Network Working Group
Request for Comments: 3030
Obsolete: 1830
Category: Standards Track

G. Vaudreuil
Lucent Technologies

December 2000

SMTP Service Extensions for Transmission of Large and Binary MIME 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.

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

Abstract

This memo defines two extensions to the SMTP (Simple Mail Transfer Protocol) service. The first extension enables a SMTP client and server to negotiate the use of an alternative to the DATA command, called "BDAT", for efficiently sending large MIME (Multipurpose Internet Mail Extensions) messages. The second extension takes advantage of the BDAT command to permit the negotiated sending of MIME messages that employ the binary transfer encoding. This document is intended to update and obsolete RFC 1830.

Working Group Summary

This protocol is not the product of an IETF working group, however the specification resulted from discussions within the ESMTP working group. The resulting protocol documented in RFC 1830 was classified as experimental at that time due to questions about the robustness of the Binary Content-Transfer-Encoding deployed in then existent MIME implementations. As MIME has matured and other uses of the Binary Content-Transfer-Encoding have been deployed, these concerns have been allayed. With this document, Binary ESMTP is expected to become standards-track.

Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Table of Contents

1. Overview 2
2. Framework for the Large Message Extensions 3
3. Framework for the Binary Service Extension 5
4. Examples 8
4.1 Simple Chunking 8
4.2 Pipelining BINARYMIME 8
5. Security Considerations 9
6. References 9
7. Author's Address 10
8. Appendix A - Changes from RFC 1830 11
9. Full Copyright Statement 12

1. Overview

The MIME extensions to the Internet message format provides for the transmission of many kinds of data that were previously unsupported in Internet mail. Anticipating the need to transport the new media more efficiently, the SMTP protocol has been extended to provide transport for new message types. RFC 1652 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 performance-based desire 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 multimedia. 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 that support binary message processing.

This memo uses the mechanism defined in [ESMTP] to define two extensions to the SMTP service whereby an SMTP server ("receiver-SMTP") may declare support for the message chunking transmission mode and support for the reception of Binary messages, which the SMTP client ("sender-SMTP") is then free to use.

2. 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, BDAT, is defined as an alternative to the "DATA" command of [RFC821]. The BDAT verb takes two arguments. The first argument indicates the length, in octets, of the binary data chunk. The second optional argument indicates that the data chunk is the last.

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

4) This extension may be used for SMTP message submission. [Submit]

5) Servers that offer the BDAT extension MUST continue to support the regular SMTP DATA command. Clients are free to use DATA to transfer appropriately encoded to servers that support the CHUNKING extension if they wish to do so.

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

After all MAIL and RCPT responses are collected and processed, the message is sent using a series of BDAT commands. The BDAT command takes one required argument, the exact length of the data segment in

octets. The message data is sent immediately after the trailing <CR> <LF> of the BDAT command line. Once the receiver-SMTP receives the specified number of octets, it will return a 250 reply code.

The optional LAST parameter on the BDAT command indicates that this is the last chunk of message data to be sent. The last BDAT command MAY have a byte-count of zero indicating there is no additional 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 MUST be sent to each successful BDAT data block within a mail transaction. If a failure occurs after a BDAT command is received, the receiver-SMTP MUST accept and discard the associated message data before sending the appropriate 5XX or 4XX code. If a 5XX or 4XX code is received by the sender-SMTP in response to a BDAT chunk, the transaction should be considered failed and the sender-SMTP MUST NOT send any additional BDAT segments. If the receiver-SMTP has declared support for command pipelining [PIPE], the receiver SMTP MUST be prepared to accept and discard additional BDAT chunks already in the pipeline after the failed BDAT.

> Note: An error on the receiver-SMTP such as disk full or imminent shutdown can only be reported after the BDAT segment has been received. It is therefore important to choose a reasonable chunk size given the expected end-to-end bandwidth.
>
> Note: Because the receiver-SMTP does not acknowledge the BDAT command before the message data is sent, it is important to send the BDAT only to systems that have declared their capability to accept BDAT commands. Illegally sending a BDAT command and associated message data to a non-CHUNKING capable system will result in the receiver-SMTP parsing the associated message data as if it were a potentially very long, ESMTP command line containing binary data.

The resulting state from a failed BDAT command is indeterminate. A RSET command MUST be issued to clear the transaction before additional commands may be sent. The RSET command, when issued 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: Correct byte counting is essential. If the sender-SMTP indicates a chunk-size larger than the actual chunk-size, the receiver-SMTP will continue to wait for the remainder of the data or when using streaming, 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 the sender-SMTP indicates a chunk-size smaller than the actual chunk-size, 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 lexicographical parsers MUST be prepared to receive, process, and reject lines of arbitrary octets.

3. 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) [8BIT] defines the BODY parameter for the MAIL command. This extension defines an additional value for the BODY parameter, "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.

7) This extension may be used for SMTP message submission. [Submit]

8) The maximum length of a MAIL FROM command line is increased by 16 characters by the possible addition of the BODY=BINARYMIME keyword and value;.

A sender-SMTP may request that a binary MIME message be sent without transport encoding by sending a BODY parameter with a value of "BINARYMIME" with the MAIL command. When the receiver-SMTP accepts a MAIL command with the BINARYMIME body-value, it agrees to preserve all bits in each octet passed using the BDAT command. Once a receiver-SMTP supporting the BINARYMIME service extension accepts a message containing binary material, the receiver-SMTP MUST deliver or relay the message in such a way as to preserve all bits in each octet.

BINARYMIME cannot be used with the DATA command. If a DATA command is issued after a MAIL command containing the body-value of "BINARYMIME", a 503 "Bad sequence of commands" response MUST be sent. The resulting state from this error condition is indeterminate and the transaction MUST be reset with the RSET command.

It is especially important when using BINARYMIME 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. Some line-oriented shortcuts will break if used with BINARYMIME. A sender-SMTP MUST use the canonical encoding for a given MIME content-type. In particular, text/* MUST be sent with <CR><LF> terminated lines.

Note: Although CR and LF do not necessarily represent ends of text lines in BDAT chunks and use of the binary transfer encoding is allowed, the RFC 2781 prohibition against using a UTF-16 charset within the text top-level media type remains.

The syntax of the extended MAIL command is identical to the MAIL command in [RFC821], except that a BODY=BINARYMIME parameter and value MUST be added. The complete syntax of this extended command is defined in [ESMTP].

If a receiver-SMTP does not indicate support the BINARYMIME message format then the sender-SMTP MUST NOT, under any circumstances, send binary data.

If the receiver-SMTP does not support BINARYMIME and the message to be sent is a MIME object with a binary encoding, a sender-SMTP has three options with which to forward the message. First, if the receiver-SMTP supports the 8bit-MIMEtransport extension [8bit] and the content is amenable to being encoded in 8bit, the sender-SMTP may implement a gateway transformation to convert the message into valid 8bit-encoded MIME. Second, it may implement a gateway transformation to convert the message into valid 7bit-encoded MIME. Third, it may treat this as a permanent error and handle it in the usual manner for delivery failures. The specifics of MIME content-transfer-encodings, including transformations from Binary MIME to 8bit or 7bit MIME are not described by this RFC; the conversion is nevertheless constrained in the following ways:

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

2. The resulting message MUST be valid 7bit or 8bit MIME. In particular, the transformation MUST NOT result in nested Base-64 or Quoted-Printable content-transfer-encodings.

Note that at the time of this writing there are no mechanisms for converting a binary MIME object into an 8-bit MIME object. Such a transformation will require the specification of a new MIME content-transfer-encoding.

If the MIME message contains a "Binary" content-transfer-encoding and the BODY parameter does not indicate BINARYMIME, the message MUST be accepted. The message SHOULD be returned to the sender with an appropriate DSN. The message contents MAY be returned to the sender if the offending content can be mangled into a legal DSN structure. "Fixing" and forwarding the offending content is beyond the scope of this document.

4. Examples

4.1 Simple Chunking

The following simple dialogue illustrates the use of the large message extension to send a short pseudo-RFC 822 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 86 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, 86 octets received
S: QUIT
R: 221 Goodbye
```

4.2 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
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
S: (Remaining 324 octets of canonical MIME message data)
S: BDAT 0 LAST
R: 250 100000 octets received
R: 250 324 octets received
R: 250 Message OK, 100324 octets received
S: QUIT
R: 221 Goodbye
```

5. Security Considerations

This extension is not known to present any additional security issues not already endemic to electronic mail and present in fully conforming implementations of [RFC821], or otherwise made possible by [MIME].

6. References

[BINARY] Vaudreuil, G., "SMTP Service Extensions for Transmission of Large and Binary MIME Messages", RFC 1830, August 1995.

[RFC821] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1982.

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

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

[SUBMIT] Gellens, R. and J. Klensin, "Message Submission", RFC 2476, December 1998.

[ESMTP] Klensin, J., Freed, N., Rose, M., Stefferud, E. and D. Crocker, "SMTP Service Extensions", RFC 1869, November 1995.

[8BIT] Klensin, J., Freed, N., Rose, M., Stefferud, E. and D. Crocker, "SMTP Service Extension for 8bit-MIMEtransport", RFC 1652, July 1994.

[PIPE] Freed, N., "SMTP Service Extensions for Command Pipelining", RFC 2920, September 2000.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

7. Author's Address

Gregory M. Vaudreuil
Lucent Technologies
17080 Dallas Parkway
Dallas, TX 75248-1905

Phone/Fax: +1-972-733-2722
EMail: GregV@ieee.org

Appendix A - Changes from RFC 1830

Numerous editorial changes including required intellectual property boilerplate and revised authors contact information

Corrected the simple chunking example to use the correct number of bytes. Updated the pipelining example to illustrate use of the BDAT 0 LAST construct.

Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.