Network Working Group
Request for Comments: 1830
Category: Experimental

G. Vaudreuil
Octel Network Services

August 1995

SMTP Service Extensions for Transmission of Large and Binary MIME Messages

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. This memo does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

1. Abstract

This memo defines two extensions to the SMTP service. The first service enables a SMTP client and server to negotiate the use of an alternate DATA command "BDAT" for efficiently sending large MIME messages. The second extension takes advantage of the BDAT command to permit the negotiated sending of unencoded binary data.

2. Introduction

The MIME extensions to the Internet message protocol provides for the transmission of many kinds of data which were previously unsupported in Internet mail. Anticipating the need to more efficiently transport the new media made possible with MIME, the SMTP protocol has been extended to provide transport for new message types. RFC 1426 defines one such extension for the transmission of unencoded 8 bit MIME messages [8BIT]. This service extension permits the receiver SMTP to declare support for 8 bit body parts and the sender to request 8 bit transmission of a particular message.

One expected result of the use of MIME is that the Internet mail system will be expected to carry very large mail messages. In such transactions, there is a need to eliminate the requirement that the message be scanned for "CR LF . CR LF" sequences upon sending and receiving to detect the end of message.

Independent of the need to send large messages, Internet mail is increasingly multi-media there is a need to avoid the overhead of base64 and quoted-printable encoding of binary objects sent using the MIME message format over SMTP between hosts which support binary message processing.

This memo uses the mechanism defined in [ESMTP] to define two extensions to the SMTP service whereby a client ("sender-SMTP") may declare support for the message chunking transmission mode using the BDAT command and support for the sending of Binary messages.

3. Framework for the Large Message Extensions

The following service extension is hereby defined:

1) The name of the data chunking service extension is "CHUNKING".

2) The EHLO keyword value associated with this extension is "CHUNKING".

3) A new SMTP verb is defined "BDAT" as an alternative to the "DATA" command of [RFC821]. The BDAT verb takes two arguments. The first argument indicates the length of the binary data packet. The second optional argument indicates that the data packet is the last.

```
bdat-cmd   ::= "BDAT" SP chunk-size
                  [ SP end-marker ] CR LF
chunk-size ::= 1*DIGIT
end-marker ::= "LAST"
```

The CHUNKING service extension enables the use of the BDAT alternative to the DATA command. This extension can be used for any message, whether 7 bit, 8BITMIME or BINARYMIME.

When a client SMTP wishes to submit (using the MAIL command) a large message using the CHUNKING extension, it first issues the EHLO command to the server SMTP. If the server SMTP responds with code 250 to the EHLO command, and the response includes the EHLO keyword value CHUNKING, then the server SMTP is indicating that it supports the BDAT command and will accept the sending of messages in chunks.

After all MAIL FROM and RCPT TO responses are collected and processed, the message is sent using a series of BDAT commands. The BDAT command takes one argument, the exact length of the data segment in octets. The message data is sent immediately after the BDAT command. Once the receiver-SMTP receives the specified number of octets, it will return a 250 reply code.

The LAST parameter on the BDAT command indicates that this is the last chunk of message data to be sent. Any BDAT command sent after the BDAT LAST is illegal and must be replied to with a 503 "Bad

sequence of commands" reply code. The state resulting from this error is indeterminate. A RSET command must be sent to clear the transaction before continuing.

A 250 response should be sent to each BDAT data block. If a 5XX code is sent in response to a BDAT chunk the message should be considered failed and, the sender SMTP must not send any additional BDAT segments. If using the ESMTP pipelining extensions [PIPE], the sender SMTP must complete the sending of the current segment and not send any more BDATs. When streaming, the receiver SMTP must accept and discard additional BDAT chunks after the failed BDAT. After receiving a 5XX error in response to a BDAT command, the resulting state is indeterminate. A RSET command must be issued to clear the transaction before additional commands may be sent.

> Note that an error on the receiver SMTP such as disk full or imminent shutdown can only be reported after the BDAT segment has been sent. It is therefore important to choose a reasonable chunk size given the expected end to end bandwidth.

The RSET command when issued during after the first BDAT and before the BDAT LAST clears all segments sent during that transaction and resets the session.

DATA and BDAT commands cannot be used in the same transaction. If a DATA statement is issued after a BDAT for the current transaction, a 503 "Bad sequence of commands" must be issued. The state resulting from this error is indeterminate. A RSET command must be sent to clear the transaction before continuing. There is no prohibition on using DATA and BDAT in the same session, so long as they are not mixed in the same transaction.

The local storage size of a message may not accurately reflect the actual size of the message sent due to local storage conventions. In particular, text messages sent with the BDAT command must be sent in the canonical MIME format with lines delimited with a <CR><LF>. It may not be possible to convert the entire message to the canonical format at once. Chunking provides a mechanism to convert the message to canonical form, accurately count the bytes, and send the message a single chunk at a time.

> Note that correct byte counting is essential. If too many bytes are indicated by the sender SMTP, the receiver SMTP will continue to wait for the remainder of the data or will read the subsequent command as additional message data. In the case where a portion of the previous command was read as data, the parser will return a syntax error when the incomplete command is read.

If too few bytes are indicated by the sender SMTP, the receiver SMTP will interpret the remainder of the message data as invalid commands. Note that the remainder of the message data may be binary and as such lexigraphical parsers must be prepared to receive, process, and reject lines of arbitrary octets.

4. Framework for the Binary Service Extension

The following service extension is hereby defined:

1) The name of the binary service extension is "BINARYMIME".

2) The EHLO keyword value associated with this extension is "BINARYMIME".

3) The BINARYMIME service extension can only be used with the "CHUNKING" service extension.

4) No parameter is used with the BINARYMIME keyword.

5) One additional parameter to the BODY keyword defined [8BIT] for the MAIL FROM command is defined, "BINARYMIME". The value "BINARYMIME" associated with this parameter indicates that this message is a Binary MIME message (in strict compliance with [MIME]) with arbitrary octet content being sent. The revised syntax of the value is as follows, using the ABNF notation of [RFC822]:

   body-value ::= "7BIT" / "8BITMIME" / "BINARYMIME"

6) No new verbs are defined for the BINARYMIME extension.

A sender SMTP may request that a binary MIME message be sent without transport encoding by sending a BINARYMIME parameter with the MAIL FROM command. When the receiver SMTP accepts a MAIL FROM command with the BINARYMIME body type requested, it agrees to preserve all bits in each octet passed using the BDAT command.

BINARYMIME cannot be used with the DATA command. If a DATA command is issued after a MAIL FROM command containing the body-value of "BINARYMIME", a 501 response should be sent. The resulting state from this error condition is indeterminate and the transaction should be reset with the RSET command.

It is important to note that when using BINARYMIME, it is especially important to ensure that the MIME message itself is properly formed. In particular, it is essential that text be canonically encoded with each line properly terminated with <CR>

<LF>. Any transformation of text into non-canonical MIME to observe local storage conventions must be reversed before sending as BINARYMIME. The usual line-oriented shortcuts will break if used with BINARYMIME.

The syntax of the extended MAIL command is identical to the MAIL command in [RFC821], except that a BODY parameter must appear after the address. The complete syntax of this extended command is defined in [ESMTP]. The ESMTP-keyword is BODY and the syntax for ESMTP-value is given by the syntax for body-value in [ESMTP].

If a receiver SMTP does not support the BINARYMIME message format (either by not responding with code 250 to the EHLO command, or by rejecting the BINARYMIME parameter to the MAIL FROM command, then the client SMTP must not, under any circumstances, send binary data using the DATA or BDAT commands.

If the receiver-SMTP does not support BINARYMIME and the message content is a MIME object with a binary encoding, a client SMTP has two options in this case: first, it may implement a gateway transformation to convert the message into valid 7bit encoded MIME, or second, it may treat this as a permanent error and handle it in the usual manner for delivery failures. The specifics of the transformation from Binary MIME to 7bit MIME are not described by this RFC; the conversion is nevertheless constrained in the following ways:

- The conversion must cause no loss of information; MIME transport encodings must be employed as needed to insure this is the case.

- The resulting message must be valid 7bit MIME.

As of present there are no mechanisms for converting a binary MIME object into a 8 bit-MIME object. Such a transformation will require the specification of a new MIME content-transfer-encoding, the standardization of which is discouraged by [MIME].

5. Examples

5.1 Simple Chunking

The following simple dialogue illustrates the use of the large message extension to send a short psudo-RFC822 message to one recipient using the CHUNKING extension:

```
R: <wait for connection on TCP port 25>
S: <open connection to server>
R: 220 cnri.reston.va.us SMTP service ready
S: EHLO ymir.claremont.edu
R: 250-cnri.reston.va.us says hello
R: 250 CHUNKING
S: MAIL FROM:<Sam@Random.com>
R: 250 <Sam@Random.com>... Sender ok
S: RCPT TO:<Susan@Random.com>
R: 250 <Susan@random.com>... Recipient ok
S: BDAT 69 LAST
S: To: Susan@random.com<CR><LF>
S: From: Sam@random.com<CR><LF>
S: Subject: This is a bodyless test message<CR><LF>
R: 250 Message OK, 69 octets received
S: QUIT
R: 221 Goodbye
```

5.2 Pipelining Binarymime

The following dialogue illustrates the use of the large message extension to send a BINARYMIME object to two recipients using the CHUNKING and PIPELINING extensions:

```
R: <wait for connection on TCP port 25>
S: <open connection to server>
R: 220 cnri.reston.va.us SMTP service ready
S: EHLO ymir.claremont.edu
R: 250-cnri.reston.va.us says hello
R: 250-PIPELINING
R: 250-BINARYMIME
R: 250 CHUNKING
S: MAIL FROM:<ned@ymir.claremont.edu> BODY=BINARYMIME
S: RCPT TO:<gvaudre@cnri.reston.va.us>
S: RCPT TO:<jstewart@cnri.reston.va.us>
R: 250 <ned@ymir.claremont.edu>... Sender and BINARYMIME ok
R: 250 <gvaudre@cnri.reston.va.us>... Recipient ok
R: 250 <jstewart@cnri.reston.va.us>... Recipient ok
S: BDAT 100000
```

```
S: (First 10000 octets of canonical MIME message data)
S: BDAT 324 LAST
S: (Remaining 324 octets of canonical MIME message data)
R: 250 100000 bytes received
R: 250 Message OK, 100324 octets received
S: QUIT
R: 221 Goodbye
```

6. Security Considerations

This RFC does not discuss security issues and is not believed to raise any security issues not already endemic in electronic mail and present in fully conforming implementations of [RFC821], or otherwise made possible by [MIME].

7. Acknowledgments

This document is the result of numerous discussions in the IETF SMTP Extensions Working Group and in particular due to the continued advocacy of "chunking" by Neil Katin.

8. References

[RFC821] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, USC/Information Sciences Institute, August 1982.

[RFC822] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, UDEL, August 1982.

[MIME] Borenstein, N., and N. Freed, "Multipurpose Internet Mail Extensions", RFC 1521, Bellcore, Innosoft, June 1992.

[ESMTP] Klensin, J., WG Chair, Freed, N., Editor, Rose, M., Stefferud, E., and D. Crocker, "SMTP Service Extensions" RFC 1425, United Nations University, Innosoft International, Inc., Dover Beach Consulting, Inc., Network Management Associates, Inc., The Branch Office, February 1993.

[8BIT] Klensin, J., WG Chair, Freed, N., Editor, Rose, M., Stefferud, E., and D. Crocker, "SMTP Service Extension for 8bit-MIMEtransport" RFC 1426, United Nations University, Innosoft International, Inc., Dover Beach Consulting, Inc., Network Management Associates, Inc., The Branch Office, February 1993.

[PIPE] Freed, N., "SMTP Service Extensions for Command Pipelining", Innosoft International, Work in Progress.

9. Author's Address

Gregory M. Vaudreuil
Octel Network Services
17060 Dallas Parkway
Suite 214
Dallas, TX 75248-1905

Voice/Fax: 214-733-2722
EMail: Greg.Vaudreuil@Octel.com