Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: February 24, 2013

J. Kundrat
August 25, 2012

IMAP SUBMIT Extension
draft-kundrat-imap-submit-00

Abstract

This document extends the IMAP protocol with a feature to submit e-mail messages for delivery. It is intended to serve as a better alternative to the URLAUTH/BURL approach.

Table of Contents

1. Introduction

In the traditional IMAP/ESMTP service model, a MUA transfers each outgoing message twice -- once for storing it in the user's "sent mail" folder, and the second time for actual message submission over (E)SMTP. Under certain circumstances, such as when the message contains data which are already available in another message stored on the same IMAP server (such as when forwarding an unread attachment to another recipient), the MUA has to download the data before the message can be composed, resulting in transmitting the data three times in total.

Many proposals exist which aim to reduce this high number of transfers to the lowest possible number. The CATENATE extension [RFC4469] enables IMAP clients to have the IMAP servers compose messages on their behalf, optionally using data already available on the IMAP server. Using CATENATE, MUAs do not have to download individual message parts before including them to the newly created message.

The LEMONADE extension family [RFC5550] mandates full support for BURL [RFC4468] and URLAUTH [RFC4467] extensions. When coupled with a

properly configured pair of ESMTP and IMAP servers, these two extensions allow MUAs to have the submission server obtain the message payload from the IMAP server. This approach completely eliminates the need to upload the message data to the ESMTP server, achieving the "forward-without-download" goal.

The BURL/URLAUTH extensions, however, put a significant burden on the server operators who suddenly have to establish an explicit trust relation between their submission and IMAP servers, and make this trust path visible to the users' MUAs. No MUA-visible means of discovering this trust relation are defined. Furthermore, the whole scheme still requires the MUAs to maintain two distinct connections speaking different protocols. Users are prompted for two sets of credentials to authenticate to each of these two services. Real-world support issues were reported where users are able to access their IMAP service while access to the submission service is blocked by a mis-configured firewall.

The SUBMIT extension of the IMAP protocol effectively moves the message submission process to be initiated by a user's request to their IMAP server. When deployed, this scenario saves the overhead of establishing and securing a separate TCP connection against the submission server, reduces the amount of the configuration data the users are required to provide, and changes the trust paths which are required to exist between the submission and the IMAP servers. When combined with the existing CATENATE extension [RFC4469], the SUBMIT command works at least as effectively as the Lemonade trio of CATENATE/BURL/URLAUTH.

1.1. Requirements Language

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].

2. Mode of Operation

The SUBMIT extension adds the UID SUBMIT IMAP command which instructs the IMAP server to arrange for delivery of an already existing IMAP message. How this message is composed is outside of scope of this extension, but it is assumed that clients will often use the APPEND or APPEND ... CATENATE commands.

Upon receiving the SUBMIT command, the IMAP server is asked for arranging the initial message submission. Clients MAY pass additional data in form of various options of the SUBMIT command. The server checks the passed data and submission options, optionally performs sanity checks on the message contents, verifies against a local policy whether the user is authorized for message submission, and if none of these checks fail, the server passes the message for subsequent delivery. The delivery method is outside of scope of this document, but typical methods would be invoking a `sendmail`-compatible binary or passing the message to an ESMTP gateway.

3. IMAP Protocol Changes

This extension introduces one new IMAP command, a few related

response codes and a new family of the IMAP capabilities.

3.1. New IMAP Capabilities

3.1.1. The SUBMIT Capability

Servers implementing this extension announce its presence through the SUBMIT capability. If the server supports this extension but message submission is unconditionally disabled by a security policy or service configuration, this capability MUST NOT be announced.

3.1.2. The SUBMIT= Capabilities Family

The SUBMIT commands issued by the IMAP clients MAY contain submission options. Servers supporting voluntary features MUST indicate so by including the appropriate strings in the CAPABILITY responses. All capabilities used for these purposes begin with the SUBMIT= prefix.

3.1.2.1. SUBMIT=DSN

If the server supports user control of generating the Delivery Status Notifications (DSN), it MUST announce the SUBMIT=DSN capability. Clients MUST NOT attempt to control DSN options through the DSN submission option unless the server announces the SUBMIT=DSN capability.

3.2. Additional Response Codes

The following response codes are defined for communicating the reason why submission failed in a machine-readable way.

3.2.1. The POLICYDENIED Response Code

The POLICYDENIED response code SHOULD be used if the server rejects message submission as a result of a policy based decision which MAY take the message content, user's behavior and transaction history into account.

3.2.2. The SUBMISSIONRACE Response Code

The SUBMISSIONRACE response code MUST be sent in the tagged response if the client asks for submission of a message that is either not marked with the $SubmitPending keyword or marked with the $Submitted keyword.

3.3. UID SUBMIT command

The UID SUBMIT command submits a message for delivery.

Arguments:

- UID of message to be sent
- optional list of submission options

Responses: FETCH response with updated message flags

Result:

OK Message submitted for delivery

NO Submission failed

BAD Invalid commands or options

This command is only valid in the selected state.

The server MUST check its local policy configuration and verify that the authenticated user is allowed to submit messages. The decision MAY be based on the user's credentials, the message contents, past history of the user, or any other criteria the server deems relevant. The server SHOULD take into account any other local policies before it proceeds with further submission.

Clients MUST NOT submit a message which is either not marked with the $SubmitPending keyword from [RFC5550], or which is marked with the $Submitted keyword. Servers MUST reject such a command with a tagged NO bearing the SUBMISSIONRACE response code.

In the course of submission, servers MUST atomically add the $Submitted flag to the message, as described in LEMONADE [RFC5550]. This transition MAY be hidden from any IMAP session or it MAY be visible in all of them.

If the command succeeded, the message MUST be marked with the $Submitted keyword, the $SubmitPending keyword MUST be cleared and a FETCH response containing the message UID and its new flags MUST be sent.

If the command fails, the server MUST clear both the $Submitted or $SubmitPending keywords.

If the server supports CONDSTORE [RFC4551] or QRESYNC [RFC5172] extensions, any flag changes MUST obey the usual MODSEQ invariants. Any changes in the MODSEQ values MUST be communicated to the clients, as mandated by the relevant extensions.

Clients MUST be prepared to handle failing submission at any time. Servers MAY reject message submission for any reason.

The server MUST process all specified submission options. The server MUST respond with a tagged BAD if the client used unrecognized or unannounced submission option, or if a recognized submission option is used in an invalid way. If the server cannot honor a recognized and announced submission option, it MUST respond with a tagged NO with the POLICYDENIED response code and the message MUST NOT be submitted.

Servers MAY alter the message payload of the outgoing message in conformance with best current practice about Internet mail. Individual recipients MAY receive different versions of the message. In particular, servers MUST change message headers so that the identity of addresses present in the Bcc headers is not revealed to other recipients. This mode of operation is described in [RFC5321] and [RFC5322]. The copy stored on the IMAP server MUST NOT be altered by these modifications.

3.3.1. Submission options

The following submission options are defined by this extension:

3.3.1.1. FROM Submission Option

Syntax: one e-mail address

The FROM submission option corresponds to the FROM field of the SMTP envelope. If not present, its value MUST be inferred from the message payload.

It is an error if the FROM submission option is present more than once. Servers MUST reject such commands using the BAD tagged response and the message MUST NOT be submitted. Message flags of the source message MUST NOT be modified.

3.3.1.2. RECIPIENT Submission Option

Syntax: one e-mail address

The RECIPIENT submission option corresponds to the RCPT TO field of the SMTP envelope.

The RECIPIENT submission option MAY be present more than once. Servers MAY impose a limit on the number of recipients of a single message.

If the RECIPIENT submission option is present, servers MUST ignore any To, Cc and Bcc headers in the message payload when determining the list of recipients of this message. That is, the final list of recipients of the message MUST consist exactly of those recipients

specified in the RECIPIENT submission options. The server MUST still sanitize the headers of the submitted message to guarantee the privacy of the recipients listed in the Bcc message header.

If the RECIPIENT submission option is missing, servers MUST infer its value from the message payload. For example, each address present in any of To, Cc and Bcc message headers MUST be present in the recipient list.

Servers MAY impose a local policy decision about always sending a copy of message to a certain address. This operation MUST NOT affect the user-specified list of recipients passed through the RECIPIENTS submission options.

Message submission MUST be atomic -- message is either submitted for delivery to all recipients, or it MUST NOT be submitted for delivery to anyone. Actual delivery MAY still fail for certain recipients.

3.3.1.3. DSN Submission Option

Syntax: delivery status notice specification

The DSN options are specified per-recipient, not per-message, and therefore their syntax is different from the other submission options.

The DSN submission option controls generating of delivery status notifications related to the currently submitted message. When not given, an implementation-defined default value MUST be used. The default value MUST be either (FAILURE) or (DELAY, FAILURE), as mandated by [RFC3461].

It is an error if the DSN submission option is present multiple times for one recipient.

Clients MUST NOT specify the DSN submission option unless the server announces the SUBMIT=DSN capability. Support for the SUBMIT=DSN submission option is OPTIONAL.

The DSN specification is either "NONE" to deactivate DSNs altogether, or a parenthesized list of any of the following options:

SUCCESS requests generating DSNs upon successful delivery of a message

DELAY activates generating DSNs when delivery is delayed

FAILURE requests generating DSNs when the delivery fails

The order of DSN requests is not significant.

3.3.1.4. DSN-RET Submission Option

Syntax: DSN return option specification

This per-message submission option corresponds to the RET=... parameter from [RFC3461]. Two values are defined, "HDRS" and "FULL", meaning to include only the headers or the full message, respectively, in the generated delivery status reports.

Clients MUST NOT use the DSN-ENVID return option unless the server announces the SUBMIT=DSN capability.

3.3.1.5. DSN-ENVID Submission Option

Syntax: specification of ESMTP Envelope ID

This per-message submission option corresponds to the ENVID=... parameter from [RFC3461]. It allows senders to attach a machine-readable ID to be received in the delivery status reports concerning this message.

Clients MUST NOT use the DSN-ENVID return option unless the server announces the SUBMIT=DSN capability or a SUBMIT=... capability defined by future extensions which make use of the ENVID ESMTP parameter.

4. Example

This section contains an example of how message submission over IMAP works.

The following example shows how client should submit a message with UID 123 in the current mailbox for delivery. If the message is passed through SMTP, its From address in the SMTP envelope MUST be set to foo@example.org and it MUST be submitted for delivery to two recipients, the a@example.org and b@example.org. The DSN options are set to report about excess delays and failures in message delivery for the first recipient. System's default policy of DSN production is retained for the second recipient.

```
C: x UID SUBMIT 123 (FROM "foo@example.org"
        FROM "bar@example.org"
        RECIPIENT "a@example.org" DSN (delay failure)
        RECIPIENT "b@example.org"
    )
S: * 10 FETCH (UID 123 FLAGS ($Submitted))
S: x OK Message passed to the sendmail binary
```

In the following example, a message is delivered to addresses specified in the message payload. No submission options are given, and therefore the From and To envelope items are inferred from the actual payload. The DSN options, if supported, are set to an implementation-defined default value.

```
C: x UID SUBMIT 123
```

```
S: * 10 FETCH (UID 123 FLAGS ($Submitted))
S: x OK Message passed to the sendmail binary
```

5. Acknowledgements

FIXME

6. IANA Considerations

IMAP4 capabilities are registered by publishing a standards track or IESG approved experimental RFC. The registry is currently located at:

http://www.iana.org/assignments/imap4-capabilities

This document defines the following list of IMAP capabilities. IANA will be asked to add them to the registry:

- SUBMIT
- SUBMIT=DSN

FIXME: response codes

7. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [RFC5234].

Non-terminals referenced but not defined below are as defined by [RFC3501], or [RFC5234].

```
   capability          =/ "SUBMIT" / "SUBMIT=DSN"
                        ;; This extension also reserves all further
                        ;; capabilities starting with the "SUBMIT="
                        ;; prefix for future extensions related to the
                        ;; message submission over IMAP

   uid                 =/ "UID" SP sendmail

   sendmail            = "SUBMIT" SP uniqueid [SP submission-options]

   submission-options  = "(" submission-option *( SP submission-option ) ")"

   submission-option   = sub-option-from / sub-option-recipient
                        / sub-option-dsn-ret / sub-option-dsn-envid

   sub-option-from     = "FROM" SP one-email-addr
                        ;; MUST NOT be present more than once

   sub-option-recipient= "RECIPIENT" SP one-email-addr [SP sub-option-rcpt-dsn]
                        ;; MAY be present more than once

   sub-option-rcpt-dsn = "DSN" SP ( "NONE" / dsn-spec )
                        ;; MUST NOT be present more than once

   dsn-spec            = "(" dsn-spec-item *( SP dsn-spec-item ) ")"
                        ;; an individual dsn-spec-item MUST NOT
                        ;; be present more than once

   dsn-spec-item       = "DELAY" / "FAILURE" / "SUCCESS"

   sub-option-dsn-ret  = "DSN-RET" SP ( "FULL" / "HDRS" )

   sub-option-dsn-envid= "DSN-ENVID" SP xtext
                        ;; <xtext> is defined in [RFC3461], section 4.
                        ;; The allowed syntax is further limited by
                        ;; its section 4.4.

   one-email-addr      = string
                        ;; valid e-mail address as per [RFC5321]
```

8. Security Considerations

This extension introduces a new way of allowing authenticated users to submit Internet mail. Servers supporting this extension SHOULD implement the same security measures as other SUBMISSION [RFC4409] servers open to users.

The redirect command from SIEVE [RFC5228] already requires some types of IMAP message stores to be able to generate outgoing mail.

Security considerations for this extension are similar.

For the IMAP-based submission to work, the server operators MUST configure their MTA systems to accept submission requests from their IMAP servers. This change MAY have security implications, even though services supporting the SIEVE filtering are already configured to accept e-mails for submission.

The new trust path MAY replace the trust path required for the BURL/URLAUTH operation required by the LEMONADE extension family.

9. References

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

[RFC3461] Moore, K., "Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs)", RFC 3461, January 2003.

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

[RFC4409] Gellens, R. and J. Klensin, "Message Submission for Mail", RFC 4409, April 2006.

[RFC4467] Crispin, M., "Internet Message Access Protocol (IMAP) - URLAUTH Extension", RFC 4467, May 2006.

[RFC4468] Newman, C., "Message Submission BURL Extension", RFC 4468, May 2006.

[RFC4469] Resnick, P., "Internet Message Access Protocol (IMAP) CATENATE Extension", RFC 4469, April 2006.

[RFC4551] Melnikov, A. and S. Hole, "IMAP Extension for Conditional STORE Operation or Quick Flag Changes Resynchronization", RFC 4551, June 2006.

[RFC5172] Varada, S., "Negotiation for IPv6 Datagram Compression Using IPv6 Control Protocol", RFC 5172, March 2008.

[RFC5228] Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language", RFC 5228, January 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

[RFC5550] Cridland, D., Melnikov, A. and S. Maes, "The Internet Email to Support Diverse Service Environments (Lemonade) Profile", RFC 5550, August 2009.

Appendix A. FIXME Items

What's got higher priority, SUBMISSIONRACE or POLICYDENIED? :)

IANA and the response codes

"if the command fails, server MUST clear both $SubmitPending and $Submitted" -- what to do when there's something like a disk error?

Appendix B. Changelog

Appendix B.1. Changes in -00 since private-01

- Renamed to SUBMIT
- DSNs are per-recipient, not per-message
- The introduction was rewritten
- Miscellaneous clarifications
- Changed DSN NIL to DSN NONE
- Clarified the semantics of the RECIPIENT submission option to guarantee Bcc privacy
- Editorial tweaks, including changes to the required SHOULD/MUST/ ...
- DSN's ENVID and RET

Appendix B.2. Changes in private-01 since private-00

- Removed the superfluous SENDER submission option
- Mandating Bcc removal in conformance with RFC 5321 / RFC 5322

Author's Address

Jan Kundrat
Eledrova 558
Prague 181 00
CZ

Email: jkt@flaska.net