Packaging Aggregate HTML Objects Inside MIME

Status of this Memo

Abstract

   Although HTML was designed within the context of MIME, more than the
   specification of HTML as defined in RFC 1866 is needed for two
   electronic mail user agents to be able to interoperate using HTML as
   a document format. These issues include the naming of objects that
   are normally referred to by URIs, and the means of aggregating
   objects that go together. This draft describes a set of guidelines
   that will allow conforming mail user agents to be able to send,
   deliver and display these HTML objects. In addition it is hoped that
   these techniques will also apply to the wider category of URI-enabled
   objects.

Table of Contents

[1]. Introduction

[1.1] Purpose

Although HTML [1] is a valid MIME [2] type, RFC 1866 does not
provide enough specification in order for two electronic mail
user agents to be able to interoperate using HTML as a document
format. This draft describes a set of guidelines that will
allow conforming mail user agents to be able to send, deliver
and display HTML objects. In addition it is hoped that these
techniques will also apply to the wider category of URI-enabled
[3] objects.

An HTML aggregate object is a MIME-encoded message that
contains an HTML document as well as other data that is
required in order to represent that object (inline pictures,
style sheets, etc.). HTML aggregate objects can also include
additional HTML documents that are linked to the first object,
as well as other arbitrary MIME content.

In designing HTML capabilities for electronic mail user agents
(UAs), it is important to keep in mind the differing needs of
several audiences. Mail sending agents will send aggregate HTML
objects as an encoding of normal day-to-day electronic mail.
Mail sending agents will also send aggregate HTML objects when
a user wishes to mail a particular document from the World Wide
Web to someone else. Finally mail sending agents will send
aggregate HTML documents as automatic responders, providing
access to WWW resources for non-IP connected clients.

Mail receiving agents also have several differing needs. Some
mail receiving agents will be able to receive an aggregate HTML
document and display it just as any other text content type
would be displayed. Others will have to pass this aggregate
HTML document to an HTML browsing program, and provisions need
to be made to make this possible.

Finally several other constraints on the problem arise. It is
important that it be possible for an HTML document to be signed

and for it to be able to be transmitted to a client and
displayed with a minimum chance of breaking the message
integrity check that is part of the signature.

1.2 Overall Operation

A mail user agent that wishes to send a content-type of HTML
can just do so, so long as the normal data encoding issues are
taken care of as specified in [2]. However at a basic level
there are some differences between HTML being transferred by
HTTP and HTML being transferred through Internet email. When
transferred through HTTP, HTML by default uses the document
character set ISO-8859-1. Within electronic mail, the default
character set is US-ASCII. If a document uses any characters
that are not in US-ASCII, the document must explicitly label
the character set and perform appropriate MIME content
encodings. Instead of applying normal MIME content encodings it
is possible to translate non US-ASCII characters to HTML
defined entity references. However it is inappropriate to use
entity references to non-US-ASCII characters without labeling
the document character set appropriately.

1.3 URI References

The use of URI references creates some additional issues for
aggregate HTML objects. Normal URI references can of course be used,
however it is likely that many user agents may not be able to
retrieve those objects referred to. This document provides a means
for these additional objects to be transmitted with the HTML and for
the links between these objects to be properly resolved.

1.3.1 Use of Multipart/Related

Multiple objects should be aggregated using the
multipart/related content type as defined in RFC 1872 [4]. RFC
1872 says that multipart/related should have parameters
"start", and "type". The "start" parameter refers to the
Content-ID of the sub-part which contains the main document, in
this case, usually the primary HTML document. The main document
should be the one first displayed by the receiving UA. The
"type" parameter serves as a label for the type of the
aggregate object, in this case "text/html".

1.3.2 Content-ID URL References

An HTML body part can use Content-ID URLs as described by draft-
levinson-cid-01.txt [5] to refer to other body parts of the
same MIME message. Content-ID URLs can also refer to body parts
in other MIME messages, but it is unlikely that many clients
will be able to resolve the reference.

1.3.3 New MIME Content Headers

In order to resolve URI references to other body parts, two new MIME content headers are required. Both of these place URIs in MIME header fields. Since MIME header fields have a limited length and URIs can get quite long, these lines may have to be folded. When the lines are folded, no additional non-white space characters may be introduced, and since white space is not allowed in URIs it is simply ignored.

1.3.3.1 Content-Base

The Content-Base header field may be included in any MIME content header. It specifies the Content-Base for the body part and should be a full URL. Any relative URL references within the body part are made relative to the body parts "base URL". An HTML body part can also include a <BASE> tag. If it does contain a <BASE> tag, that <BASE> tag takes precedence over the Content-Base header field.

1.3.3.2 Content-Location

The Content-Location content header may be included in any MIME subpart header. It specifies the URI that corresponds to the object present in that subpart. If a URL is specified using the Content-Location header, it should be a fully qualified URL.

1.4 Other MIME issues

Several other tricky issues may exist regarding the deployment of HTML email, however they are out of the scope of this document. These issues include the use of multipart/alternative for content negotiation, the use of mail/WWW gateways, and the use of URLs referring to objects outside of the encapsulation (both to WWW-based objects, objects in other MIME documents, and objects in other parts of the same MIME message but not in the same multipart/related).

The use of multipart/alternative is in no way an HTML-specific issue and no clear solution exists at this time for the problem of content negotiation though electronic mail. The use of mail/WWW gateways should be facilitated by the provisions of this document. However this document makes no attempt to specify the format of a request to such a gateway. The use of references to outside of the "encapsulating MIME object" is not something that can be prohibited, but it is simply something that the sending UA needs to realize creates a danger of the receiving UA not being able to resolve the reference.

Finally, a sending user agent should not make any assumptions about the method that the receiving user agent will use to display the HTML files. For example it should not use Content-Disposition and/or "file:" URLs under the assumption that the receiving UA is going to save the pieces of the HTML aggregate

object as files on a disk to be displayed by a separate
browser. Content-Disposition should only be used as described
in RFC 1806 in order to distinguish between file attachments
and inline message components.

2. **Examples**

The first example is the simplest form of an HTML email message.
This is not an aggregate HTML object, but simply one by itself. This
message contains a hot-link but does not provide the ability to
resolve the hot-link. To resolve the hot-link the receiving client
would need either IP access to the Internet, or an electronic mail
web gateway.

```
From: some.user@resnova.com
To: someone.else@entropy.net
Subject: Hello there
Mime-Version: 1.0
Content-Type: text/html

<html>
<head></head>
<body>
<h1>Hi there!</h1>
This is a rather pointless example of an HTML message.<p>
Try clicking <a href=3D"http://www.resnova.com/">here.</a><p>
</body></html>
```

The second example shows a simple HTML document with a picture in it:

```
From: some.user@resnova.com
To: someone.else@entropy.net
Subject: Hello there
Mime-Version: 1.0
Content-Type: multipart/related; boundary=3Dabcdefghij
    start=3D"<12345@entropy.net>"; type=3D"text/html"

--abcdefghij
Content-Type: text/html
Content-ID: <12345@entropy.net>
<html>
<head></head>
<body>
<h1>Hi there!</h1>
This is a rather pointless example of an HTML message.<p>
<img src=3D"cid:<45678@entropy.net>"><p>
Try clicking <a href=3D"http://www.resnova.com/">here.</a><p>
</body></html>
```

```
--abcdefghij
Content-Type: image/jpeg
Content-ID: <45678@entropy.net>
Content-Transfer-Encoding: Base64

jdisfdsufhsdjvbfdhvbfhbvfhbvfdvifdjrivjifdjvfivfd
etc=85
--abcdefghij=97
```

The third example shows the use of Content-Location. This
example could be a web page that was mailed to someone. Note
that the starting body part still needs a Content-ID.

```
From: some.user@resnova.com
To: someone.else@entropy.net
Subject: Hello there
Mime-Version: 1.0
Content-Type: multipart/related; boundary=3Dabcdefghij
    start=3D"<12345@entropy.net>"; type=3D"text/html"

--abcdefghij
Content-Type: text/html
Content-ID: <12345@entropy.net>
Content-Location: http://www.entropy.net/mydoc.html
<html>
<head></head>
<body>
<h1>Hi there!</h1>
This is a rather pointless example of an HTML message.<p>
<img src=3D"http://www.entropy.net/myimg.jpg"><p>
Try clicking <a href=3D"http://www.resnova.com/">here.</a><p>
</body></html>

--abcdefghij
Content-Type: image/jpeg
Content-Location: http://www.entropy.net/myimg.jpg
Content-Transfer-Encoding: Base64

jdisfdsufhsdjvbfdhvbfhbvfhbvfdvifdjrivjifdjvfivfd
etc=85
--abcdefghij=97
```

3. Security

Some Security Considerations include the potential to mail someone
an object, and claim that it is represented by a particular URI (by
giving it a Content-Location: header field). There can be no
assurance that a WWW request for that same URI would normally result
in that same object. Because of this problem, receiving User Agents
should not cache this data in the same way that data that was

retrieved through an HTTP or FTP request might be cached.

In addition, by allowing people to mail aggregate HTML objects, we are opening the door to other potential security problems that until now were only problems for WWW users. For example, some HTML documents now either themselves contain executable content (JavaScript) or contain links to executable content (The "INSERT" specification, Java). It would be exceedingly dangerous for a receiving User Agent to execute content received through a mail message without careful attention to restrictions on the capabilities of that executable content.

4. Acknowledgments

Thanks to Dave Crocker, Roy Fielding, Ed Levinson, and Paul Hoffman who, with me, worked out most of the details of this draft in an informal discussion at the Dallas December 1995 IETF. Additional thanks to Greg Herlihy and Ed Levinson for reviewing this draft. Thanks also to Jacob Palme who encouraged the existence of the Mail HTML effort and addressed many of the issues in his drafts.

5 References

[1] T. Berners-Lee and D. Connolly.
    "HyperText Markup Language Specification 2.0"
    RFC 1866, Proposed Standard
    <URL: http://ds.internic.net/rfc/rfc1866.txt>, November 1995.

[2] N. Borenstein and N. Freed.
    "MIME (Multipurpose Internet Mail Extensions) Part One"
    RFC 1521, Proposed Standard
    <URL: http://ds.internic.net/rfc/rfc1521.txt>, September 1993.

[3] T. Berners-Lee, L. Masinter, and M. McCahill
    "Uniform Resource Locators (URL)"
    RFC 1738, Proposed Standard
    <URL: http://ds.internic.net/rfc/rfc1738.txt>, December 1994.

[4] E. Levinson
    "The MIME Multipart/Related Content-type"
    RFC 1872, Experimental
    <URL: http://ds.internic.net/rfc/rfc1872.txt>, December 1995.

6 Author's Address

Alex Hopmann
alex.hopmann@resnova.com
President
ResNova Software, Inc.
5011 Argosy Dr. #13
Huntington Beach, CA 92649