Email Attachments, MIME and GNU Emacs

Hong Feng
fred@mail.rons.net.cn

March 08, 2002

In issue 02 of FSM, RMS brought us an article in his column — *"We Can Put an End to Word Attachments"*, which suggested how to fight against proprietary software such as MS-Word.

In reality, email attachments, as RMS mentioned, usually have other formats, such as PostScript or PDF. Free software can handle these formats.

IF you are a GNU Emacs user, do you know how to read and create attachments in GNU Emacs?

Frankly speaking, I was quite frustrated by the versions of GNU Emacs earlier than 20.3. My Chinese friends sent me email in Chinese plain text, but I could not read the messages.

Every Chinese character consists of two 8-bit bytes, i.e. two 8-bit characters. However, the Internet, which is based on US 7-bit ASCII characters, converts 8-bit characters to 7-bit characters by removing the eighth bit and substituting another bit for it (a parity bit). To solve the problem, I either asked my friends to FTP the message to my FTP site, or else asked them to uuencode their messages. I could then FTP the messages or else run uudecode to convert the messages that were emailed to me back to the original 8-bit character set.

Either method has its down side: First, while the uuencode/uudecode utility programs come with every Un*x operating system, most of my friends used Microsoft Windows. Microsoft does not include these programs be default in its distributions. Worse, my friends did not know how to find and install the uuencode/uudecode programs on their computers, even though the Microsoft Windows version of uuencode/uudecode is freely available.

Second, anonymous FTP is bad because of poor security. My Internet host's `/var` partitiion is smaller than 500MB. When several malicious guys sent rubbish files to my anonymous FTP directory, they not only filled up my FTP space, they prevented me from receiving incoming email, since email is stored in the same /var partition.

GNU Emacs 20.7 was better than the earlier versions. It could handle base64 encoded email attachments. Unfortunately, this feature was not well documented in GNU Emacs Manual. GNU Emacs 21 is even better.

Before further discussing email, I would like to add some background information about Internet email protocols and concepts in GNU Emacs that are related.

Unlike the UUCP or X.400 email systems, which are based on a store-forward mechanism, the Internet email system adopted a mechanism of end-to-end delivery. The Internet email system is heavily combined with the Domain Name Server (DNS) systems and with Bind.

Most Internet email is delivered by a Mail Transfer Agent (a program called an "MTA") such as sendmail or exim to machine that serves as a cache. Often, this machine is owned by the user's ISP. The user then fetches that email using a Mail Delivery Agent (MDA) program. (An MDA may use the third version of the "Post Office Protocol", POP3, or the "Internet Message Access Protocol", IMAP, or some other protocol.) GNU Emacs is an MDA that can fetch mail using either the POP3 or the IMAP protocols, but not any other protocol, such as POP2.

However, Internet email was designed to delivery 7-bit ASCII email, which mostly means English text. (Early GNU Emacs users did not think this was big problem, since most knew and wrote in English.)

The over all Internet email protocol, which uses POP3 or IMAP for a portion of a message's journey is called SMTP, or the "Simple Mail Transport Protocol". The SMTP protocol defines the header and body for an eamil message. (It is defined in the document called RFC 822. Incidentally, the RFC 822 standard won an early "email standards war" because it depends on the TCP/IP protocol, which succeeded, in contrast to the alternative ISO/OSI protocal, which lost.)

Unfortunately, the SMTP protocol suffers two fatal disadvantage: it can only carry 7-bit ASCII encoded text, and it can only carry one object of text in the body.

Eventually, people recognized SMTP's disadvantages and developed the MIME (Multipurpose Internet Mail Extensions) protocol as a solution. The new protocol enables an email message to carry more than one object in a message, regardless whether the object is text or something else.

The MIME protocol made is possible to creat email attachments: the content transfer header can define a type as encoded binary data, which means that those bytes can contain an email attachments.

Messages that use MIME often encode their binary attachements using an 8-bit to 7-bit conversion protocol called "base64".

Although, the jargon word "base64" is somewhat frightening, it is a simple encoding mechanims. In 7-bit encoding, for example, the English word "GNU", is a series of bits — `"1000111" "1001110" "1010101"`. These segments of 7 bits are first transformed into 8-bit bytes by adding a `"0"` bit in front of each. We get `"01000111" "01001110" "01010101"`.

That is to say:

```
GNU

==> 1000111 1001110 1010101
       ^       ^       ^
    7 bits  7 bits  7 bits

==> 01000111 01001110 01010101
        ^        ^        ^
     8 bits   8 bits   8 bits
```

Then, these three 8-bit bytes — 24 bits total — are are grouped into 4 groups of 6-bits each, like this `"010001" "110100" "111001" "010101"`.

```
01000111 01001110 01010101
   ^        ^        ^
 8 bits   8 bits   8 bits

==>

010001 110100 111001 010101
   ^      ^      ^      ^
6 bits 6 bits 6 bits 6 bits
```

Next, each of the groups is transformed from 6 bits into 8 bit bytes by adding two `"0"` bits in front of the each 6-bit. We get `"00010001" "00110100" "00111001" "00010101"`

```
010001 110100 111001 010101
   ^      ^      ^      ^
6 bits 6 bits 6 bits 6 bits

==> 00010001 00110100 00111001 00010101
       ^        ^        ^        ^
     8 bits   8 bits   8 bits   8 bits
```

These 8-bit bytes are ready to be sent over the Internet. When they are sent over the Internet, the first bit will be removed, since the Internet only deals with 7 bit characters; but that will not effect the end result since the first bit of each of these encoded 8-bit bytes (or "octets" as they are called in some documents) is a zero.

If the original data is Chinese text using the GB2312-80 Chinese encoding format, rather than English text using the ASCII encoding format, the first step will be the same. This is because each Chinese character is made up of two 7-bit ASCII codes.

Now supposed you have correctly configured the GNU Emacs and you have received the email with the encoded attachments. Emacs Lisp — the built-in programming language in GNU Emacs — provided functions to do the bitwise operation which usually only available in C, they are `lsh, ash, logand, logior, logxor, lognot`.

But you need not to understand how to use them now, as GNU Emacs has provided the command `base64-decode-region`, so just provide the parameters and enter the command by `M-x base64-decode-region`, you could get the attachment out.

The base64-decode-region command is easy to use. You need to tell it the region to decode: In GNU Emacs, *a "buffer" is a copy of the file that are you editing*, (Here, the buffer it is the body of the email message on which you are working.) In Emacs, *a "region" is the space between the current location of the cursor, called "point" and a mark*, which can be set using the `C-@` or `C-Spacebar` (`set-mark-command`) keystrokes.

However, you will run into a problem is you try to run the `base64-decode-region` command in an email buffer that is intended for reading: By default, such a buffer is read-only. This means means you can not change the buffer. The `base64-decode-region` command. Instead, you must first copy the region into a new file.

You can use `C-x C-f` to create a new, and in this case empty, file and `C-y` to yank the region — that is to say, to "paste" the region, into the new file. You can use the `C-x C-s` keystrokes to save the file. After you do this, you can mark the whole file using the `C-x h` keystrokes (the `mark-whole-buffer` command) and then decode it.

Now it is time to read your email attachment if you have done the above steps correctly. If the email attachment is a photo (like the one of Miguel de Icaza which I received from his secretary) we published on this issue, use GIMP to view it; if the attachment is a piece of music, then play it with a free MP3 player.

If the attachment is a piece of Chinese text, then you need to open it with GNU Emacs's MULE package. MULE stands for "MULti-lingual Enhancement to GNU Emacs", you could use it to configurate GNU Emacs to support Chinese by setting the code system (to `GB-2312`) and setting language environment (to `Chinese-GB`), after done these, then Chinese text can be displayed.

Have fun with email attachements!

About the Author Hong Feng is the publisher of **FREE SOFTWARE** magazine which you are reading, he started hacking on GNU/Linux in 1995 with 30 floppy discs. As a programmer, he developers device drivers for hardware, Sometimes he also joins his typesetting workshop in Wuhan to produce high quality papers and books with TEX.

He launched MNM Project on March 05, 2001 for supporting the free software community in China. He often visits universities and gives speeches about free software to the students, discusses with professors on curriculum reform for computer science departments, and find programmers for MNM Project. He offers training courses for free software tools. Sometimes he acts as a "doctor in clinic" for free software companies and developers to help them to grow their business by giving practical consulting advice.

In the spare time, he likes reading (especially books about history), studying philosophy and mathematics theory. He can be reached by email `fred@mail.rons.net.cn`.