e-PG Pathshala

Subject : Computer Science

Paper: Computer Networks
Module: Applications – Accessing Mail
Module No: CS/CN/5

Quadrant 1 – e-text

In the previous module, we have discussed about the SMTP protocol used for mail transfer. In this module we will look at a few more mail-related application layer protocols – those that facilitate sending all types of data and attachments via email, and mail access mechanisms. The objectives for this module are as follows.

Learning Objectives

- To discuss the details of the protocols related to e-mail
  - MIME
  - POP3
  - IMAP
  - Web-mail

5.1 SMTP and non-ASCII data

SMTP, we said, uses only 7-bit ASCII based messages. So the question that arises is : how do we send attachments, especially with non-ASCII content, as in images, audio, video etc. The answer to this question is addressed by a message formatting standard known as MIME – Multipurpose Internet Mail Extension, which is used for non-text content. As it can be guessed, MIME consists of mechanisms to convert non-ASCII content to ASCII so that it can be carried using SMTP. It also defines additional headers that are included in the mail message, so that the MIME data can be interpreted appropriately.

5.1.1. MIME - Multipurpose Internet Mail Extension

One of the features of MIME is the support for encoding schemes. One of the common encoding schemes used is the Base-64 encoding. In this scheme, the entire content to be encoded is grouped into 6-bit chunks or base-64 numbers. (With 6 bits remember that we can have 64 combinations). These 6-bit numbers are then mapped to a set of 7-bit ASCII characters so that they can be handled by SMTP (Fig. 5.1).

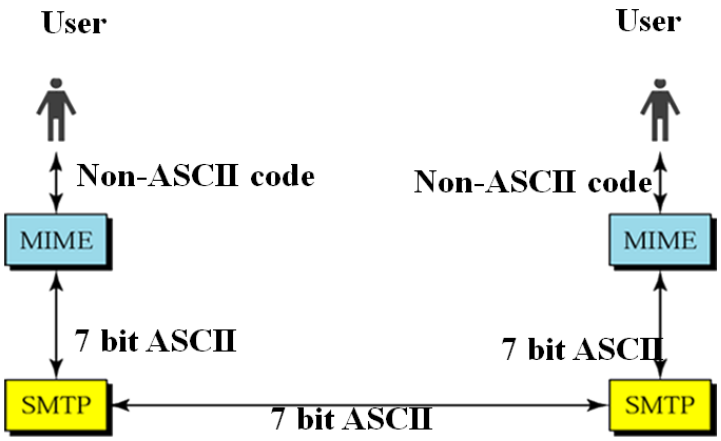

Figure 5.1 Using MIME with SMTP

An example of the base-64 encoding is shown in Fig. 5.2 for three bytes of non-ASCII data. The three bytes are first rearranged as four 6-bit groups, and the ASCII character corresponding to these 6-bits are determined. It is these ASCII characters that are sent as text in the SMTP protocol. On the receiving end, the reverse process will be carried out to get back the three non-ASCII bytes.

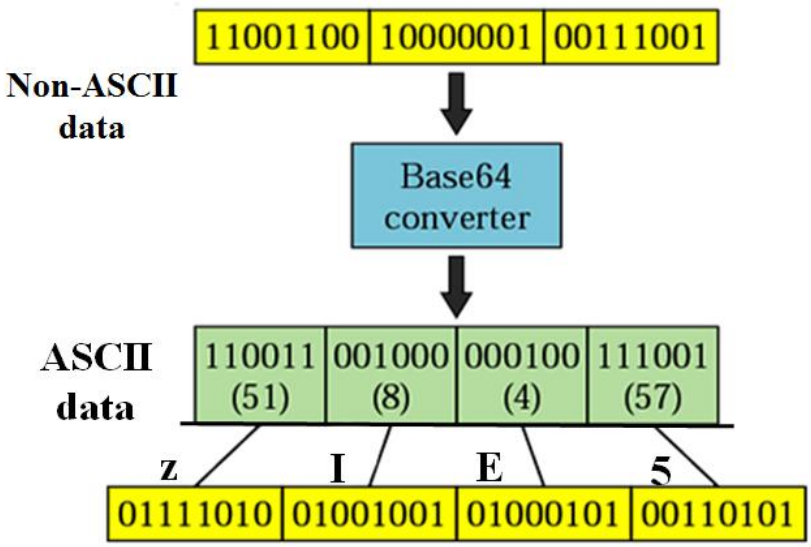

Figure 5.2 Example showing base-64 encoding

A MIME message consists of three basic pieces. The first piece is a collection of header lines that augment the original set of headers of the SMTP protocol as defined by RFC 822. These header lines describe, in various ways, the data being carried in the message body. They include MIME-Version: (the version of MIME being used), Content-Description: (a human-readable description of what's in the message, analogous to the Subject: line), Content-Type: (the type of data contained in the message), and Content-Transfer-Encoding (how the data in the message body is encoded).

The second piece is definitions for a set of content types (and subtypes). For example, MIME defines two different still image types, denoted image/gif and image/jpeg, each with

the obvious meaning.

The third piece is a way to encode the various data types so they can be shipped in an ASCII email message.

Multiple files having ASCII/non-ASCII characters may be sent in the MIME format. If more than one file is sent, there is a boundary used to delimit and distinguish between the files being sent. An example shown in Fig. 5.3 explains this.

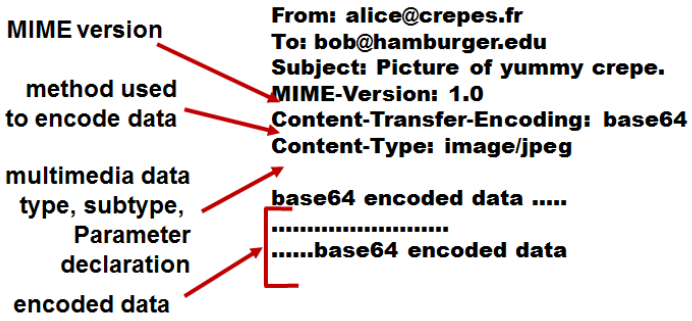

Figure 5.3 MIME-based message format

An example MIME message with multiple attachments (or multiple parts as it is called in MIME), is shown in Fig. 5.4. Note the boundary header line, and also note that this pattern given here (12345678), repeats at the end of each part. Since everything is in 7-bit ASCII, it is self-explanatory. The first part contains plain text, but the content-type of multipart/mixed indicates that there is more than one part. Following this content is the boundary information, and following that we have an image file converted to base64 encoding. Following that is an audio file. Note the dot on a single line all by itself. Recollect that this denotes the end of the message in SMTP.

```
From hacker@buu.ac.th Tue Sep  7 15:13:27 2014
Subject:  You just won one million dollars
MIME-Version: 1.0
Content-Type:  multipart/mixed;
Boundary=Start-12345678
Status: R

--Start-12345678
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline

This is the plain text part of the message.
Next is some binary stuff.

--Start-12345678
Content-Type: image/gif; name="file.gif"
Content-Transfer-Encoding: base64
Content-Description: file.gif
Content-Disposition: inline; filename="file.gif"

R0lGODlhKAAoAPcAAAAAAAgAAAgACBAAABAIABAICBAIEBgIEBgQEBgQGCEQ
ECEQGCEYECEYGCkhGCkhITEpKTExKTExMTkpKTkpMTkxMTkxOTk5MTk5OU
...more...

--Start-12345678
Content-Type: audio/wav; name="chord.wav"
Content-Transfer-Encoding: base64
```

```
Content-Disposition: inline; filename="chord.wav"
/9j/4AAQSkZJRgABAQEASABIAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsL
.
```

Figure 5.4 Mail message with multi-part MIME

The content type indicates, the type of content as text, image, audio, application, etc., and the file type in that category. The following are some examples of this information :

Common Content-Types Parameters

multipart/mixed; Boundary="separator-string"

text/plain charset="tis-620"

text/html charset="windows-874"

image/jpg name="filename.jpg"

image/gif name="filename.gif"

audio/mp3

audio/wav

application/x-zip-compressed name="homepage.zip"

application/msword

A subtype "Parallel" means the parts should be viewed together. This is used for HTML messages containing graphics.

5.2 Extended SMTP

About a decade after SMTP was introduced, a few enhancements were considered to enable secure and efficient mail transfer. The Extended SMTP (or enhanced SMTP) – ESMTP – was the result of this effort (ref - RFC 2821, 5321). There are new headers and verbs (commands) added for this purpose. A few of these are explained below.

- EHLO : This is a command used to check support for ESMTP. Recollect that an SMTP session is initiated by a client opening a connection to the server, and the server responding with an OK response. Then a HELO is sent by the client. Instead, of a HELO command, in ESMTP, the first command to be sent to the other end, is the EHLO to check if the other end supports ESMTP. If it does it will respond with an OK status, else it will send a ‘command not recognized’ kind of response. Then the client will fall back to basic SMTP, and start with the HELO command.
- AUTH : This command is used to authenticate the client to the server. It is used to send login information (username, password etc.). It can be combined with some other keywords, such as PLAIN, LOGIN and CRAM-MD5 (e.g. AUTH LOGIN) to use different login methods and different levels of security. Since username and password could use non-ASCII characters, they are also sent using base-64 encoding.
- STARTTLS : This again is a security related command. This command is used to

send the message using the Transport layer security protocol (TLS). If this command is supported, the message will be encrypted and sent by the transport layer protocol.

- SIZE : This command is used to exchange information on the size of the message being sent. The server can use this to send the maximum size it can accept, and the client can use this to specify the size of the message being sent.
- BDAT & CHUNKING : These commands are used to split a message into smaller parts for efficiency.
- PIPELINING : Instead of sending one command, waiting for its response, and then sending the next command, the commands can be sent in a pipelined manner, in batches. This command is used to indicate this method of sending commands.
- 8BITMIME (RFC 6152) : This is a keyword value used by the server to indicate that it can accept 8-bit mime without converting to 7-bit ASCII. While converting to 7-bit ASCII using some encoding solves the transmission problem, it creates another problem – inflation of size. 8BITMIME is meant to overcome this problem. With this feature, the client can send MIME content aligned to 8-bits.

5.3 Mail access protocols

For many years, the majority of email was moved from host to host using only SMTP. But that is no longer the only protocol in use to move mails. As usage patterns have changed, there is a requirement for newer protocols.

To understand this in the right context, let us recollect how SMTP works. First, users interact with a *mail reader where they compose, file, search, and read their email.*

Second, there is a *mail daemon (or process) running on each host that holds a mailbox.* This is our *mail/message transfer agent (MTA),* which plays the role of a post office: users (or their mail readers) give the daemon messages they want to send to other users, the daemon uses SMTP running over TCP to transmit the message to a daemon running on another machine, and the daemon puts incoming messages into the user's mailbox (where that user's mail reader can later find it).

The next step is for the user to actually retrieve his or her messages from the mailbox, read them, reply to them, and possibly save a copy for future reference. We can see that the mail reader is what makes all these possible.

In the early days of the Internet, users typically logged into the machine on which their *mailbox resided, and the mail reader they invoked was a local application* program that extracted messages from the file system. Since it is on the same machine, it can simply read and write the file that implements the mailbox.

Today, of course, users remotely access their mailbox from their laptop or smartphone; they do not log into the host that stores their mail (a mail server). So there are a new set of mail access protocols that help the user in accessing the mail server, and retrieve the messages from the server. The two predominant mail access protocols are

- the Post Office Protocol (POP) [RFC 1939] and
- the Internet Message Access Protocol (IMAP) [RFC 1730].

Thus the usage of protocols we have today is as shown in Fig. 5.5.

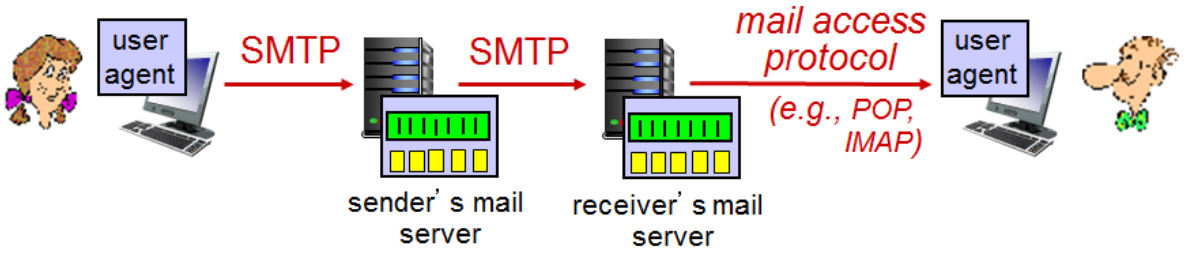

Figure 5.5 Using mail access protocols

In addition to using these protocols to access mail, we also have a third mechanism which has become very much prevalent today ***– web-based mail access***. In this case, the protocol we are using is actually HTTP to connect to the server. We use the web-browser as our interface, and login to the mail server by sending messages over HTTP.

We will examine these three access mechanisms in detail.

5.3.1 Post Office Protocol (POP/POP3)

The Post Office Protocol is a simple protocol that provides for authorization of the agent at the server and download the required mails from the server. The current version in use is POP version 3 or POP3 as it is called. It runs on TCP. The server port number used is 110 for plain text transmission, and 995 for secure (encrypted) transmission. As in the case of SMTP, the commands are sent as plain text.

The protocol cycles through three phases, namely

- authorization,
- transaction, and
- update.

In the *authorization* phase, the commands sent from the client are:

  - user: to declare the username, and
  - pass: to give the password.

The server's responses are either of the following indicating it is correct or in error:

  - +OK
  - -ERR.

In the *transaction* phase, the client's commands are:

- list: to list the message numbers (ids)
- retr: to retrieve the message by number

- dele: to delete the message from the servers mailbox.

In the *update* phase, there is a command

- quit: to update and close the session.

The exchange of commands and responses in a typical POP3 session is shown in the example given in Table 5.1.

Table 5.1 POP3 transaction

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**

In this example, the client gets authenticated, and then sends a list command. Two mails are listed. It then retrieves them one by one and deletes them. This is called the ***download and delete*** mode of POP3. The idea is that if you have downloaded the messages, you do not need to keep a copy on the server.

The disadvantage of this is that if you were to access the POP3 server from different machines (for instance from your laptop and desktop), you will not be able to view it on both machines, because it has been deleted from the server.

The alternative is to use a ***download and keep*** mode. In this mode, the client will not send the “dele” command after retrieving the file, so that it can be accessed from another agent.

In this sense, we say that *POP3 is stateless across sessions.*

5.3.2 Internet Mail Access Protocol (IMAP)

IMAP is similar to SMTP in many ways. It is a client/server protocol running over TCP, where the client (running on the user’s desktop machine) issues commands in the form of <CRLF>-terminated ASCII text lines and the mail server (running on the machine that

maintains the user's mailbox) responds in-kind. The exchange begins with the client authenticating him or herself, and identifying the mailbox he or she wants to access.

The IMAP protocol described in RFC 3501 provides more features than POP3 to manage the mail messages on the server.

1. Often, we would like to organize the mail into different folders. IMAP allows the user to manage multiple folders on server.
2. It is more bandwidth efficient than POP3.

   With IMAP the user can search for messages in the server without having to download them. Another option is provides is that you can download the message headers alone. Also, you can download just 1 part of a multi-part MIME message. These last two options are especially useful when we have multiple attachments that are huge. Instead of downloading all of them, one could first get the headers, determine which part is really needed, and download that part alone. Obviously, this will save bandwidth.

In IMAP, the server maintains more state about the mailbox than POP3. Each message is marked as Seen, Answered, Recent, or Deleted. Deleted messages are erased only when the client issues an "Expunge" command.

The state diagram for IMAP is given in Fig. 5.6. There are four states that IMAP can be in:

(i) Open Connection – Not authenticated,

(ii) Authenticated,

(iii) Selected, and

(iv) Logout.

As soon as the client establishes a TCP connection with the server, the server responds with a greeting. At this stage there are two options – establish an open connection (step 1 of Fig. 5.6) or an authenticated connection (step 2). If step 1 is taken, IMAP connection goes to the 'not authenticated' state. From this state, if a login or authenticate command is issued, it will move to the authenticated state on successful authentication. If step 2 is taken, it directly goes to the authenticated state.

In the authenticated state, several commands may be issued to the server to manipulate the mailboxes. They include: SELECT, EXAMINE, CREATE, DELETE, RENAME, SUBSCRIBE, UNSUBSCRIBE, LIST, LSUB, STATUS, and APPEND. In response to the SELECT or EXAMINE command, the server will move to the Selected state.

In the Selected state, several commands can be given to manipulate the messages in the selected folder. These include CHECK, CLOSE, EXPUNGE, SEARCH, FETCH, STORE, COPY, and UID (unique identifier).

From the Selected state, it will move to the Logout state, if a CLOSE command is given, or if the Select or Examine commands fail.

In the Logout state, a logout command is used to close the connection. The Logout state can also be reached from any of the previous states on receiving a Logout command, or if the connection is rejected by the server (step 3).

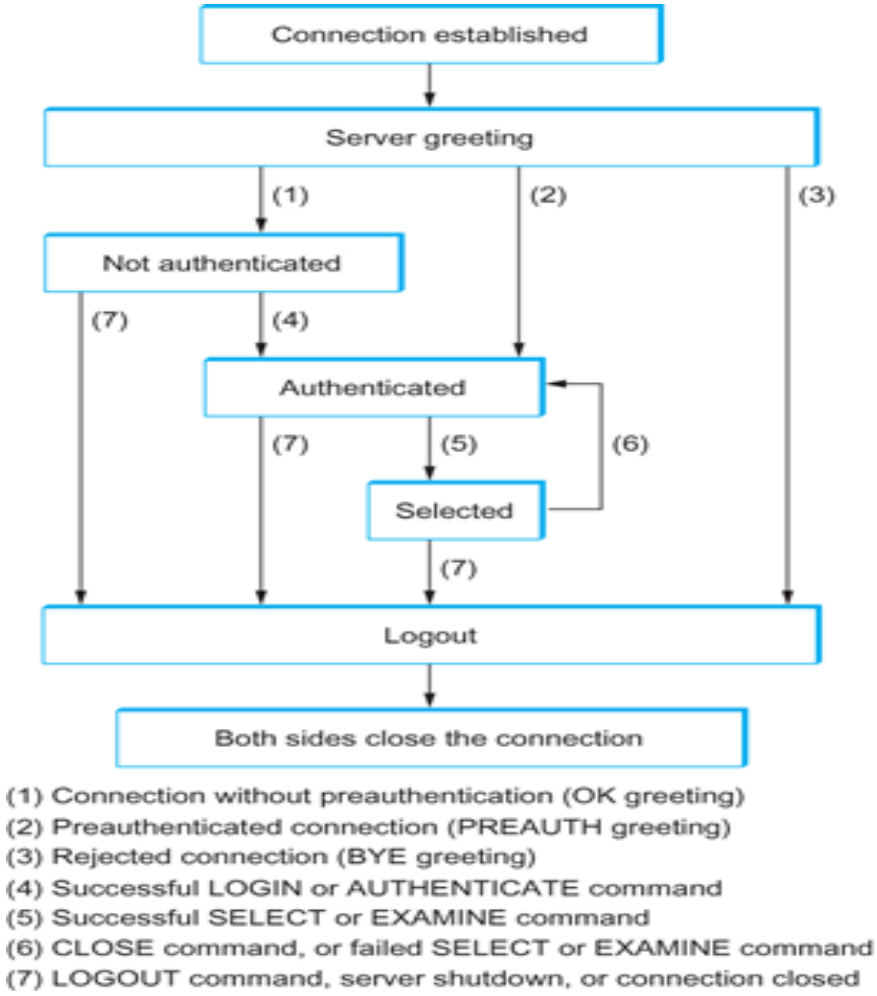

Figure 5.6 IMAP State Transition Diagram

5.3.3. Web E-mail

Web-based e-mail is one of the most popular ways of accessing mailbox, where the web browser becomes the e-mail reader. It gives the flexibility to access mail from anywhere, at any time. Examples of commercial webmail include Yahoo mail, gmail, etc. An example of free webmail is Squirrelmail. With web-email, HTTP becomes the protocol for accessing mail. All messages between the browser, and the web server are HTTP messages.

The e-mail server could be the same as the web server, in which case it can directly access the mail-boxes through some back-end mechanism. Back-end processing is specific to and depends on how e-mail is stored and managed on the system. If the mail server is different from the web server, IMAP or POP3 can be used by the web server to access the messages from the mail server. The web server accepts your authentication (login and password), then connects to the IMAP server. This is more portable, as it is independent of the mail backend. However, it has more overhead, but places less cpu load on the web server.

5.4 Summary

To summarize, we have examined the protocols related to E-mail. After the basic SMTP proposed many decades ago, many enhancements have been done to handle the growing needs of users. These are to do with sending different types of data, as well as the ease of access to mails. We looked at how the MIME protocol enable transfers of all kinds of data including, images, audio and video. We also discussed the extensions to the SMTP itself. Then we discussed the two predominant mail access protocols - POP3 and IMAP. Finally, we briefly discussed we-mail and the access of mail using web browsers.

http://www.samlogic.net/articles/smtp-commands-reference.htm

References:

RFC 5321 - Simple Mail Transfer Protocol (2008 specification)
RFC 1869 - SMTP Service Extensions
RFC 1870 - SMTP Service Extension for Message Size Declaration
RFC 3207 - SMTP Service Extension for Secure SMTP over TLS
RFC 4954 - SMTP Service Extension for Authentication

RFC 6152 - SMTP Service Extension for 8-bit MIME Transport