Computer Networks

Lecture on

Selected application layer protocols

Plan of This Lecture

- Trivial File Transfer Protocol (TFTP)
- File Transfer Protocol (FTP)
- HyperText Transfer Protocol (HTTP)
- Electronic mail protocols

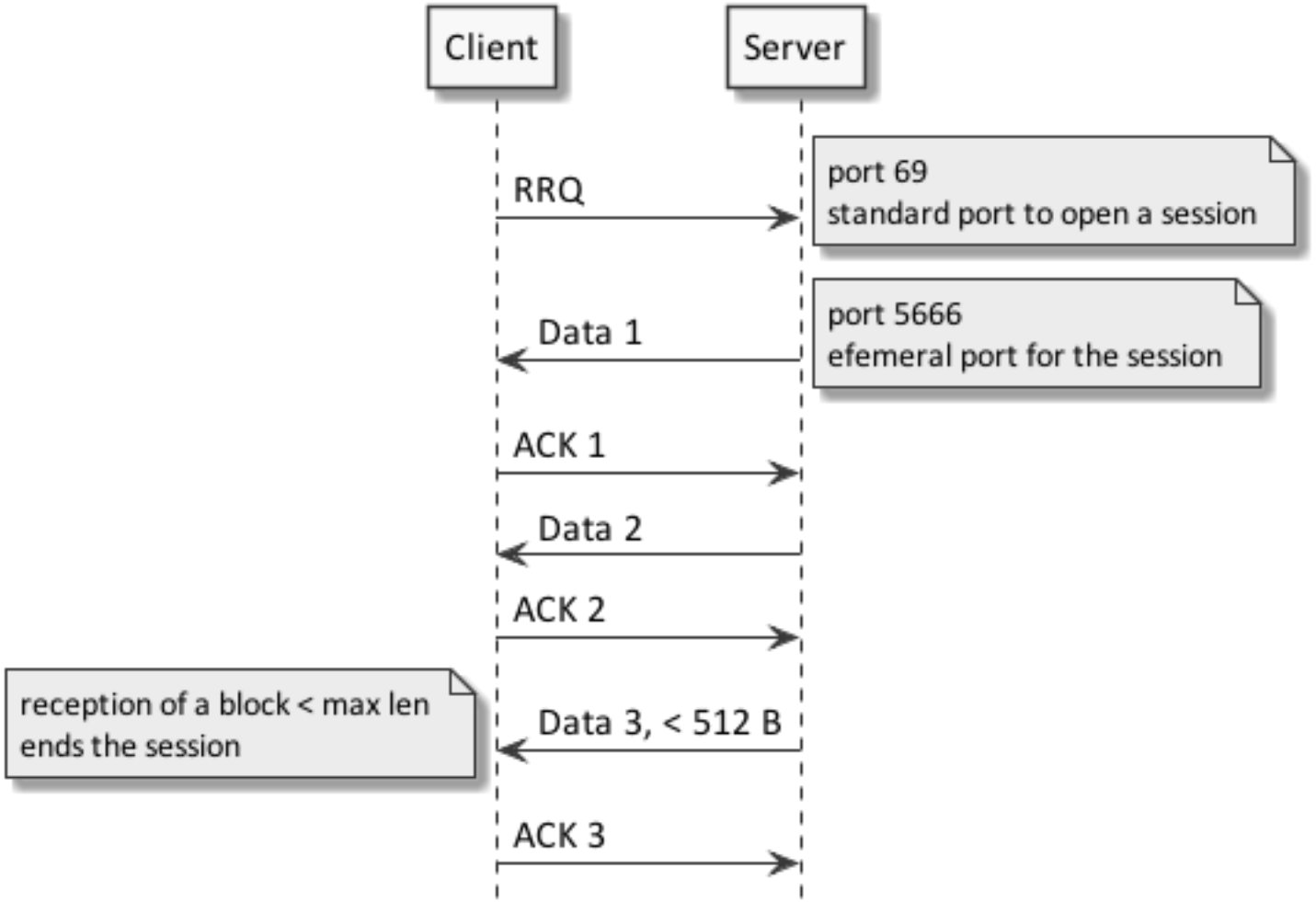
TFTP

Used by disk-less hosts when booting from a LAN server

- Works over UDP
- No authentication
  - security services could be supplied above or below TFTP
- Small memory footprint
- Several extension has been proposed 1981 - 2015

- Protocol Data Units
  - Read ReQuest – RRQ contains the filename and a text/binary indication
  - Write ReQuest – WRQ
  - Data – contains a 16-bit block number and up to 512 bytes of data
  - ACK – contains a 16-bit block number
  - Error

TFTP Session Example



A timer can start the retransmission of the last Data or ACK

FTP *File Transfer Protocol*

	First implementations	early 70's
RFC 959	Basic protocol for file transfer	1985
RFC 1579	Firewall-Friendly FTP	1994
RFC 2228	FTP Security Extensions	1997

- Client server
- TCP port 21 – control port 20 – data
- Plain text authorization or anonymous access
- PASV mode – when server cannot open a connection to the client
  not every FTP server handles it

Use SFTP (SSH File Transfer Protocol) – for secure data access!

Web browsers allow for:

ftp://<ftpserveraddress>

ftp://<login>:<password>@<ftpserveraddress>

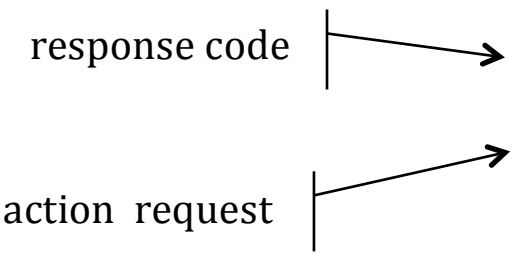
Do it yourself: see man sftp and man scp

What can a user order?

open, user	open [port]
dir, ls	dir [rdir] [file-name]
cd, lcd, pwd	cd/lcd – remote/ local
binary, ascii, cr	cr – transl. CR/LF
put, get	put local-file [remote-file]
mput, mget, mdelete, mls	wild characters can be used
prompt	
nmap	name translation, e.g.: nmap $1.$2 $1_$2
ntrans	character translation
bye	close session

Try to use it: sftp *your_login@your_server*

Session Example



1xx = OK I will
2xx = OK done
3xx = OK so far
4xx = NO temp
5xx = ERROR

```
client % ftp ftp.ii.pw.edu..pl
Connected to ftp ftp.ii.pw.edu..pl
220 Welcome to II PW FTP Server
530 Please login with USER and PASS.
Name (ftp.ii.pw.edu..pl): anonymous
331 Please specify the password.
Password: XXXXXXX
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub
250 Directory successfully changed.
ftp> ls
200 PORT command successful.
150 Here comes the directory listing.
drwxrwxrwx   2 0   0   2048   Aug 23 12:56   mirrors
226 Directory send OK.
ftp> quit
221 Goodbye.
client %
```

What Are the PDUs?

Exemplary requests

USER *username*

PASS *password*

LIST give file list

RETR *filename* get the file

STOR *filename* take the file

Exemplary responses

331 username OK, password required

125 data connection already open;
transfer starting

425 Can't open data connection

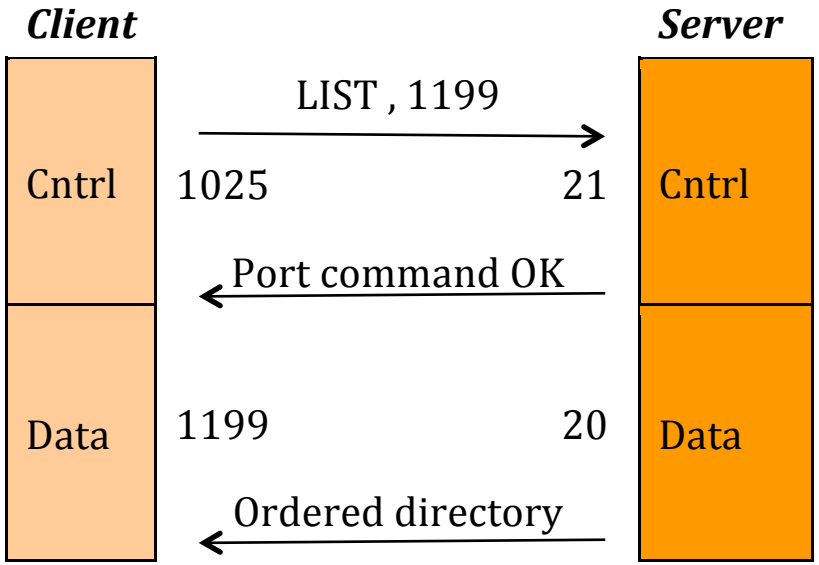
452 Error writing file

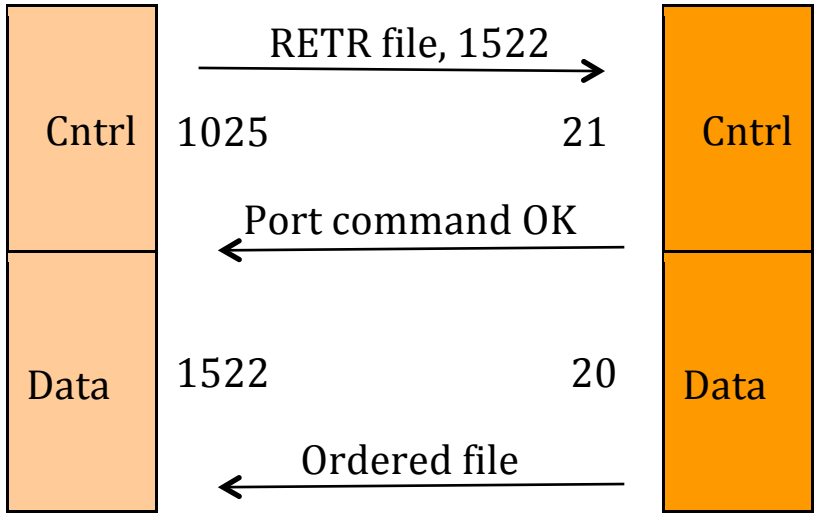
Problems with FTP

Occur with firewalls, NAT and load-balancing devices

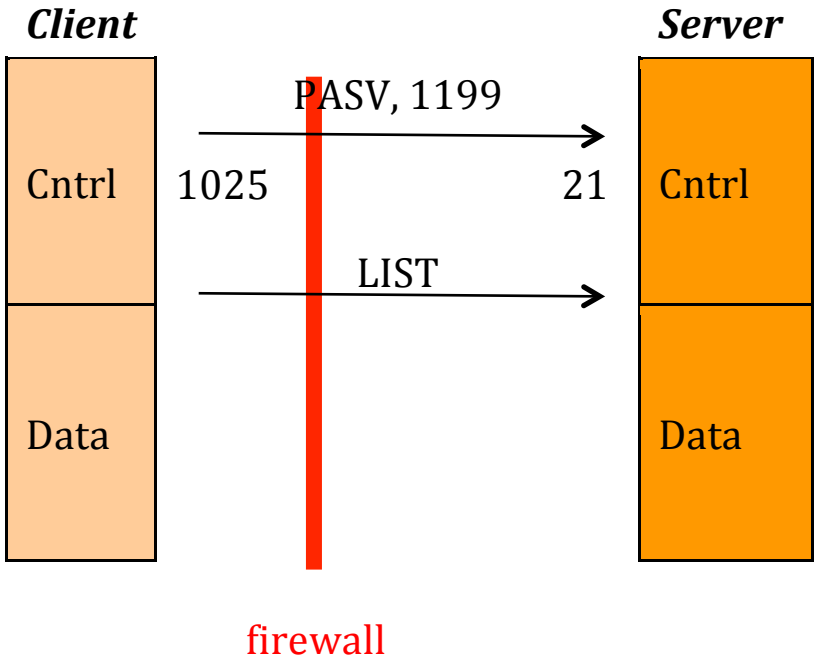
1. Additional TCP/IP connections are used for data transfers
2. Data connections may be sent to random port numbers
3. Data connections may originate from the server to the client,
   as well as originating from the client to the server
4. Data connections' destination addresses are negotiated on the fly
   between the client and server over the channel used for the control connection
5. The control connection is idle while the data transfer takes place on the data connection

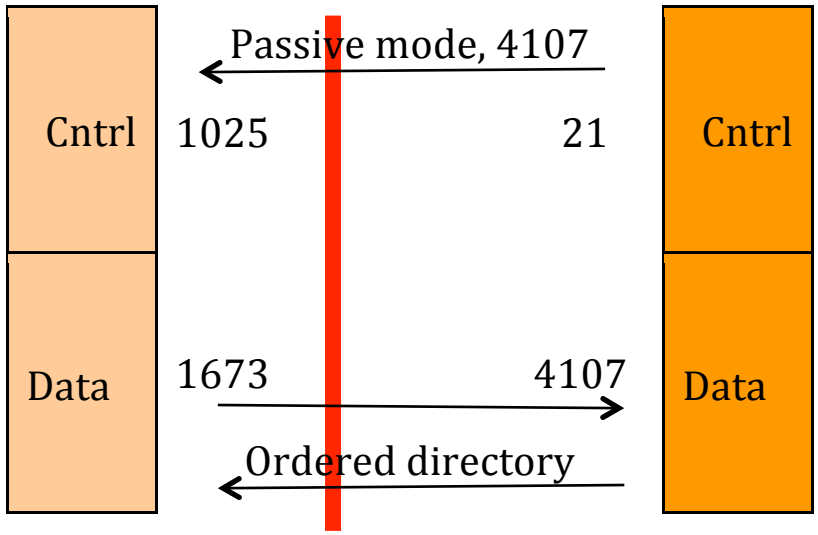
FTP Session – Normal Mode





FTP Session – Passive Mode





HTTP *HyperText Transfer Protocol*

1992 first draft

1996 RFC1945 HTTP 1.0

1999 RFC2616 HTTP 1.1

- persistent TCP connections enabled by default
- works well with proxies
- supports request pipelining

2015 RFC7540 HTTP 2.0

HTTP – stateless protocol

Do not keep any session state unlike to FTP and SMTP

- easy to implement
- more reliable
- other transport protocols (than TCP) can be used – e.g. UDP, QUIC

World Wide Web = network of documents accessible by HTTP

WWW document = hypertext with hyperlinks – HTML format

HTTP Session Schema

U: writes a URL, e.g., http://www.yahoo.com

C: DNS lookup, IP address designation

C: send to server "GET /HTTP/1.0\cr\lf\cr\lf"

S: decide how to serve a request

S: send a reply header and optional data

C, S: close the TCP session

HTTP Requests

GET	get data	
HEAD	get only header response	
POST	submit data to the identified resource	
PUT	uploads a representation of the specified resource	
DELETE	delete the specified resource	
TRACE	give back the request	
OPTIONS	get available options	new in HTTP 1.1
CONNECT	switch proxy mode (SSL tunnel)	
LINK		only in HTTP 1.0
UNLINK		
PATCH	partial modifications to a resource	

Full-Request ::=

Request-Line *(	GET /cgi-bin/q HTTP/1.1
General-Header |	Connection: Keep-Alive
Request-Header |	User-agent: Mozzilla/5.0
	Host: www.blahblah.com.pl
	Accept: text/html
	Accept-language: en
	Accept-charset …
	Accept-encoding …
	Referrer … – *previous URL*
Entity-Header)	ContentType: application/x-www-form-urlencoded
CRLF	
[Entity-Body] – *a MIME document*	

Full-Response ::=

Status-Line *(	HTTP/1.1 200 OK
General-Header |	Keep-Alive: timeout=15, max=100
	Connection: Keep-Alive
Response-Header |	Server Apache/1.3.0 Unix
	Date … Transfer-Encoding …
	Pragma … Cache-Control …
Entity-Header)	Last-Modified: Mon, 22 Jun 2001 …
	Content-Length: 6166
	Content-Type: text/html
	Content-Encoding … Content-Language …
	Content-MD5 … Expires …

CRLF

[Entity-Body] – *a MIME document*

HTTP Cookies

It is a simple mechanism for session continuation

Server generates and sends a cookie

Client just stores it

Client resend it to the server only if:

- server name matches
- access path matches
- cookie date is valid

S: Set-cookie: Customer="WILE_E_COYOTE";
domain="host.xyz.com.pl"; path="/"; Max-Age="86400"

C: Cookie: Customer="WILE_E_COYOTE";

HTTP Authentication

Subject of authentication: *a realm* e.g., a subtree of file system

Server requests by: 401 (Unauthorized) and

challenge: auth-scheme “realm=name” *(“,” auth-param)

Client responds

E.g.:

S: WWW-Authenticate: Basic realm=”WallyWorld”

C: Authorization: Basic QwdgfsHmnnfHT;onjfaQ==

- Authorization needed for every realm access
- Web browser resends authorization when it was once taped by a user

Proxy / HTTP Cache

1997 RFC 2186 ICP Internet Cache Protocol

2000 RFC 2756 HTCP Hyper Text Caching Protocol

Advantages:

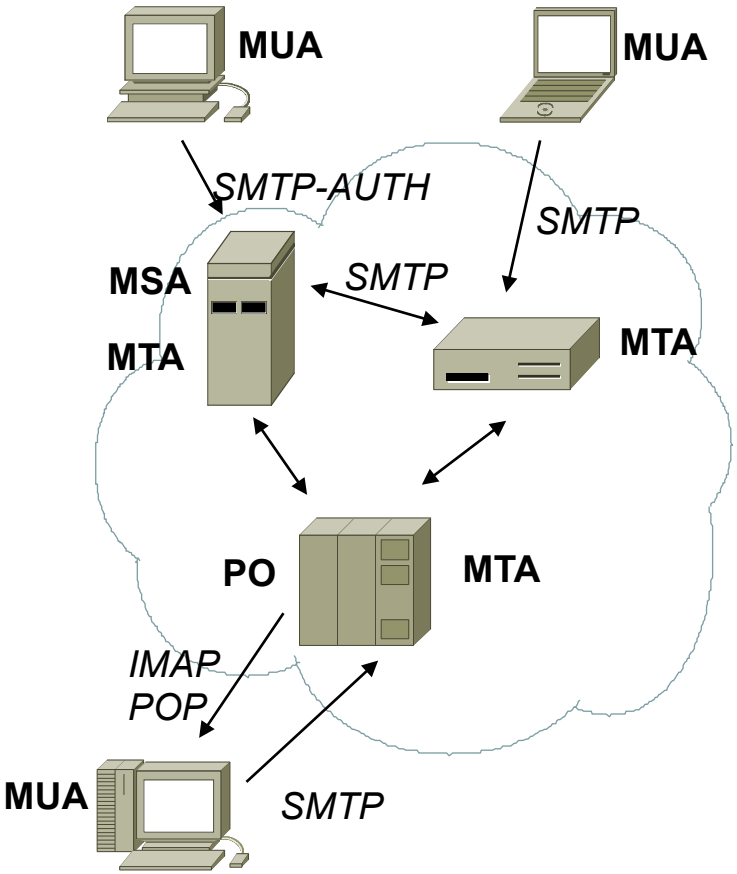
- efficiency
  - faster response
  - lower bandwidth consumption
  - lower server load
- security
  - proxy can filter data
  - captive portal
- others
  - different conversions e.g., http → https http1.0 → 1.1
  - advertisement injection

Do not cache:
- dynamic pages
- ciphered data
- cookies
- out of date data

Cache levels:
- Web browser
- local server
- ISP server

Cache-control:
- No-Cache
- No-Store
- Max-Age
- Max-Stale
- Only-If-Cached

Electronic Mail Protocols



MUA – *Mail User Agent* (client)
Mozilla, Outlook, Eudora, ...

MSA – *Mail Submission Agent*
accepts message submissions port 587

MTA – *Mail Transfer Agent* (server)
sendmail, postfix, Exchange, ... port 25

PO – *Post Office* – keeps mailoboxes
can be distinct from MTA

SMTP – *Simple Mail Transfer Protocol*
over TLS for security – recomended

SSMTP – Secure SMTP – obsoleted port 465

POP3, IMAP4, OWA, ... – "maildrop"

Outlook Web Access is a webmail service of Microsoft Exchange Server

PGP / GnuPG – for signing and ciphering

SMTP Commands

HELO	Identify the SMTP sender to the SMTP receiver – *obsolete*
EHLO	Identify the SMTP sender to the SMTP receiver under Extended SMTP
MAIL	Set the envelope return path (sender) and clear the list of envelope recipient addresses
RCPT	Add one address to the list of envelope recipient addresses.
DATA	Consider the lines following the command to be e-mail from the sender
RSET	Reset the envelope
NOOP	Ask the receiver to send a valid reply (but specify no other action)
QUIT	Ask the receiver to send a valid reply, and then close the transmission channel
HELP	Ask the receiver to send helpful information to the sender – *optional*
VRFY	Ask the receiver to confirm that a user has been identified
EXPN	Ask the receiver to confirm that a mailing list has been identified

SMPT Session Example

S: **220** This is XYZ smtp server at ...
...
C: **MAIL FROM**: <smith@pr.an_enterprice.com>
S: **250** smith@pr.an_enterprice.com ... Sender ok
C: **RCPT TO**: <brown@school.edu>
S: **250** brown@school.edu ... Recipient ok
C: **DATA**

envelope

S: **354** Enter mail, end with "." ...
C: **To**: brown@school.edu
C: **Subject**: Support proposal

header

C:
C: Dear customer
C: Please accept our congratulations ...
C: ...
C: .

body

S: **250** Message accepted for delivery
C: **QUIT**

Printable ASCII character set (64 values) is used for email representation

MIME *Multipurpose Internet Mail Extensions*

Extends the format of e-mail to support:

- Text in character sets other than ASCII
  - text/plain; charset=us-ascii
  - text/html
- Non-text attachments
  - image/jpeg, image/gifApplication
  - video/mpeg, video/quicktime
  - application/msword, application/octet-stream
- Message bodies with multiple parts
  - multipart/mixed; boundary=bndr-string
  - multipart/alternative, digest, parallel, partial
- Header information in non-ASCII character sets
  - e.g.: From: =?ISO-8859-1?Q?Olle_J=E4rnefors?=

Nowadays MIME is also used by HTTP and others

Do it yourself: Analyse raw format of a few emails.

Security Proposals

- **SMTP-AUTH** – include an authentication step through which the client effectively logs in to the mail server
- **Internet Mail 2000** – a new Internet mail architecture
- **Sender Policy Framework** (SPF) – makes it easier to counter most forged "From" addresses in email, and thus helps to counter e-mail spam
- **Sender ID** – an anti-spam proposal
- **Certified Server Validation** (CSV) – an authentication method, intended to fight spam
- **DomainKeys Identified Mail** – receiver checks that an email was authorized by the owner of the claimed origin domain
- **Domain-based Message Authentication, Reporting and Conformance** (DMARC) – expands (SPF + DomainKeys)
- …

Summary

- Domain Name System
  - names
  - principles
  - main record types
  - implementations
  - cashes
  - particular solutions: DDNS, mDNS, DNS-SD
- Trivial File Transfer Protocol
  - TFTP session example
- File Transfer Protocol
  - Functions of an FTP client application
  - FTP messages
  - Problems with FTP
- HyperText Transfer Protocol
  - Session schema
  - HTTP messages
  - Cookies
  - Authentication
  - Proxies
- Electronic mail protocols
  - SMTP, POP, IMAP
  - SMTP session schema
  - Multipurpose Internet Mail Extensions
  - Security proposals

Questions

1. What is the main usage of TFTP?
2. What is the main disadvantage of FTP?
3. How many connections are open during an FTP session?
4. Is it possible to transfer files using FTP via a gateway, which hides a client machine?
5. How does HTTP keep session data?
6. What is new in HTTP 1.1 with respect to HTTP 1.0?
7. What is the structure of HTTP request and HTTP response?
8. Mention 5 principal HTTP requests.
9. Does HTTP support authentication?
10. What are the advantages of HTTP proxy deployments?
11. Mention the locations where an HTTP cache can work.
12. How and what for the *telnet* command is used to connect with an SMTP server?
13. Are HTTP and SMTP stateless protocols?
14. What are the functions of SMTP (mention 3)?
15. Describe a structure of SMTP data exchange.
16. What are the main functional difference between POP and IMAP?
17. When you link a *Mail User* Agent to a *Mail Transfer* Agent, it is possible to set SSL/TLS for SMTP. What does this setting protect?

18. It is possible to set SSL for IMAP. What does this setting protect?
19. What for can we use GNU Privacy Guard?

Questions for curious minds

1. What for are the Base64 and Quoted-Printable encodings used?