                     Local Mail Transfer Protocol

Status of this Memo

   This memo provides information for the Internet community. This memo
   does not specify an Internet standard of any kind. Distribution of
   this memo is unlimited.

1. Abstract

   SMTP [SMTP] [HOST-REQ] and its service extensions [ESMTP] provide a
   mechanism for transferring mail reliably and efficiently. The design
   of the SMTP protocol effectively requires the server to manage a mail
   delivery queue.

   In some limited circumstances, outside the area of mail exchange
   between independent hosts on public networks, it is desirable to
   implement a system where a mail receiver does not manage a queue.
   This document describes the LMTP protocol for transporting mail into
   such systems.

   Although LMTP is an alternative protocol to ESMTP, it uses (with a
   few changes) the syntax and semantics of ESMTP. This design permits
   LMTP to utilize the extensions defined for ESMTP. LMTP should be
   used only by specific prior arrangement and configuration, and it
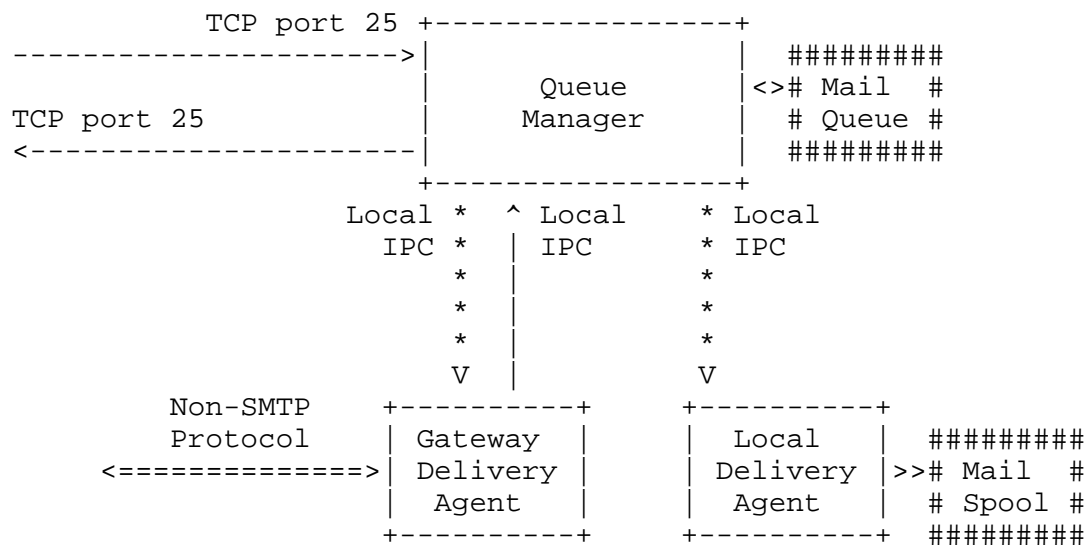   MUST NOT be used on TCP port 25.

Table of Contents

2. Conventions Used in this Document

   In examples, "C:" and "S:" indicate lines sent by the client and
   server respectively.

3. Introduction and Overview

   The design of the SMTP protocol effectively requires the server to
   manage a mail delivery queue. This is because a single mail
   transaction may specify multiple recipients and the final "." of the
   DATA command may return only one reply code, to indicate the status
   of the entire transaction. If, for example, a server is given a
   transaction for two recipients, delivery to the first succeeds, and
   delivery to the second encounters a temporary failure condition,
   there is no mechanism to inform the client of the situation. The
   server must queue the message and later attempt to deliver it to the
   second recipient.

   This queuing requirement is beneficial in the situation for which
   SMTP was originally designed: store-and-forward relay of mail between
   networked hosts. In some limited situations, it is desirable to have
   a server which does not manage a queue, instead relying on the client
   to perform queue management. As an example, consider a hypothetical
   host with a mail system designed as follows:

```
            TCP port 25 +-----------------+
       ---------------------->|                 |  #########
                      |       |     Queue       |<>#  Mail #
       TCP port 25    |       |    Manager      |  # Queue #
       <---------------------|                 |  #########
                      +-----------------+
                 Local *   ^ Local    * Local
                   IPC *   | IPC      * IPC
                       *   |          *
                       *   |          *
                       *   |          *
                       V   |          V
       Non-SMTP    +---------+     +---------+
       Protocol    | Gateway |     |  Local  |  #########
       <=============>| Delivery |     | Delivery |>>#  Mail #
                   |  Agent  |     |  Agent  |  # Spool #
                   +---------+     +---------+  #########
```

   The host's mail system has three independent, communicating
   subsystems. The first is a queue manager, which acts as a

traditional SMTP agent, transferring messages to and from other hosts
over TCP and managing a mail queue in persistent storage. The other
two are agents which handle delivery for addresses in domains for
which the host takes responsibility. One agent performs gatewaying
to and from some other mail system. The other agent delivers the
message into a persistent mail spool.

It would be desirable to use SMTP over a local inter-process
communication channel to transfer messages from the queue manager to
the delivery agents. It would, however, significantly increase the
complexity of the delivery agents to require them to manage their own
mail queues.

The common practice of invoking a delivery agent with the envelope
address(es) as command-line arguments, then having the delivery agent
communicate status with an exit code has three serious problems: the
agent can only return one exit code to be applied to all recipients,
it is difficult to extend the interface to deal with ESMTP extensions
such as DSN [DSN] and ENHANCEDSTATUSCODES [ENHANCEDSTATUSCODES], and
exits performed by system libraries due to temporary conditions
frequently get interpreted as permanent errors.

The LMTP protocol causes the server to return, after the final "." of
the DATA command, one reply for each recipient. Therefore, if the
queue manager is configured to use LMTP instead of SMTP when
transferring messages to the delivery agents, then the delivery
agents may attempt delivery to each recipient after the final "." and
individually report the status for each recipient. Connections which
should use the LMTP protocol are drawn in the diagram above using
asterisks.

Note that it is not beneficial to use the LMTP protocol when
transferring messages to the queue manager, either from the network
or from a delivery agent. The queue manager does implement a mail
queue, so it may store the message and take responsibility for later
delivering it.

4. The LMTP protocol

The LMTP protocol is identical to the SMTP protocol SMTP [SMTP]
[HOST-REQ] with its service extensions [ESMTP], except as modified by
this document.

A "successful" RCPT command is defined as an RCPT command which
returns a Positive Completion reply code.

A "Positive Completion reply code" is defined in Appendix E of STD
10, RFC 821 [SMTP] as a reply code which "2" as the first digit.

4.1. The LHLO, HELO and EHLO commands

   The HELO and EHLO commands of ESMTP are replaced by the LHLO command.
   This permits a misconfiguration where both parties are not using the
   same protocol to be detected.

   The LHLO command has identical semantics to the EHLO command of ESMTP
   [ESMTP].

   The HELO and EHLO commands of ESMTP are not present in LMTP. A LMTP
   server MUST NOT return a Postive Completion reply code to these
   commands. The 500 reply code is recommended.

4.2. The DATA command

   In the LMTP protocol, there is one additional restriction placed on
   the DATA command, and one change to how replies to the final "." are
   sent.

   The additional restriction is that when there have been no successful
   RCPT commands in the mail transaction, the DATA command MUST fail
   with a 503 reply code.

   The change is that after the final ".", the server returns one reply
   for each previously successful RCPT command in the mail transaction,
   in the order that the RCPT commands were issued. Even if there were
   multiple successful RCPT commands giving the same forward-path, there
   must be one reply for each successful RCPT command.

   When one of these replies to the final "." is a Positive Completion
   reply, the server is accepting responsibility for delivering or
   relying the message to the corresponding recipient. It must take
   this responsibility seriously, i.e., it MUST NOT lose the message for
   frivolous reasons, e.g., because the host later crashes or because of
   a predictable resource shortage.

   A multiline reply is still considered a single reply and corresponds
   to a single RCPT command.

   EXAMPLE:

   S: 220 foo.edu LMTP server ready
   C: LHLO foo.edu
   S: 250-foo.edu
   S: 250-PIPELINING
   S: 250 SIZE
   C: MAIL FROM:<chris@bar.com>
   S: 250 OK

```
C: RCPT TO:<pat@foo.edu>
S: 250 OK
C: RCPT TO:<jones@foo.edu>
S: 550 No such user here
C: RCPT TO:<green@foo.edu>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: Blah blah blah...
C: ...etc. etc. etc.
C: .
S: 250 OK
S: 452 <green@foo.edu> is temporarily over quota
C: QUIT
S: 221 foo.edu closing connection
```

NOTE: in the above example, the domain names of both the client and
   server are identical. This is because in the example the client and
   server are different subsystems of the same mail domain.

4.3. The BDAT command

   If the server supports the ESMTP CHUNKING extension [BINARYMIME], a
   BDAT command containing the LAST parameter returns one reply for each
   previously successful RCPT command in the mail transaction, in the
   order that the RCPT commands were issued. Even if there were
   multiple successful RCPT commands giving the same forward-path, there
   must be one reply for each successful RCPT command. If there were no
   previously successful RCPT commands in the mail transaction, then the
   BDAT LAST command returns zero replies.

   When one of these replies to the BDAT LAST command is a Positive
   Completion reply, the server is accepting responsibility for
   delivering or relaying the message to the corresponding recipient.
   It must take this responsibility seriously, i.e., it MUST NOT lose
   the message for frivolous reasons, e.g., because the host later
   crashes or because of a predictable resource shortage.

   A multiline reply is still considered a single reply and corresponds
   to a single RCPT command.

   The behavior of BDAT commands without the LAST parameter is not
   changed; they still return exactly one reply.

5. Implementation requirements

   As LMTP is a different protocol than SMTP, it MUST NOT be used on the
   TCP service port 25.

   A server implementation MUST implement the PIPELINING [PIPELINING]
   and ENHANCEDSTATUSCODES [ENHANCEDSTATUSCODES] ESMTP extensions. A
   server implementation SHOULD implement the 8BITMIME [8BITMIME]
   extension.

   Use of LMTP can aggravate the situation described in [DUP-MSGS]. To
   avoid this synchronization problem, the following requirements are
   made of implementations:

   A server implementation which is capable of quickly accepting
   responsibility for delivering or relaying a message to multiple
   recipients and which is capable of sending any necessary notification
   messages SHOULD NOT implement the LMTP protocol.

   The LMTP protocol SHOULD NOT be used over wide area networks.

   The server SHOULD send each reply as soon as possible. If it is
   going to spend a nontrivial amount of time handling delivery for the
   next recipient, it SHOULD flush any outgoing LMTP buffer, so the
   reply may be quickly received by the client.

   The client SHOULD process the replies as they come in, instead of
   waiting for all of the replies to arrive before processing any of
   them. If the connection closes after replies for some, but not all,
   recipients have arrived, the client MUST process the replies that
   arrived and treat the rest as temporary failures.

6. Acknowledgments

   This work is a refinement of the MULT extension, which was invented
   by Jeff Michaud and was used in implementing gateways to the Mail-11
   mail system.

   Many thanks to Matt Thomas for assisting me in understanding the
   semantics of the Mail-11 MULT extension.

7. References

   [8BITMIME] Klensin, J., et. al, "SMTP Service Extension for 8bit-MIME
        transport", RFC 1652, July 1994.

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

   [DSN] Moore, K., Vaudreuil, G., "An Extensible Message Format for
        Delivery Status Notifications", RFC 1894, January 1996.

   [DUP-MSGS] Partridge, C., "Duplicate messages and SMTP", RFC 1047,
        February 1988.

   [ENHANCEDSTATUSCODES] Freed, N., "SMTP Service Extension for
        Returning Enhanced Error Codes", RFC 2034, October 1996.

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

   [HOST-REQ] Braden, R., "Requirements for Internet hosts - application
        and support", STD 3, RFC 1123 section 5, October 1989.

   [PIPELINING] Freed, N., Cargille, A, "SMTP Service Extension for
        Command Pipelining", RFC 1854, October 1995.

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

   There are no known security issues with the issues in this memo.

9. Author's Address

   John G. Myers
   Carnegie-Mellon University
   5000 Forbes Ave.
   Pittsburgh PA, 15213-3890

   EMail: jgm+@cmu.edu