Communicating Presentation Information in Internet Messages:

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This memo provides a mechanism whereby messages conforming to the MIME specifications [RFC 2045, RFC 2046, RFC 2047, RFC 2048, RFC 2049] can convey presentational information. It specifies the "Content-Disposition" header field, which is optional and valid for any MIME entity ("message" or "body part"). Two values for this header field are described in this memo; one for the ordinary linear presentation of the body part, and another to facilitate the use of mail to transfer files. It is expected that more values will be defined in the future, and procedures are defined for extending this set of values.

This document is intended as an extension to MIME. As such, the reader is assumed to be familiar with the MIME specifications, and [RFC 822]. The information presented herein supplements but does not replace that found in those documents.

This document is a revision to the Experimental protocol defined in RFC 1806. As compared to RFC 1806, this document contains minor editorial updates, adds new parameters needed to support the File Transfer Body Part, and references a separate specification for the handling of non-ASCII and/or very long parameter values.

1. Introduction

MIME specifies a standard format for encapsulating multiple pieces of data into a single Internet message. That document does not address the issue of presentation styles; it provides a framework for the interchange of message content, but leaves presentation issues solely in the hands of mail user agent (MUA) implementors.

Two common ways of presenting multipart electronic messages are as a main document with a list of separate attachments, and as a single document with the various parts expanded (displayed) inline. The display of an attachment is generally construed to require positive action on the part of the recipient, while inline message components are displayed automatically when the message is viewed. A mechanism is needed to allow the sender to transmit this sort of presentational information to the recipient; the Content-Disposition header provides this mechanism, allowing each component of a message to be tagged with an indication of its desired presentation semantics.

Tagging messages in this manner will often be sufficient for basic message formatting. However, in many cases a more powerful and flexible approach will be necessary. The definition of such approaches is beyond the scope of this memo; however, such approaches can benefit from additional Content-Disposition values and parameters, to be defined at a later date.

In addition to allowing the sender to specify the presentational disposition of a message component, it is desirable to allow her to indicate a default archival disposition; a filename. The optional "filename" parameter provides for this. Further, the creation-date, modification-date, and read-date parameters allow preservation of those file attributes when the file is transmitted over MIME email.

NB: The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC 2119].

2. The Content-Disposition Header Field

Content-Disposition is an optional header field. In its absence, the MUA may use whatever presentation method it deems suitable.

It is desirable to keep the set of possible disposition types small and well defined, to avoid needless complexity. Even so, evolving usage will likely require the definition of additional disposition types or parameters, so the set of disposition values is extensible; see below.

In the extended BNF notation of [RFC 822], the Content-Disposition header field is defined as follows:

```
disposition := "Content-Disposition" ":"
               disposition-type
               *(";" disposition-parm)

disposition-type := "inline"
                  / "attachment"
                  / extension-token
                  ; values are not case-sensitive

disposition-parm := filename-parm
                  / creation-date-parm
                  / modification-date-parm
                  / read-date-parm
                  / size-parm
                  / parameter

filename-parm := "filename" "=" value

creation-date-parm := "creation-date" "=" quoted-date-time

modification-date-parm := "modification-date" "=" quoted-date-time

read-date-parm := "read-date" "=" quoted-date-time

size-parm := "size" "=" 1*DIGIT

quoted-date-time := quoted-string
                  ; contents MUST be an RFC 822 `date-time'
                  ; numeric timezones (+HHMM or -HHMM) MUST be used
```

NOTE ON PARAMETER VALUE LENGHTS: A short (length <= 78 characters) parameter value containing only non-`tspecials' characters SHOULD be represented as a single `token'. A short parameter value containing only ASCII characters, but including `tspecials' characters, SHOULD be represented as `quoted-string'. Parameter values longer than 78 characters, or which contain non-ASCII characters, MUST be encoded as specified in [RFC 2184].

`Extension-token', `parameter', `tspecials' and `value' are defined according to [RFC 2045] (which references [RFC 822] in the definition of some of these tokens). `quoted-string' and `DIGIT' are defined in [RFC 822].

2.1. The Inline Disposition Type

A bodypart should be marked `inline' if it is intended to be displayed automatically upon display of the message. Inline bodyparts should be presented in the order in which they occur, subject to the normal semantics of multipart messages.

2.2. The Attachment Disposition Type

Bodyparts can be designated `attachment' to indicate that they are separate from the main body of the mail message, and that their display should not be automatic, but contingent upon some further action of the user. The MUA might instead present the user of a bitmap terminal with an iconic representation of the attachments, or, on character terminals, with a list of attachments from which the user could select for viewing or storage.

2.3. The Filename Parameter

The sender may want to suggest a filename to be used if the entity is detached and stored in a separate file. If the receiving MUA writes the entity to a file, the suggested filename should be used as a basis for the actual filename, where possible.

It is important that the receiving MUA not blindly use the suggested filename. The suggested filename SHOULD be checked (and possibly changed) to see that it conforms to local filesystem conventions, does not overwrite an existing file, and does not present a security problem (see Security Considerations below).

The receiving MUA SHOULD NOT respect any directory path information that may seem to be present in the filename parameter. The filename should be treated as a terminal component only. Portable specification of directory paths might possibly be done in the future via a separate Content-Disposition parameter, but no provision is made for it in this draft.

Current [RFC 2045] grammar restricts parameter values (and hence Content-Disposition filenames) to US-ASCII. We recognize the great desirability of allowing arbitrary character sets in filenames, but it is beyond the scope of this document to define the necessary mechanisms. We expect that the basic [RFC 1521] `value' specification will someday be amended to allow use of non-US-ASCII characters, at which time the same mechanism should be used in the Content-Disposition filename parameter.

Beyond the limitation to US-ASCII, the sending MUA may wish to bear in mind the limitations of common filesystems. Many have severe length and character set restrictions. Short alphanumeric filenames are least likely to require modification by the receiving system.

The presence of the filename parameter does not force an implementation to write the entity to a separate file. It is perfectly acceptable for implementations to leave the entity as part of the normal mail stream unless the user requests otherwise. As a consequence, the parameter may be used on any MIME entity, even `inline' ones. These will not normally be written to files, but the parameter could be used to provide a filename if the receiving user should choose to write the part to a file.

2.4. The Creation-Date parameter

The creation-date parameter MAY be used to indicate the date at which the file was created. If this parameter is included, the paramter value MUST be a quoted-string which contains a representation of the creation date of the file in [RFC 822] `date-time' format.

UNIX and POSIX implementors are cautioned that the `st_ctime' file attribute of the `stat' structure is not the creation time of the file; it is thus not appropriate as a source for the creation-date parameter value.

2.5. The Modification-Date parameter

The modification-date parameter MAY be used to indicate the date at which the file was last modified. If the modification-date parameter is included, the paramter value MUST be a quoted-string which contains a representation of the last modification date of the file in [RFC 822] `date-time' format.

2.6. The Read-Date parameter

The read-date parameter MAY be used to indicate the date at which the file was last read. If the read-date parameter is included, the parameter value MUST be a quoted-string which contains a representation of the last-read date of the file in [RFC 822] `date- time' format.

2.7. The Size parameter

The size parameter indicates an approximate size of the file in octets. It can be used, for example, to pre-allocate space before attempting to store the file, or to determine whether enough space exists.

2.8. Future Extensions and Unrecognized Disposition Types

In the likely event that new parameters or disposition types are needed, they should be registered with the Internet Assigned Numbers Authority (IANA), in the manner specified in Section 9 of this memo.

Once new disposition types and parameters are defined, there is of course the likelihood that implementations will see disposition types and parameters they do not understand. Furthermore, since x-tokens are allowed, implementations may also see entirely unregistered disposition types and parameters.

Unrecognized parameters should be ignored. Unrecognized disposition types should be treated as `attachment'. The choice of `attachment' for unrecognized types is made because a sender who goes to the trouble of producing a Content-Disposition header with a new disposition type is more likely aiming for something more elaborate than inline presentation.

Unless noted otherwise in the definition of a parameter, Content-Disposition parameters are valid for all dispositions. (In contrast to MIME content-type parameters, which are defined on a per-content-type basis.) Thus, for example, the `filename' parameter still means the name of the file to which the part should be written, even if the disposition itself is unrecognized.

2.9. Content-Disposition and Multipart

If a Content-Disposition header is used on a multipart body part, it applies to the multipart as a whole, not the individual subparts. The disposition types of the subparts do not need to be consulted until the multipart itself is presented. When the multipart is displayed, then the dispositions of the subparts should be respected.

If the `inline' disposition is used, the multipart should be displayed as normal; however, an `attachment' subpart should require action from the user to display.

If the `attachment' disposition is used, presentation of the multipart should not proceed without explicit user action. Once the user has chosen to display the multipart, the individual subpart dispositions should be consulted to determine how to present the subparts.

2.10. Content-Disposition and the Main Message

It is permissible to use Content-Disposition on the main body of an [RFC 822] message.

3. Examples

Here is a an example of a body part containing a JPEG image that is intended to be viewed by the user immediately:

```
      Content-Type: image/jpeg
      Content-Disposition: inline
      Content-Description: just a small picture of me

       <jpeg data>
```

The following body part contains a JPEG image that should be displayed to the user only if the user requests it. If the JPEG is written to a file, the file should be named "genome.jpg". The recipient's user might also choose to set the last-modified date of the stored file to date in the modification-date parameter:

```
      Content-Type: image/jpeg
      Content-Disposition: attachment; filename=genome.jpeg;
        modification-date="Wed, 12 Feb 1997 16:29:51 -0500";
      Content-Description: a complete map of the human genome

      <jpeg data>
```

The following is an example of the use of the `attachment' disposition with a multipart body part. The user should see text-part-1 immediately, then take some action to view multipart-2. After taking action to view multipart-2, the user will see text-part-2 right away, and be required to take action to view jpeg-1. Subparts are indented for clarity; they would not be so indented in a real message.

```
      Content-Type: multipart/mixed; boundary=outer
      Content-Description: multipart-1

      --outer
        Content-Type: text/plain
        Content-Disposition: inline
        Content-Description: text-part-1

        Some text goes here

      --outer
        Content-Type: multipart/mixed; boundary=inner
        Content-Disposition: attachment
        Content-Description: multipart-2

        --inner
          Content-Type: text/plain
          Content-Disposition: inline
          Content-Description: text-part-2

          Some more text here.

        --inner
          Content-Type: image/jpeg
          Content-Disposition: attachment
          Content-Description: jpeg-1

          <jpeg data>
        --inner--
      --outer--
```

4. Summary

Content-Disposition takes one of two values, `inline' and `attachment'. `Inline' indicates that the entity should be immediately displayed to the user, whereas `attachment' means that the user should take additional action to view the entity.

The `filename' parameter can be used to suggest a filename for storing the bodypart, if the user wishes to store it in an external file.

5. Security Considerations

There are security issues involved any time users exchange data. While these are not to be minimized, neither does this memo change the status quo in that regard, except in one instance.

Since this memo provides a way for the sender to suggest a filename, a receiving MUA must take care that the sender's suggested filename does not represent a hazard. Using UNIX as an example, some hazards would be:

• Creating startup files (e.g., ".login").
• Creating or overwriting system files (e.g., "/etc/passwd").
• Overwriting any existing file.
• Placing executable files into any command search path (e.g., "~/bin/more").
• Sending the file to a pipe (e.g., "| sh").

In general, the receiving MUA should not name or place the file such that it will get interpreted or executed without the user explicitly initiating the action.

It is very important to note that this is not an exhaustive list; it is intended as a small set of examples only. Implementors must be alert to the potential hazards on their target systems.

6. References

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC2184]   Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2184, August 1997.

[RFC2045]   Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2046]   Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[RFC2047]   Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[RFC2048]   Freed, N., Klensin, J., and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", RFC 2048, November 1996.

[RFC2049]   Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996.

[RFC822]   Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

7. Acknowledgements

We gratefully acknowledge the help these people provided during the preparation of this draft:

Nathaniel Borenstein

Ned Freed

Keith Moore

Dave Crocker

Dan Pritchett

8. Authors' Addresses

You should blame the editor of this version of the document for any changes since RFC 1806:

Keith Moore (editor)
University of Tennessee
Computer Science Dept.
107 Ayres Hall
Knoxville, TN 37996-1301
USA
EMail: moore@cs.utk.edu

The authors of RFC 1806 are:

Rens Troost
New Century Systems
324 East 41st Street #804
New York, NY 10017
USA
Phone: +1 (212) 557-2050
Fax: +1 (212) 557-2049
EMail: rens@century.com

Steve Dorner
QUALCOMM Incorporated
6455 Lusk Boulevard
San Diego, CA 92121
USA
EMail: sdorner@qualcomm.com

9. Registration of New Content-Disposition Values and Parameters

New Content-Disposition values (besides "inline" and "attachment") may be defined only by Internet standards-track documents, or in Experimental documents approved by the Internet Engineering Steering Group.

New content-disposition parameters may be registered by supplying the information in the following template and sending it via electronic mail to IANA@IANA.ORG:

```
To: IANA@IANA.ORG
Subject: Registration of new Content-Disposition parameter

Content-Disposition parameter name:

Allowable values for this parameter:
     (If the parameter can only assume a small number of values,
     list each of those values.  Otherwise, describe the values
     that the parameter can assume.)
Description:
     (What is the purpose of this parameter and how is it used?)
```

10. Changes since RFC 1806

The following changes have been made since the earlier version of this document, published in RFC 1806 as an Experimental protocol:

- Updated references to MIME documents. In some cases this involved substituting a reference to one of the current MIME RFCs for a reference to RFC 1521; in other cases, a reference to RFC 1521 was simply replaced with the word "MIME".

- Added a section on registration procedures, since none of the procedures in RFC 2048 seemed to be appropriate.

- Added new parameter types: creation-date, modification-date, read-date, and size.

- Incorporated a reference to draft-freed-pvcsc-* for encoding long or non-ASCII parameter values.

- Added reference to RFC 2119 to define MUST, SHOULD, etc. keywords.