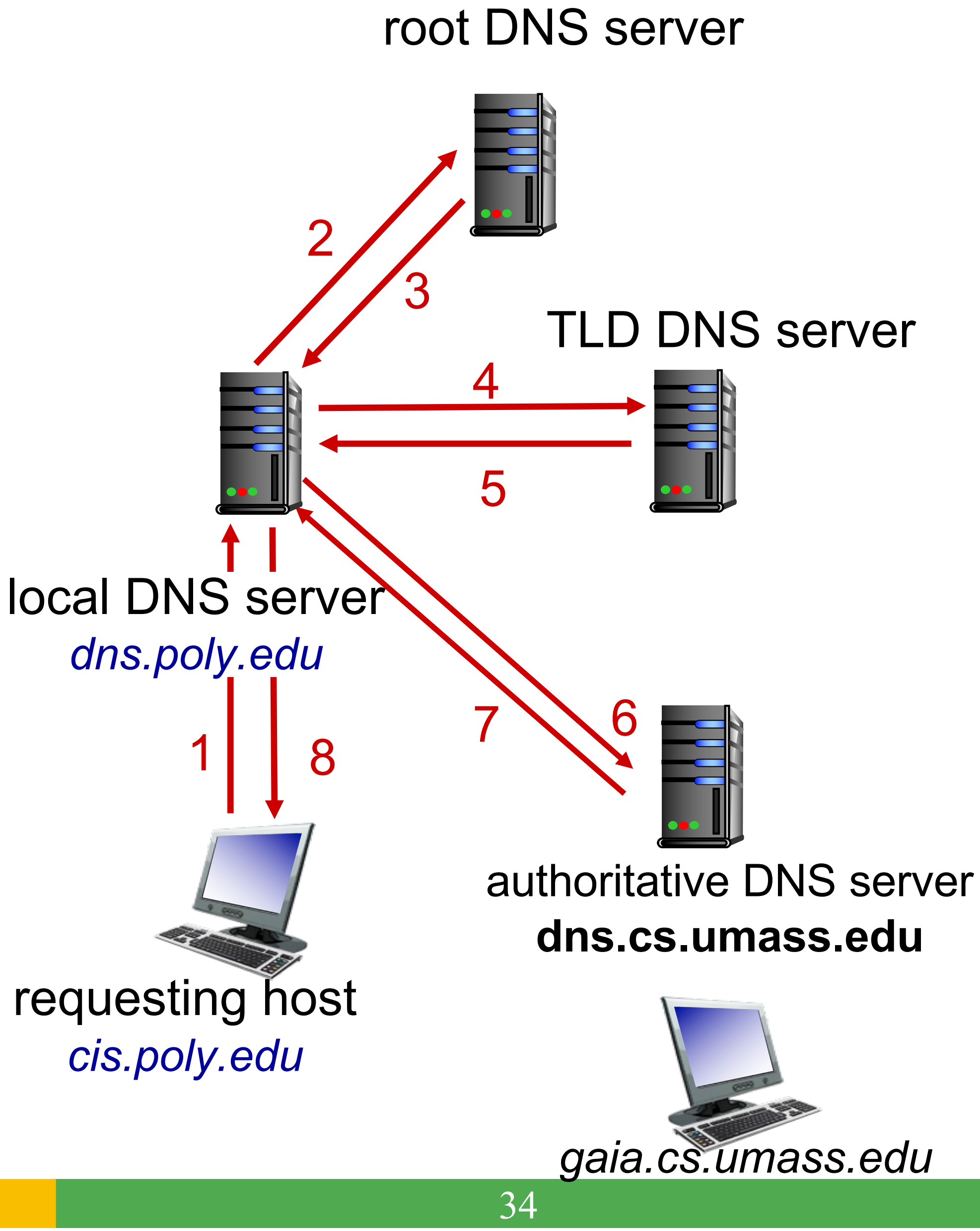
Source: https://docs.umbrella.com/deployment-umbrella/docs/6-local-dns-forwarding

DNS Name Resolution Example

- host at cis.poly.edu wants IP address for gaia.cs.umass.edu

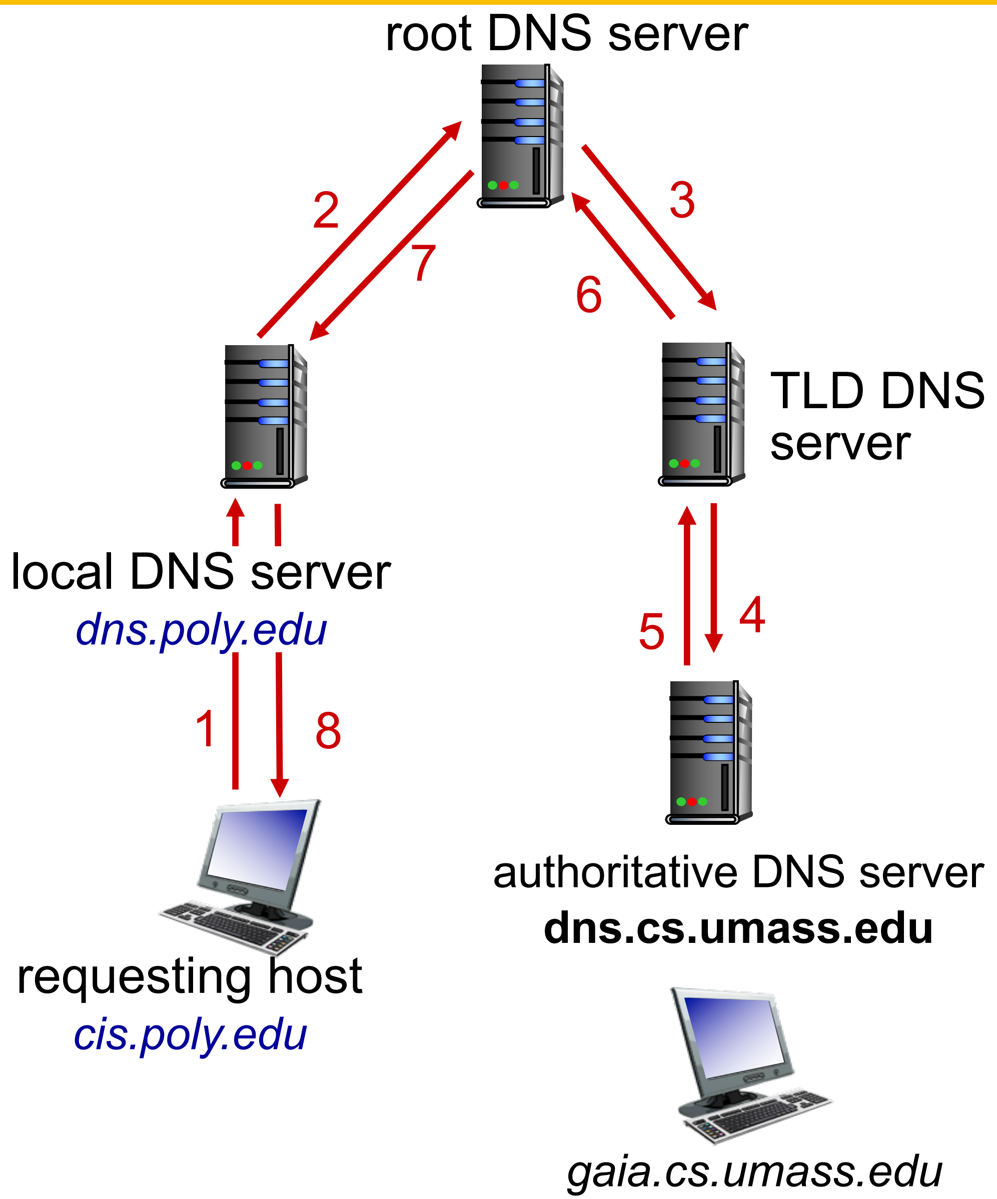
Iterated query:

- 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

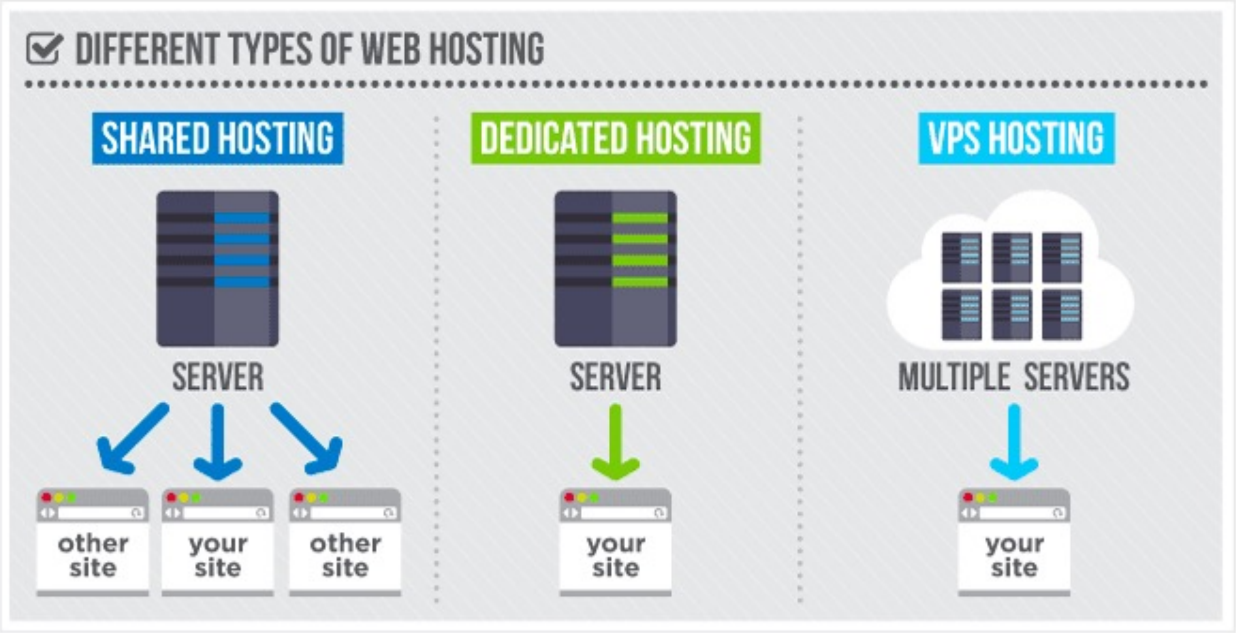
- Once (any) name server learns mapping, it *caches* mapping
  - cache entries timeout (disappear) after some time (*TTL)
  - TLD servers typically cached in local name servers
    - thus root name servers not often visited
- Cached entries may be *out-of-date* (best effort name-to-address translation!)
  - if name host changes IP address, may not be known Internet-wide until all TTLs expire
- Update/notify mechanisms proposed IETF standard
  - RFC 2136

*TTL: time to leave

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

Web Hosting



VPS = Virtual Private Servers

Domain names and web hosting are two different services. However, they work together to make websites possible. Basically a DNS is like a massive address book that is constantly updated. To build a website you will need both a domain name and web hosting account

Image Source: https://www.hostingadvice.com/the-basics/#types-hosting

Inserting Records into DNS

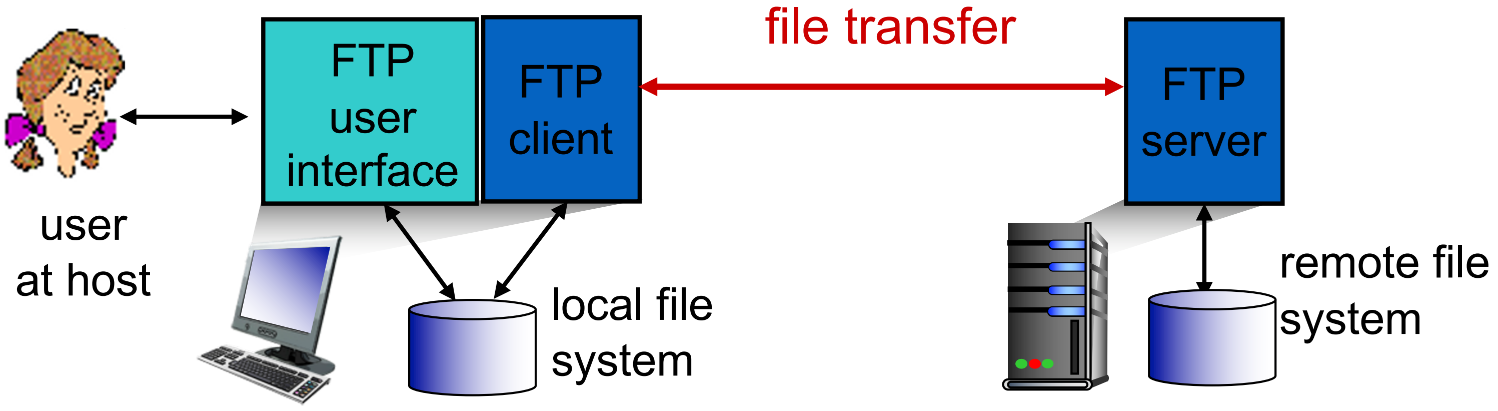
- Example: new startup “Network Utopia”

- 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:

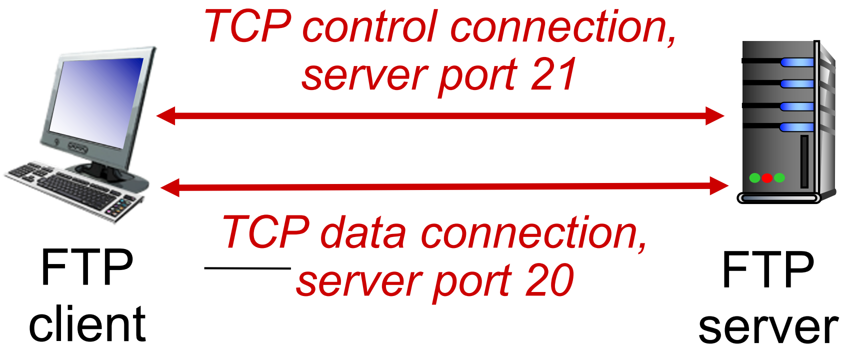
```
(networkutopia.com, dns1.networkutopia.com, NS)
(dns1.networkutopia.com, 212.212.212.1, A)
```

- Create authoritative server type A record for `www.networkuptopia.com`; type MX record for `networkutopia.com`

File Transfer Protocol (FTP)

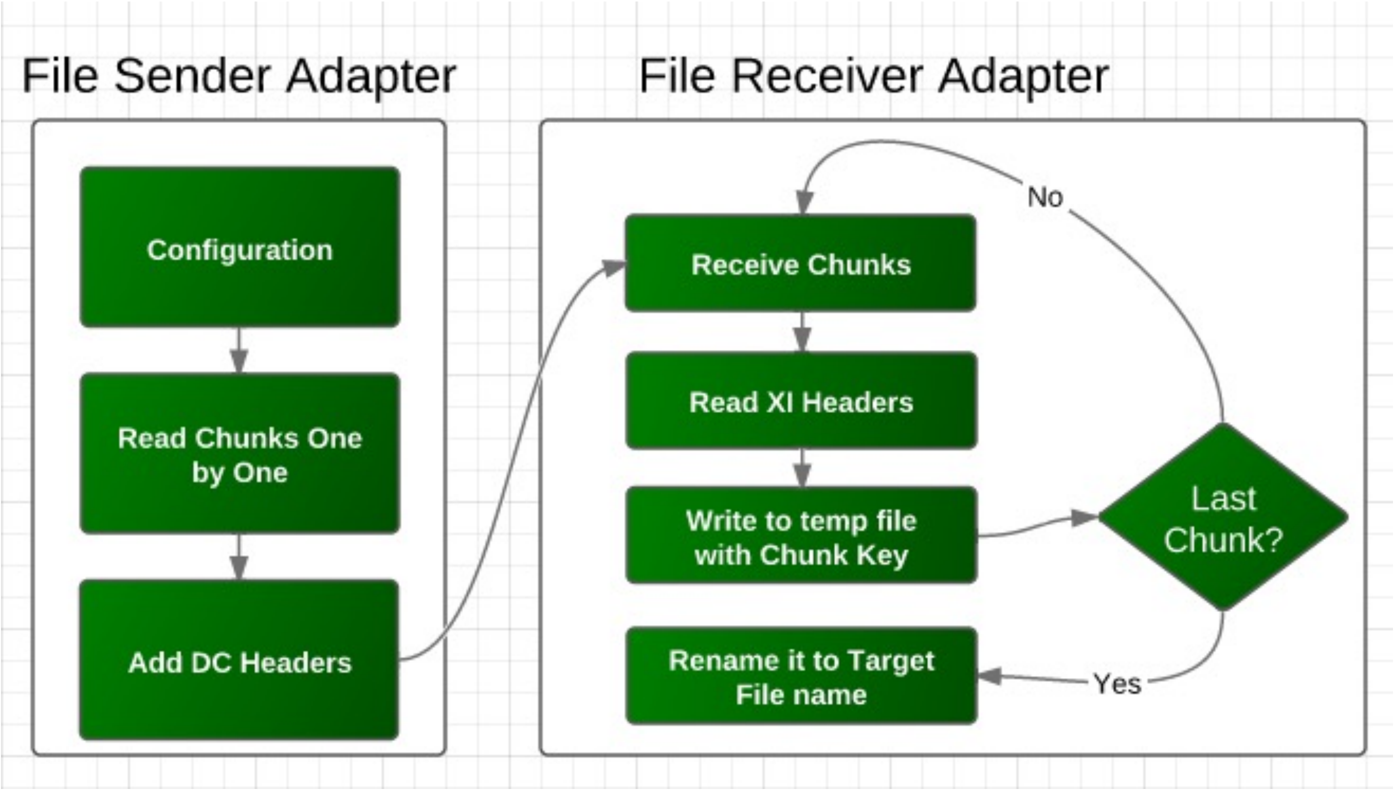


FTP: separate control, data connections



- FTP client contacts FTP server at port 21, using TCP
- Client authorized over control connection
- Client browses remote directory, sends commands over control connection
- When server receives file transfer command, server opens 2nd TCP data connection (for file) to client
- After transferring one file, server closes data connection
- FTP server maintains “state”: current directory, earlier authentication

FTP Transfer



Source: https://blogs.sap.com/2011/12/26/fileftp-adapter-large-file-transfer-chunk-mode/

Summary

- Electronic Mail
- SMTP
- POP3
- IMAP
- DNS
- FTP