RCOMP - Redes de Computadores (Computer Networks)

2021/2022

Lecture 12

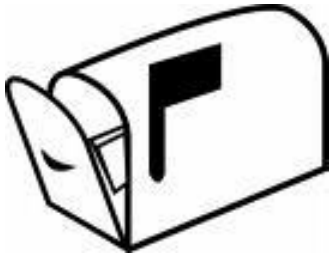
- Electronic mail.
- SMTP, POP3 and IMAP. Webmail.
- MIME.

Electronic mail

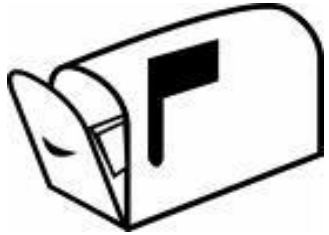
Electronic mail mimics the traditional physical mail. The goal is providing a mail messages delivery system between users, in off-line and non-interactive mode. The main difference is messages are not physically written in paper.

One notorious feature of mail systems is the destination user may not be present when the message arrives (off-line mode), therefore a place is required for the postman (the mailing system) to deposit messages until the user appears to read them.

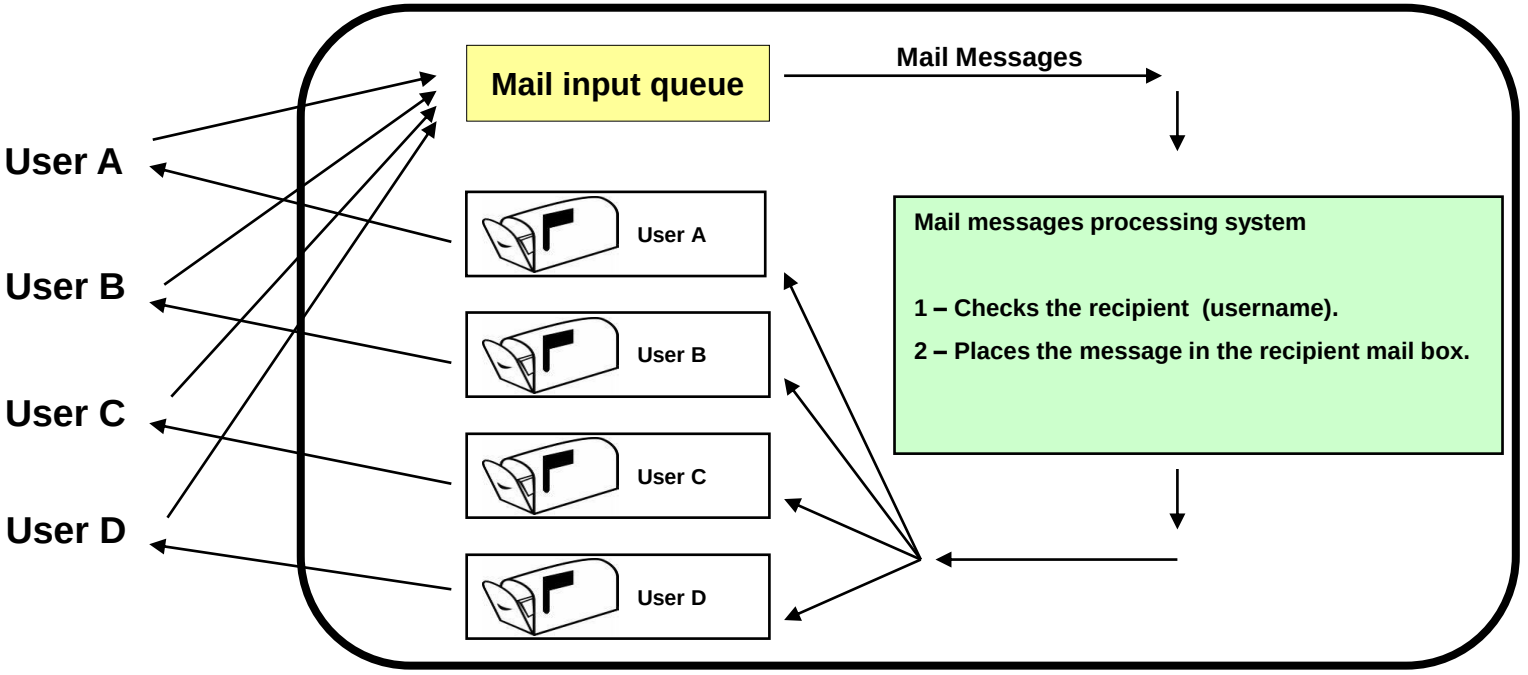
Thus, like in a physical mail system, every user must have a personal storage place for messages intended to him. The mailing system's (postman's) role is depositing there messages intended for that user. This personal storing place for messages is, of course, known as the **mail-box**.



Mailboxes



A typical electronic mail system can be pretty simple:

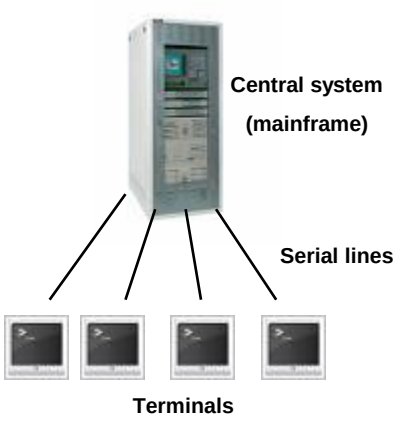


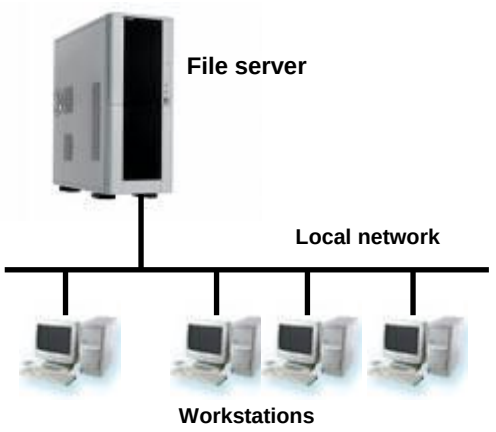
Each user has a mailbox, usually matching the username, only the user and the system (postman) can access the mailbox. Messages have a sender and recipient, they represent mailboxes, but also usernames.

The mail system's role is receiving user messages, usually through a **mail input queue**, and deposit them on the recipient's mail-box.

Shared file system based electronic mail

Electronic mail systems can be implemented by using a shared file system. Both user mailboxes and the input queue are file system objects, usually either files or folders, with the appropriate permissions.
A system like this is confined in a single shared file system (usually in a single server), thus networks are not directly used for mail delivery.





Because under these mail systems' point of view the whole world is their own shared file system, mailboxes can be identified just by the username. However, they will not allow a user in one mail system (one shared file system) to communicate with users in other mail systems (on other shared file systems).

Local mail systems interconnection

With the progressive evolution of computer networks and the internet, arises the need to expand electronic mail systems in such a way users of different local mail systems can communicate with each other.

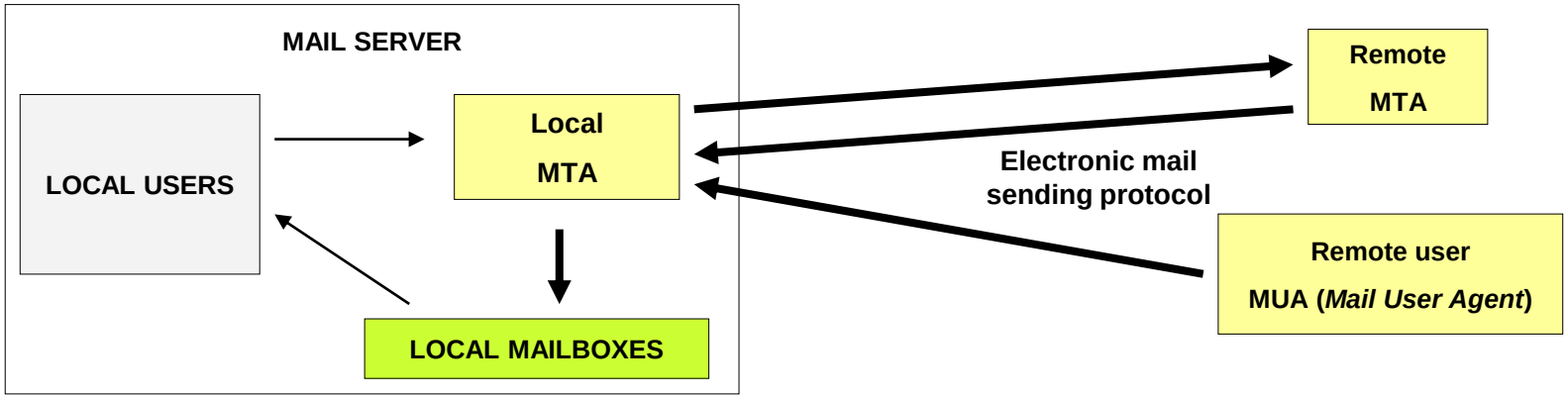
To achieve that, local mail systems are required to use the network infrastructure and a proper **application protocol,** known to both systems, for **messages transfer**.

Mailboxes identification (sender and recipient) must now include the local system identification, where the mailbox is located.

To globally identify a user's mailbox the ***username@system*** form is the most widely used.

MTA – Mail Transport Agent

Each local mail processing system has now the ability to interact with remote mail systems through the network and it's now called a MTA (Mail Transport Agent or Message Transfer Agent).



Local users interact with the local file system as before, both for sending messages (mail input queue) and for reading mail from their own mailboxes.

In addition, the same protocol used to send mail messages between MTAs may as well be used by remote users to send mail messages. This is achieved by using a specific software called MUA (*Mail User Agent*).

Simple Mail Transfer Protocol (SMTP)

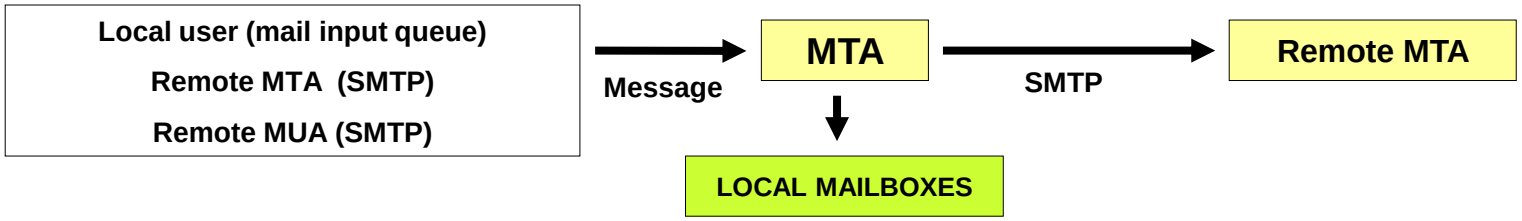
Over the internet (TCP/IP), the most widely used application protocol for sending electronic mail messages is SMTP.

Users (mailboxes) are identified in the form: **USER@DNS-NAME**

Where DNS-NAME refers to the DNS qualified name of the mail server where this user's mailbox is hosted.

When a MTA examines a message, it will verify if the DNS-NAME is its own, in that case deposits the message in the local mailbox for the USER.

On the other hand, if the DNS-NAME belongs to another server, the MTA will contact it (by resolving the DNS-NAME) and then sending the mail message to it through the SMTP protocol.



SMTP – Domain names and MX records

SMTP mailboxes are identified by **USER@DNS-NAME**, the DNS-NAME should be resolved to the IP address of the server holding the USER mailbox, that server will be then contacted via SMTP to send a mail message to be deposited in the specified mailbox.

In practice over the internet, it's more convenient associating mailboxes to **domain names**, rather than host names. The first approach to solve this issue was associating an A and/or AAAA record to the domain name, within the upper-level domain. This record would resolve to the IP address of the mail server holding the mailboxes of the domain.

Nowadays the DNS system supports MX (Mail Exchanger) records to handle this issue more efficiently. MX records are associated with the domain name, and they settle one or several mail servers for that domain.

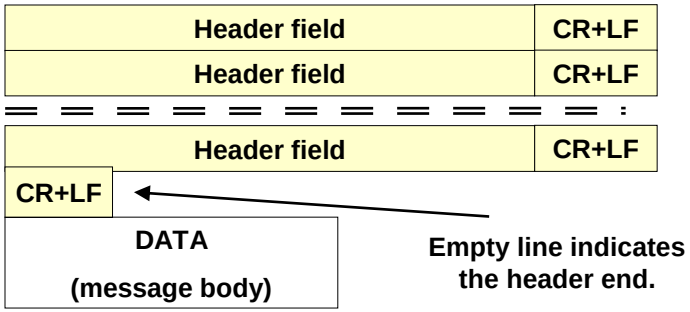
Current MTAs first request the domain MX records, and only if that fails, they will then try for A and AAAA records (the older strategy).

SMTP – Message format

SMTP messages are similar to HTTP messages, though we should say this in reverse because SMTP is far older than HTTP. Unlike with HTTP, there is no request/reply line, but elsewhere it's similar.

The message is started by a sequence of header field lines (CR+LF terminated). An empty line defines the header's end and the content's start.

Yet, SMTP poses several constrains regarding the message: only 7-bit characters are allowed, both in the header and the body.



SMTP HEADER EXAMPLE

```
From: Utilizador <user@dei.isep.ipp.pt>
Subject: Mensagem de Teste
Date: Wed, 21 May 2008 15:54:50 +0100
Reply-To: <user@ipp.pt>
To: <admin@dei.isep.ipp.pt>
Cc: <root@isep.ipp.pt>
Return-Path: <erros@dei.isep.ipp.pt>
Message-ID: <011701c8bb52$a1ca6f10$e55f4d30$@dei.isep.ipp.pt>
In-Reply-To: <8AB511FE5C834F8F8308E52E6437D5DB@ipp.pt>
```

SMTP – Protocol

To send a mail message, the client starts by creating a TCP connection towards the server, for that purpose SMTP servers should be listening on TCP port number 25. Once the connection is established a dialog session starts by using a set of SMTP supported commands (RFC 821).

The example below shows such a dialog, lines sent by the client are highlighted in blue.

```
220 frodo.dei.isep.ipp.pt ESMTP Mailer DEINET-1.1; Wed, 21 May 2008 18:15:30 +0100
HELO frodo.dei.isep.ipp.pt
250 frodo.dei.isep.ipp.pt Hello pci14ppp.dei.isep.ipp.pt [193.136.62.213], pleased to meet you
MAIL FROM:<andre@dei.isep.ipp.pt>
250 2.1.0 <andre@dei.isep.ipp.pt>... Sender ok
RCPT TO:<asc@isep.ipp.pt>
250 2.1.5 <asc@isep.ipp.pt>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
From: "Andre Moreira" <andre@dei.isep.ipp.pt>
To: <asc@isep.ipp.pt>
Subject: Teste

Mensagem de teste
.
250 2.0.0 m4LHFUWx004991 Message accepted for delivery
QUIT
221 2.0.0 frodo.dei.isep.ipp.pt closing connection
```

ESMTP – Extended SMTP or Enhanced SMTP

ESMTP (RFC 1869) allows a broader set of commands than the normal SMTP. If a client wishes to use ESMTP instead of SMTP, it should use the **EHLO** command instead of **HELO** to greet the server.

If the server supports ESMTP, it will reply with a success code (250), otherwise replies with an error code (5xx), it the last case, the client will have to send a HELO and settle for using the normal SMTP.

```
220 frodo.dei.isep.ipp.pt ESMTP Mailer DEINET-1.1; Wed, 21 May 2008 18:46:30 +0100
EHLO frodo.dei.isep.ipp.pt
250-frodo.dei.isep.ipp.pt Hello pci14ppp.dei.isep.ipp.pt [193.136.62.213], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE 33554432
250-DSN
250-ETRN
250-AUTH DIGEST-MD5
250-DELIVERBY
250 HELP
QUIT
```

In the above SMTP session, no message was actually sent but we can see this server supports ESMTP and also which extensions it supports.

SMTP/ESMTP security issues

The main security issue around SMTP architecture results from the absence of a sender's authentication. When an MTA receives a mail message through SMTP, there no way to guarantee the **From:** field matches the user who actually created the message.

The first step to solve this issue is imposing user's authentication with the ESMTP AUTH feature, however, this is possible only if the MTA has direct access to a user's database with authentication credentials. This may be a solution for MUA accesses, of course, the user is then required to use only the specific local MTA (SMTP server) where he has an account.

But that's not all, SMTP is also used for MTA to MTA messages transfer. In fact, when the local MTA receives a message from the MUA, it's able to authenticate the sender, but then it will use SMTP to send the message to the recipient's MTA.

There's no straightforward approach for the recipient's MTA to validate if the message's **From:** field matches the real message's creator.

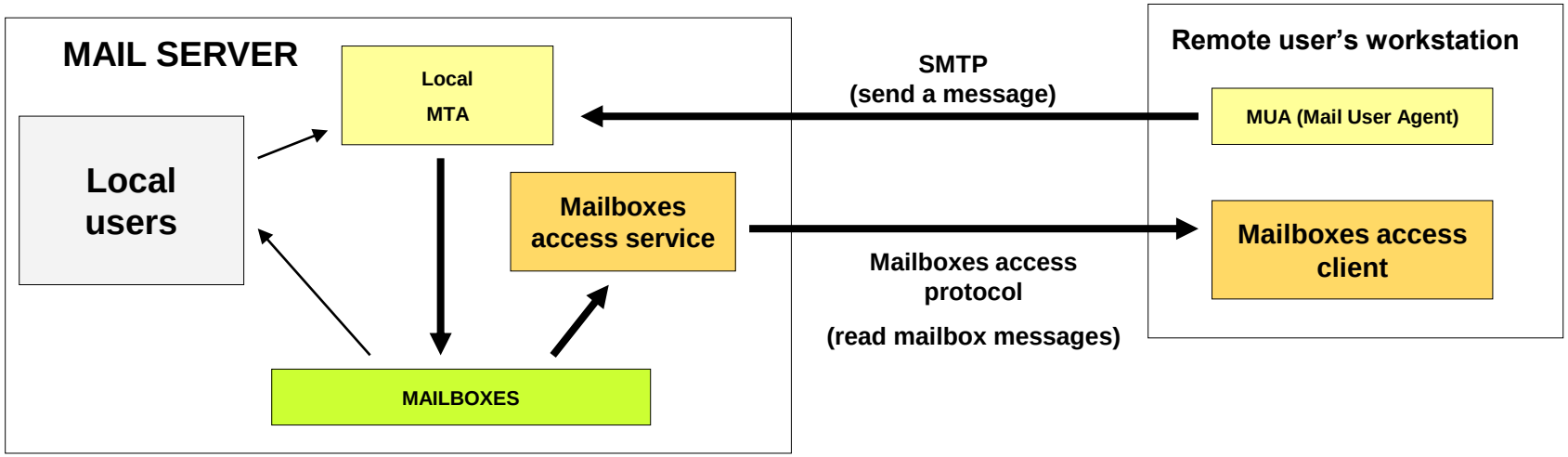
To implement a mail system in a DNS domain, MX records must be defined to publicly announce SMTP servers available to receive mail messages for recipients on the domain. These servers must be publicly accessible by SMTP. To avoid misuse, several security criteria are required:

- Messages from locally authenticated users, as far as the **From:** field is the right one, are always accepted.
- Messages from any remote MTA can't be checked, so they are only accepted if intended to local recipients. Otherwise the server could be used by an attacker to forge messages with arbitrary senders identification. **This is called an open mail relay**.
- As detected, open mail relays are placed in public blacklists, once there, all SMTP servers will block any incoming SMTP message from them. This is another check all SMTP servers should do (if the message is coming from a blacklisted MTA).
- SPF is very helpful here, each DNS domain should define a TXT record declaring which SMTP servers are allowed to send messages with the From: field containing a recipient belonging to the domain. Usually "**v=spf1 +mx –all**" meaning only domain's MX defined servers are allowed. This is yet another check all SMTP servers should do.

Electronic mail – remote mailboxes access

The single purpose of SMTP is sending mail messages, and thus ensure messages are deposited on the recipient's mailbox. Remote users of a mail system can also use SMTP to send messages by using a MUA application.

However, to read messages deposited in their own mailboxes, users must either be locally logged to the server that houses the mailbox (local session), or use an additional and independent application protocol.



The two currently most widely used application protocols for remote mailboxes access are: **IMAP4** and **POP3**.

POP3 – Post Office Protocol version 3

In POP3 (RFC 1939), the client creates a TCP connection toward port number 110 of the POP3 server, and establishes a command based dialog session. As with SMTP commands, they are CR+LF terminated text lines.
After receiving the identification from the server, the POP3 client should authenticate the user, in the following example, as before, lines sent by the client are highlighted in blue:

```
+OK POP3 frodo.dei.isep.ipp.pt 2004.89mdk server ready
USER andre
+OK User name accepted, password please
PASS xxxxxx
+OK Mailbox open, 0 messages
STAT
+OK 0 0
LIST
+OK Mailbox scan listing follows
.
QUIT
+OK Sayonara
```

Username/password authentication (USER/PASS commands) should only be used over a secure connection (**POP3S**). CHAP authentication is also supported by using the **APOP** command.

IMAP4 – Internet Message Access Protocol version 4

POP3 has some shortcomings, it's true if fulfils its basic purposes, it can be used to list messages, download messages and delete messages, but goes no further. It's normally used to download the full mailbox content and then the connection is closed.

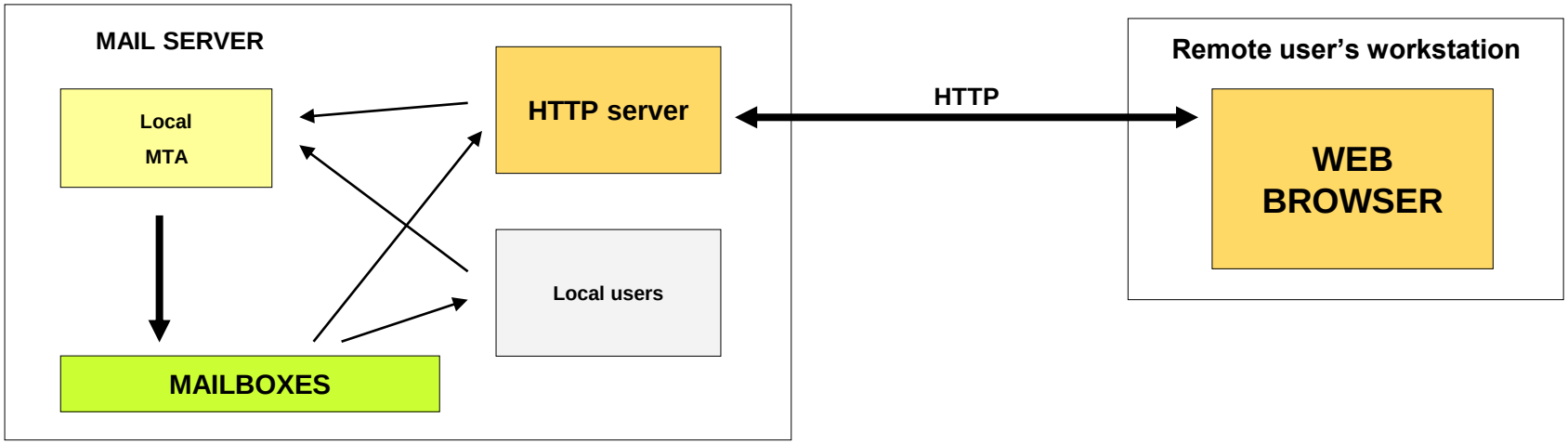
IMAP4 is oriented to rather more long-lasting interactive sessions. It also uses a TCP connection, in this case to port number 143, however, the connection is kept open while the client is running. This allows other features, including server triggered new mail arrival notification.

Among others, IMAP4 features include:

- Getting a list of messages in the mailbox (POP3 also supports this).
- Downloading one specific message (POP3 also supports this).
- Downloading part of a message content.
- Tagging messages with different states (on the server side).
- Organizing the mailbox in folders (on the server side).
- Searching for messages (on the server side).
- Multiple clients connected to the same mailbox.

Webmail

Equivalent features to those provided by IMAP are currently available by using a web server application known as webmail. The webmail application may be CGI based running over a standard HTTP server, or it may be a dedicated HTTP server, either case it will run on the mail server. Remote access is, therefore, achieved by using a **standard web browser**.



Because under the webmail application point of view, both mailboxes and the mail input queue are local, it interacts with them the same way a local user does. One major advantage is neither SMTP, nor any other protocol beyond HTTP is required by the remote user's workstation.

MIME - Multipurpose Internet Mail Extensions

The MIME message format is intended to overcome the simple text limitations, allowing other content types. Although initially developed for electronic mail, MIME is widely used, notably in HTTP where it's implicit.

In SMTP, MIME is optional, a mail message must be explicitly declared to be in MIME format by including the MIME-Version header field, the current version of MIME is 1.0, so the header line will look like: **MIME-Version: 1.0**

Mail messages in MIME format can use content related header fields to identify the content type (*Content-Type:*) and the way that content is encoded (*Content-Transfer-Encoding:*). The last is most important for SMTP.

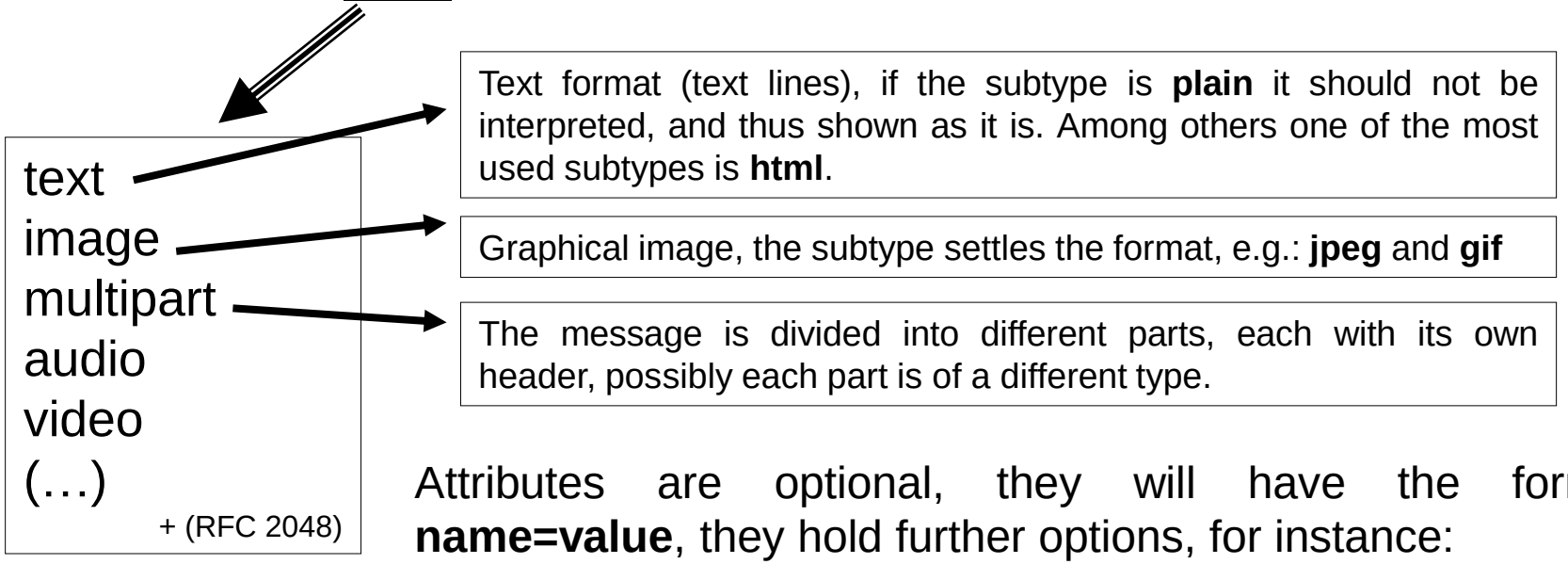
The issue is, SMTP messages are allowed to use only 7-bit characters, nevertheless, thanks to MIME content encoding, these 7-bit characters can ultimately be used to represent any kind of data.

In HTTP the default content encoding is **binary**, this stands for no encoding at all, that's because **HTTP supports a row binary data message body, however SMTP does not.**

MIME – Content-Type header field

Content-Type provides the message reader with information about how content should be interpreted, but only after being decoded.

`Content-Type: type/subtype [; attribute [; attribute […]]]`



Attributes are optional, they will have the form **name=value**, they hold further options, for instance:

```
Content-Type: text/html; charset=us-ascii
```

The role for *Content-Type* is informing the end-user application about how it should interpret the content, and thus how it should be presented to the user. In SMTP many content types are not directly supported, that's where the content encoding comes in.

MIME – Content-Type: multipart

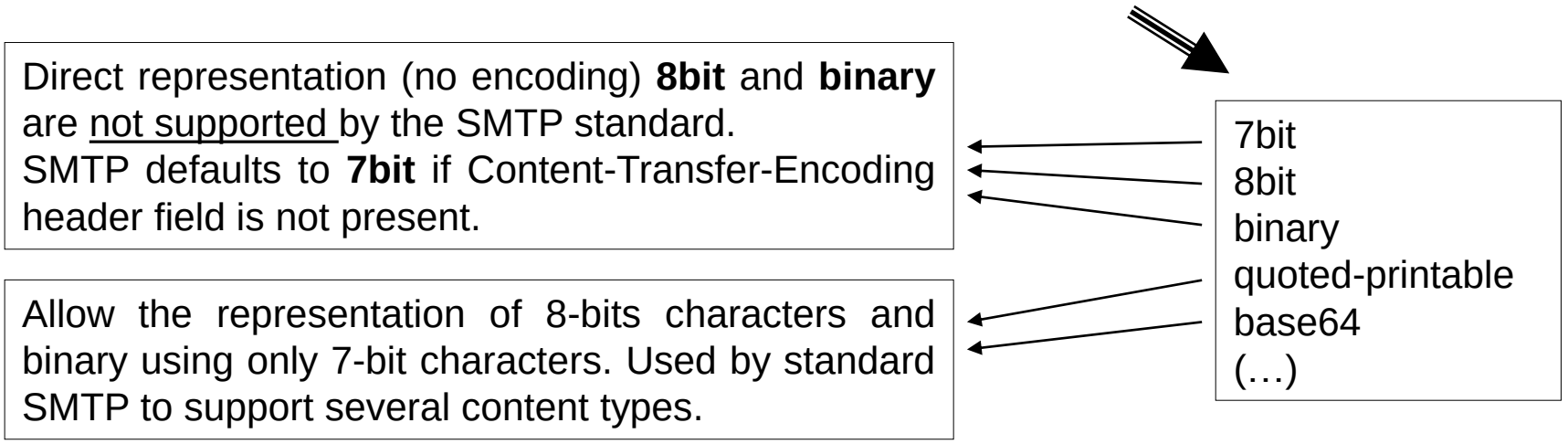
The *multipart* content type allows a message to hold several independent parts of different types. In electronic mail it's typically used to attach files to a message. Example:

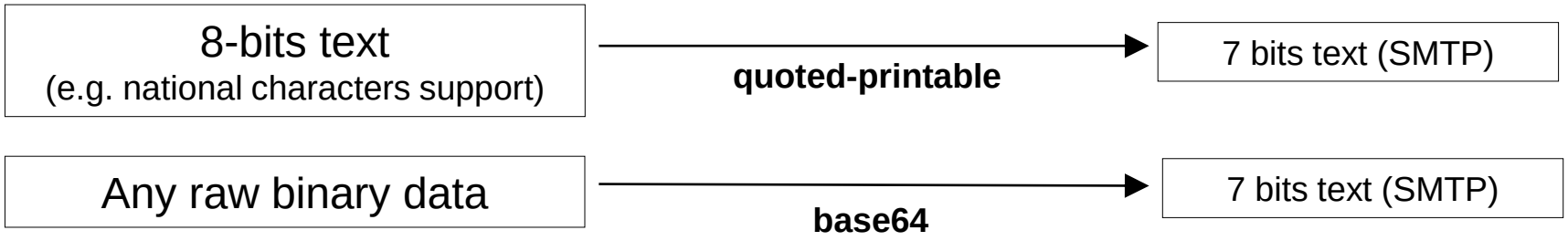
```
MIME-Version: 1.0
Content-Type: multipart/mixed;
           boundary="----=_NextPart_000_038E_01C8BB64.2C534EC0“
Date: Wed, 21 May 2008 17:00:14 +0100

------=_NextPart_000_038E_01C8BB64.2C534EC0
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

Bom dia, em anexo segue o ficheiro pedido.
Cumprimentos

------=_NextPart_000_038E_01C8BB64.2C534EC0
Content-Type: application/msword; name=“documento.doc"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=“documento.doc"

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAA==

------=_NextPart_000_038E_01C8BB64.2C534EC0--
```

MIME – Content-Transfer-Encoding header field

SMTP supports only 7-bits characters (ESMTP may support 8-bits or even binary). Whenever the content is not limited to 7-bit characters, it must be encoded into 7-bit characters, this means the content must be somehow represented by using only 7-bit characters.

`Content-Transfer-Encoding:` **`ENCODING-TYPE`**





MIME – Content-Transfer-Encoding: quoted-printable

Also known as QP encoding, uses 7-bits characters to represent 8-bit characters. Most visible 7-bits characters require no conversion at all, one big exception is the equal signal because it has a special meaning. Basic principles are:

Any 8-bits character, excepting CR and LF may be represented by **=XX**, were XX represents the hexadecimal value of the character's ASCII code.

Characters with decimal ASCII codes from 33 to 60 and 62 to 126 do not require any conversion. The equal symbol (decimal code 61) must therefore be represented by **=3D**

Encoded lines can't be longer than 76 characters (also an SMTP limitation). The equal symbol at the end of an encoded line is a soft break, meaning it's a line break that does not exist on the decoded message.

TAB and spaces (ASCII codes 7 and 32) also don't require any conversion unless they are at the end of the encoded line. On that case, they can either be represented in hexadecimal, or a soft line break can be added.

MIME – Content-Transfer-Encoding: base64

With this encoding, any kind of binary data can be represented by 7-bit characters. However, assuming each 7-bit character is a byte, encoded data will be 33% larger than the original decoded data.

Sixty four 7-bits characters were chosen, they are: **`A..Z a..z 0..9 + /`**

Because $\log_2(64) = 6$, any 6-bits number can be represented using one of these characters.

For each **three bytes** to be encoded there are 24 bits, thus, they will result in **four** 7-bits characters (from those selected above). Encoded text lines are limited to 76 characters long, however encoded text line breaks are discarded in decoding.

Of course, most often the original message length will not be a multiple of three. On that case either one or two zero value bytes are appended to the original data before encoding, for each appended byte one equal symbol is also appended to the encoded text. This informs the decoder it should discard either the last one or the last two bytes.