Internet-Draft

T. Finch
University of Cambridge

February 2006

SMTP service extension for specifying recipients in the message header

Abstract

This document describes an SMTP service extension for use with RFC 2476 message submission, which allows a Message User Agent to submit a message without any RCPT commands, and instead it requests the Message Submission Agent to extract the list of recipients from the message's header. This may be useful for simple MUAs operating in restricted environments.

Document revision

$Cambridge: hermes/doc/qsmtp/draft-fanf-smtp-rcpthdr.xml,v 1.7 2006/03/14 13:16:40 fanf2 Exp $

Table of Contents

1. Introduction

[RFC2476] specifies a profile of SMTP [RFC2821] for introducing new messages into the Message Transfer Agent (MTA) routing network. It includes provision for a Message User Agent (MUA) to submit unfinished messages which are fixed up by the Message Submission Agent (MSA), for example by adding Date: and/or Message-ID: header fields as necessary.

However, the MUA must still generate the message envelope - the MAIL and RCPT commands - and transmit it to the MSA. This leads to redundancy between the message envelope and the message header, which can be significant for small messages or messages tranmitted over low-bandwidth links (for an example, see Section 5.1). It also adds complexity which can be a concern for MUAs that must operate in extremely restricted environments.

This document specifies an SMTP service extension for use with [RFC2476] message submission which allows an MUA to submit a message without any recipients specified in the envelope, and instead request the MSA to generate the list of recipients based on the message's header.

Note: A number of existing systems have similar simplified message submission arrangements, albeit not based on SMTP. These include the non-standard POP3 XTND XMIT command and the Unix "sendmail -t" command. The sections of this specification that describe how to generate a recipient list from a message header are also relevant to implementations of these interfaces, and to the components of full-featured MUAs that generate a recipient list for normal message submission.

This document arose from considering what must be implemented on the client end and the server end of message submission, and the observation that both must parse the message header to extract addresses: the client in order to create the envelope, and the server in order to perform any fix-ups that may be necessary. The RCPTHDR extension allows the client to offload this duplicated effort to the server.

2. Terminology

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 [RFC2119].

Where this specification requires a message to be "rejected", this

should be interpreted as described in [RFC2476] section 3.2.

Message Transfer Agent (MTA): A full SMTP implementation that handles message routing, relay, and delivery.

Message Submission Agent (MSA): An SMTP server which handles the introduction of messages into the MTA network.

Message User Agent (MUA): An SMTP client which submits messages via an MSA. MUAs are typically also POP3 [RFC1939] or IMAP [RFC3501] clients, but that is not relevant to this specification.

In the message submission examples, commands issued by the MUA and the message data it sends are prefixed by "u:", and responses issued by the MSA are prefixed by "S:". In the onward transmission examples, commands issued by the MSA and the message data it sends are prefixed by "s:", and responses issued by the MTA are prefixed by "T:". SMTP clients have a lower-case prefix and SMTP servers have an upper-case prefix. For simplicity, the examples omit details such as TLS and SASL negotiation which should be present in a real message submission trace.

3. SMTP service extension declaration

The SMTP service extension is defined as follows:

1. the name of the SMTP service extension is "specifying recipients in the message header";

2. the EHLO keyword value associated with the extension is "RCPTHDR";

3. there are no parameters associated with the RCPTHDR EHLO keyword value;

4. one optional parameter to the MAIL command with the keyword "RCPTHDR" which does not have an associated value;

5. the maximum length of the MAIL command is increased by 8 characters;

6. there are no additional parameters to the RCPT command defined by this extension;

7. there are no additional SMTP verbs defined by this extension.

4. Specifying recipients in the message header

An MSA that allows messages to be submitted with the recipients specified in the message header instead of in the envelope MUST advertise this with the RCPTHDR EHLO keyword. MSAs SHOULD only advertise this extension to clients that have been properly authenticated, e.g. via [RFC2554]. MTAs that act as simple message relays or that are referred to by MX records or otherwise accept messages from the public Internet SHOULD NOT advertise this extension.

An MUA that wishes to submit a message with the recipients specified in the message header instead of in the envelope MUST include the RCPTHDR parameter to the MAIL command that starts the transaction. It MUST NOT then issue any RCPT commands as part of the same transaction, but instead proceed straight to the DATA command.

When a message is submitted by an MUA to an MSA with the RCPTHDR parameter to the MAIL command, the MSA MUST process the message header as described in the following sections, in order to extract the list of recipients and perform any modifications that might be necessary. The MSA SHOULD apply its policies to this list as it would to recipients specified in RCPT commands, and reject the message if necessary, as described in [RFC2476].

This specification is made somewhat more complicated by the need to handle both completely new messages, and messages that are being re-sent. There are in fact four possibilities: new messages, described in Section 5; re-sent messages conforming to [RFC2822], described in Section 6; re-sent messages conforming to [RFC0822], described in Section 7; and badly-formed messages that do not fall into the above categories, described in Section 8.

5. Extracting recipients from a new message’s header

This section applies to new messages, which includes replies to previous messages. These messages MUST NOT contain any Received: or Resent- fields in their headers. See Section 6 or Section 8 for how to handle messages that do not conform to these requirements.

The MUA transmits the message to the MSA with the recipients specified in the To:, CC:, and Bcc: fields of the message header, using the syntax described in [RFC2822]. When the message is subsequently transmitted onward by the MSA via SMTP, each addr-spec in these fields is specified in a RCPT command generated by the MSA.

If the message header contains a Bcc: field, the MSA MUST strip it using one of the methods described in [RFC2822] section 3.6.3. That is, MSA MUST delete the Bcc: field or replace it with an empty Bcc: field, before transmitting the message onward.

The MSA SHOULD also perform the header fix-ups specified in [RFC2476] sections 8.1, 8.2, and 8.3, that is, the addition or correction of the Sender:, Date:, and Message-ID: fields. (This is a stronger requirement than that specified by [RFC2476].) Note that the fixed-up Sender: field MUST be omitted if it would be the same as the From: field.

5.1. Examples

The following shows a message submitted with the recipients specified in the message header.

```
      S: 220 smtp.hermes.example.edu ESMTP
      u: EHLO fanf2.botolph.example.edu
      S: 250-smtp.hermes.example.edu ESMTP
      S: 250-RCPTHDR
      S: 250 HELP
      u: MAIL FROM:<fanf2@example.edu> RCPTHDR
      S: 250 OK
      u: DATA
      S: 354 Go ahead
      u: To: postmaster@example.com
      u: CC: mail-support@ucs.example.edu
      u: Subject: Re: problems with email from example.com
      u:
      u: Thanks for investigating that problem for us!
      u:
      u: --
      u: mail-support@ucs.example.edu
      u: .
      S: 250 OK id=d41d8cd98f00b204
      u: QUIT
      S: 221 Good bye
```

The following shows the same message as transmitted onward by the MSA, including fixes described above and in Section 8.6.

```
      T: 220 ppsw.example.edu ESMTP
      s: EHLO smtp.hermes.example.edu
      T: 250-ppsw.example.edu ESMTP
      T: 250 HELP
      s: MAIL FROM:<fanf2@example.edu>
      T: 250 OK
      s: RCPT TO:<postmaster@example.com>
      T: 250 OK
      s: RCPT TO:<mail-support@ucs.example.edu>
      T: 250 OK
      s: DATA
      T: 354 Go ahead
      s: Received: from fanf2.botolph.example.edu ([192.0.2.48])
      s:     by smtp.hermes.example.edu with esmtp
      s:     id d41d8cd98f00b204; Tue, 14 Feb 2006 12:34:56 -0000
      s: Message-ID: <d41d8cd98f00b204@smtp.hermes.example.edu>
      s: Date: Tue, 14 Feb 2006 12:34:56 -0000
      s: From: Tony Finch <fanf2@example.edu>
      s: To: postmaster@example.com
      s: CC: mail-support@ucs.example.edu
      s: Subject: Re: problems with email from example.com
      s:
      s: Thanks for investigating that problem for us!
      s:
      s: --
      s: mail-support@ucs.example.edu
      s: .
      T: 250 OK id=e9800998ecf8427e
      s: QUIT
      T: 221 Good bye
```

The following is an example of a message submitted with a Bcc: field.

--something--

The following shows the same message as transmitted onward by the MSA.

--something--

6. Handling messages re-sent according to RFC 2822

[RFC2822] allows a message to be re-sent multiple times. Each time a message is re-sent, a contiguous set of Resent- fields is added to the top of the message header, such that when the message reaches its new destination, the set of Resent- fields appears between the Received: field corresponding to the initial hop of the most recent transmission and the Received: field corresponding to the final hop of the previous transmission.

A message conforming to [RFC2822] MUST NOT have any Resent- fields below the bottom Received: field in the message header. When submitting a message to be re-sent, there MUST be a set of Resent- fields at the very top of the message header, above one or more Received: fields. This is the "most recent" set of Resent- fields. This set SHOULD contain at most one of each kind of Resent- field. See Section 7 or Section 8 for how to handle messages that do not conform to these requirements.

The MUA transmits a re-sent message to the MSA with the recipients specified in the most recent Resent-To:, Resent-CC:, and Resent-Bcc: fields in the message header, using the syntax described in [RFC2822]. When the message is transmitted onward via SMTP, each addr-spec in these fields is specified in a RCPT command generated by the MSA. The MSA MUST NOT extract recipients from the To:, CC:, or Bcc: header fields, or from any Resent- fields that are not contiguous with the most recent set.

If the message header contains a Resent-Bcc: field in the most recent set, the MSA MUST strip it in the same manner as described for the Bcc: field in Section 5.

The MSA SHOULD also perform header fix-ups analogous to those specified in [RFC2476] sections 8.1, 8.2, and 8.3, that is, the MSA SHOULD add or replace the Resent-Sender:, Resent-Date:, and Resent-Message-ID: fields in the most recent set of Resent- fields, according to the same logic it uses for the Sender:, Date:, and Message-ID: fields in new messages. These fields MUST be added contiguous with the rest of the set, that is below the Received: field added by the MSA. (Note that [RFC2476] does not mention Resent- fields at all.)

6.1. Examples

The following is an example of a message being re-sent in conformance with [RFC2822].

--something--

The following shows the same message as transmitted onward by the MSA.

--something--

7. Handling messages re-sent according to RFC 822

[RFC0822] only allows for one set of Resent- fields in the message header, thus its semantics are strictly weaker than those of [RFC2822]. MUAs conforming to this SMTP service extension MUST re-send messages as specified in Section 6. This section is included for completeness, and describes how MSAs can support old clients re-sending messages via old simplified message submission interfaces.

A message conforming to this section’s requirements MUST have all its Resent- fields below the bottom Received: field in the message header. There MUST be at most one occurrence of each kind of Resent- field, otherwise the interpretation is ambiguous. See Section 8 for how to handle messages that do not conform to these requirements.

When an MUA submits a message with the recipients specified in the Resent-To:, Resent-CC:, and Resent-Bcc: fields as described in the previous paragraph, the MSA MUST move all the Resent- fields to the top of the message header, leaving the relative order of the other fields unchanged. The MSA MUST then proceed as specified in Section 6.

7.1. Examples

The following is an example of a message being re-sent in the old manner.

-- something involving sendmail -t --

The following shows the same message as transmitted onward by the MSA.

--something--

8. Handling badly-formed messages

This section provides some recommendations for how to handle messages which are not well-formed according to the requirements in the previous sections. Note that [RFC2476] already specifies how to handle syntax errors in email addresses; this section is therefore concerned with larger header syntax problems.

8.1. A new message containing Received: fields

A new message might already contain one or more Received: fields owing to its submission arrangements having more than one stage. For example, a simple Unix mail reader might submit messages via a sendmail-compatible program on the local machine, which adds a Received: field before acting as a message submission client.

An MSA MAY accept messages containing Received: fields but no Resent- fields. If so it MUST ignore the Received: fields and, apart from that difference, handle the message as described in Section 5. (Note that "ignore" means no special handling; in particular the MSA MUST NOT strip, re-order, or rename the fields.) The MSA SHOULD have a low limit (such as one or two) on the number of Received: fields that it will accept in these circumstances, in order to protect against loops caused by inappropriate re-submission of messages.

8.2. A message whose Resent- fields are not at the top

For the same reason as described in the previous section, a re-sent message that otherwise conforms to Section 6 might have Received: fields above the topmost set of Resent- fields from which the recipients are to be extracted.

An MSA MAY accept a message containing Received: fields and Resent- fields, where all the Resent- fields are above the bottom Received: field, but which has no Resent- fields at the very top. If so it MUST identify the most recent set of Resent- fields as the topmost contiguous set of Resent- fields, and apart from that difference, handle the message as described in Section 6. (Note that any altered or added Resent- fields MUST still be contiguous with the most recent set, and therefore below the fields that are above the topmost set, as well as below the Received: field added by the MSA.) The MSA SHOULD have a low limit (such as one or two) on the number of Received: fields above the topmost Resent- field that it will accept in these circumstances, in order to protect against loops caused by inappropriate re-submission of messages.

8.3. A re-sent message containing no Received: fields

Some MUAs conforming to [RFC0822] rename Received: fields when re-sending a message, such that the message contains no Received: fields at submission time. An MSA MAY accept messages containing Resent- fields but no Received: fields. If so it MUST handle the message as described in Section 7. It SHOULD also have a low limit (such as 50) on the number of header fields that it will accept in these circumstances, as a replacement for detecting loops by counting Received: fields.

8.4. A Resent- set with more than one of the same field

In this situation the header is ambiguous, and the problem might imply it has been re-ordered at some point. If it does not affect the most recent set of Resent- fields, the MSA MAY ignore the problem. If it does affect the most recent set of Resent- fields, then the MSA SHOULD reject the message; the user can instead forward the message as a message/rfc822 attachment, as described in [RFC2046].

8.5. Resent- sets above and below the bottom Received: field

In this situation the header is ambiguous, and the problem might imply it has been re-ordered at some point. If there is a set of Resent- fields very near the top of the message, the MSA MAY ignore any Resent- fields below the bottom Received: field, and apart from that difference, handle the message as described in Section 6. (Note that "ignore" means no special handling; in particular the MSA MUST NOT strip, re-order, or rename the fields.) For safety, the MSA SHOULD also check the other Resent- sets for plausibility, such as containing at most one of each kind of Resent- field.

In this case MSA SHOULD reject the message without trying to fix it up; the user can instead forward the message as a message/rfc822 attachment, as described in [RFC2046].

8.6. Unusual From: fields

When fixing up a message as described in Section 5, the MSA MAY add a From: field if it is missing, instead of rejecting the message for a syntax error. If it does, it SHOULD use the same field body as it would when fixing the Sender: field.

The MSA MAY also accept a message that has no Sender: field and multiple authors in the From: field, instead of rejecting it as a syntax error. The fix-ups will correct the problem.

The MSA MAY act correspondingly when fixing up a message as described in Section 6, except performing the fix-ups on the most recent set of Resent- fields instead of the plain fields.

This is an extension to [RFC2476] to further reduce the redundancy between the submitted message’s envelope and header.

9. Security considerations

[RFC2822] section 5 includes a discussion of the privacy implications of the Bcc: field, which are also relevant to this specification.

[RFC2822] has an extremely flexible syntax for specifying email addresses in message headers. When this is combined with MUAs that omit the addr-spec and show only the display-name, users can be confused about who a message was sent by or who it will be sent to. This is even more of a problem if the MSA fixes up abbreviated addresses "intelligently", such as qualifying unqualified addresses. Consider the difference between a message "To: Sarah Jones <jones@example.com>" and a message "To: Sarah, Jones", both of which might be displayed almost the same, but the second might be interpreted as having the extra recipient "sarah@example.com". Care should be taken when choosing policies for what to fix-up and what to reject for being badly formed.

MUAs sometimes do not handle the more obscure parts of [RFC2822] correctly, or do not handle syntax errors gracefully. Most MSA services are implemented using special configurations of MTA software, which has a generally better reputation in this area. Therefore adoption of this specification may improve the robustness of email systems.

Packet traces of SMTP conversations can reveal the number of recipients in a message, even if TLS is used: an eavesdropper can count the number of short command/reply exchanges before the large message data transmission. Even with SMTP pipelining [RFC2920], the size of the packet containing the message envelope gives significant information. This specification makes it much harder for an eavesdropper to make this kind of inference, since message sizes are mostly independent of the number of recipients.

Experience shows that a significant proportion of damaging email loops are caused by software that obtains messages after final delivery (e.g. via POP3 [RFC1939]) and re−injects them to an MTA after attempting to reconstruct their envelopes from their headers. This software exists to work around the lack of [RFC0974] connectivity for the MTA in question: it performs automated relaying, rather than MUA−style manual re−sending. Therefore this specification includes a number of defences against loops, in particular restricting its scope to message submission, and applying stricter limits than usual to messages with dubious formatting.

10. IANA Considerations

This document introduces no new namespaces to be managed by IANA.

This document defines a new SMTP service extension in Section 3.

11. References

11.1. Normative references

[RFC0822] Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

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

[RFC2476] Gellens, R. and J. Klensin, "Message Submission", RFC 2476, December 1998.

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[RFC2822] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

11.2. Informative references

[RFC0974] Partridge, C., "Mail routing and the domain system", RFC 974, January 1986.

[RFC1939] Myers, J. and M. Rose, "Post Office Protocol − Version 3", STD 53, RFC 1939, May 1996.

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

[RFC2554] Myers, J., "SMTP Service Extension for Authentication", RFC 2554, March 1999.

[RFC2920] Freed, N., "SMTP Service Extension for Command Pipelining", STD 60, RFC 2920, September 2000.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL − VERSION 4rev1", RFC 3501, March 2003.

Author’s Address

Tony Finch
University of Cambridge Computing Service
New Museums Site
Pembroke Street
Cambridge CB2 3QH
ENGLAND

Phone: +44 797 040 1426
Email: dot@dotat.at
URI: http://dotat.at/