Network Working Group
Request for Comments: 1939
STD: 53
Obsoletes: 1725
Category: Standards Track

J. Myers
Carnegie Mellon
M. Rose
Dover Beach Consulting, Inc.

May 1996

Post Office Protocol - Version 3

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.

Table of Contents

1. Introduction

On certain types of smaller nodes in the Internet it is often impractical to maintain a message transport system (MTS). For example, a workstation may not have sufficient resources (cycles, disk space) in order to permit a SMTP server [RFC821] and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive (or impossible) to keep a personal computer interconnected to an IP-style network for long amounts of time (the node is lacking the resource known as "connectivity").

Despite this, it is often very useful to be able to manage mail on these smaller nodes, and they often support a user agent (UA) to aid the tasks of mail handling. To solve this problem, a node which can support an MTS entity offers a maildrop service to these less endowed nodes. The Post Office Protocol - Version 3 (POP3) is intended to permit a workstation to dynamically access a maildrop on a server host in a useful fashion. Usually, this means that the POP3 protocol is used to allow a workstation to retrieve mail that the server is holding for it.

POP3 is not intended to provide extensive manipulation operations of mail on the server; normally, mail is downloaded and then deleted. A more advanced (and complex) protocol, IMAP4, is discussed in [RFC1730].

For the remainder of this memo, the term "client host" refers to a host making use of the POP3 service, while the term "server host" refers to a host which offers the POP3 service.

2. A Short Digression

This memo does not specify how a client host enters mail into the transport system, although a method consistent with the philosophy of this memo is presented here:

> When the user agent on a client host wishes to enter a message into the transport system, it establishes an SMTP connection to its relay host and sends all mail to it. This relay host could be, but need not be, the POP3 server host for the client host. Of course, the relay host must accept mail for delivery to arbitrary recipient addresses, that functionality is not required of all SMTP servers.

3. Basic Operation

Initially, the server host starts the POP3 service by listening on TCP port 110. When a client host wishes to make use of the service, it establishes a TCP connection with the server host. When the connection is established, the POP3 server sends a greeting. The client and POP3 server then exchange commands and responses (respectively) until the connection is closed or aborted.

Commands in the POP3 consist of a case-insensitive keyword, possibly followed by one or more arguments. All commands are terminated by a CRLF pair. Keywords and arguments consist of printable ASCII characters. Keywords and arguments are each separated by a single SPACE character. Keywords are three or four characters long. Each argument may be up to 40 characters long.

Responses in the POP3 consist of a status indicator and a keyword possibly followed by additional information. All responses are terminated by a CRLF pair. Responses may be up to 512 characters long, including the terminating CRLF. There are currently two status indicators: positive ("+OK") and negative ("-ERR"). Servers MUST send the "+OK" and "-ERR" in upper case.

Responses to certain commands are multi-line. In these cases, which are clearly indicated below, after sending the first line of the response and a CRLF, any additional lines are sent, each terminated by a CRLF pair. When all lines of the response have been sent, a final line is sent, consisting of a termination octet (decimal code 046, ".") and a CRLF pair. If any line of the multi-line response begins with the termination octet, the line is "byte-stuffed" by pre-pending the termination octet to that line of the response. Hence a multi-line response is terminated with the five octets "CRLF.CRLF". When examining a multi-line response, the client checks to see if the line begins with the termination octet. If so and if octets other than CRLF follow, the first octet of the line (the termination octet) is stripped away. If so and if CRLF immediately follows the termination character, then the response from the POP server is ended and the line containing ".CRLF" is not considered part of the multi-line response.

A POP3 session progresses through a number of states during its lifetime. Once the TCP connection has been opened and the POP3 server has sent the greeting, the session enters the AUTHORIZATION state. In this state, the client must identify itself to the POP3 server. Once the client has successfully done this, the server acquires resources associated with the client's maildrop, and the session enters the TRANSACTION state. In this state, the client requests actions on the part of the POP3 server. When the client has

issued the QUIT command, the session enters the UPDATE state. In this state, the POP3 server releases any resources acquired during the TRANSACTION state and says goodbye. The TCP connection is then closed.

A server MUST respond to an unrecognized, unimplemented, or syntactically invalid command by responding with a negative status indicator. A server MUST respond to a command issued when the session is in an incorrect state by responding with a negative status indicator. There is no general method for a client to distinguish between a server which does not implement an optional command and a server which is unwilling or unable to process the command.

A POP3 server MAY have an inactivity autologout timer. Such a timer MUST be of at least 10 minutes' duration. The receipt of any command from the client during that interval should suffice to reset the autologout timer. When the timer expires, the session does NOT enter the UPDATE state--the server should close the TCP connection without removing any messages or sending any response to the client.

4. The AUTHORIZATION State

Once the TCP connection has been opened by a POP3 client, the POP3 server issues a one line greeting. This can be any positive response. An example might be:

```
S:  +OK POP3 server ready
```

The POP3 session is now in the AUTHORIZATION state. The client must now identify and authenticate itself to the POP3 server. Two possible mechanisms for doing this are described in this document, the USER and PASS command combination and the APOP command. Both mechanisms are described later in this document. Additional authentication mechanisms are described in [RFC1734]. While there is no single authentication mechanism that is required of all POP3 servers, a POP3 server must of course support at least one authentication mechanism.

Once the POP3 server has determined through the use of any authentication command that the client should be given access to the appropriate maildrop, the POP3 server then acquires an exclusive-access lock on the maildrop, as necessary to prevent messages from being modified or removed before the session enters the UPDATE state. If the lock is successfully acquired, the POP3 server responds with a positive status indicator. The POP3 session now enters the TRANSACTION state, with no messages marked as deleted. If the maildrop cannot be opened for some reason (for example, a lock can not be acquired, the client is denied access to the appropriate

maildrop, or the maildrop cannot be parsed), the POP3 server responds with a negative status indicator. (If a lock was acquired but the POP3 server intends to respond with a negative status indicator, the POP3 server must release the lock prior to rejecting the command.) After returning a negative status indicator, the server may close the connection. If the server does not close the connection, the client may either issue a new authentication command and start again, or the client may issue the QUIT command.

After the POP3 server has opened the maildrop, it assigns a message-number to each message, and notes the size of each message in octets. The first message in the maildrop is assigned a message-number of "1", the second is assigned "2", and so on, so that the nth message in a maildrop is assigned a message-number of "n". In POP3 commands and responses, all message-numbers and message sizes are expressed in base-10 (i.e., decimal).

Here is the summary for the QUIT command when used in the AUTHORIZATION state:

```
   QUIT

      Arguments: none

      Restrictions: none

      Possible Responses:
          +OK

      Examples:
          C: QUIT
          S: +OK dewey POP3 server signing off
```

5. The TRANSACTION State

Once the client has successfully identified itself to the POP3 server and the POP3 server has locked and opened the appropriate maildrop, the POP3 session is now in the TRANSACTION state. The client may now issue any of the following POP3 commands repeatedly. After each command, the POP3 server issues a response. Eventually, the client issues the QUIT command and the POP3 session enters the UPDATE state.

Here are the POP3 commands valid in the TRANSACTION state:

STAT

Arguments: none

Restrictions:
may only be given in the TRANSACTION state

Discussion:
The POP3 server issues a positive response with a line containing information for the maildrop. This line is called a "drop listing" for that maildrop.

In order to simplify parsing, all POP3 servers are required to use a certain format for drop listings. The positive response consists of "+OK" followed by a single space, the number of messages in the maildrop, a single space, and the size of the maildrop in octets. This memo makes no requirement on what follows the maildrop size. Minimal implementations should just end that line of the response with a CRLF pair. More advanced implementations may include other information.

NOTE: This memo STRONGLY discourages implementations from supplying additional information in the drop listing. Other, optional, facilities are discussed later on which permit the client to parse the messages in the maildrop.

Note that messages marked as deleted are not counted in either total.

Possible Responses:
+OK nn mm

Examples:
C: STAT
S: +OK 2 320

LIST [msg]

Arguments:
a message-number (optional), which, if present, may NOT refer to a message marked as deleted

Restrictions:
may only be given in the TRANSACTION state

Discussion:
If an argument was given and the POP3 server issues a positive response with a line containing information for that message. This line is called a "scan listing" for that message.

If no argument was given and the POP3 server issues a positive response, then the response given is multi-line. After the initial +OK, for each message in the maildrop, the POP3 server responds with a line containing information for that message. This line is also called a "scan listing" for that message. If there are no messages in the maildrop, then the POP3 server responds with no scan listings--it issues a positive response followed by a line containing a termination octet and a CRLF pair.

In order to simplify parsing, all POP3 servers are required to use a certain format for scan listings. A scan listing consists of the message-number of the message, followed by a single space and the exact size of the message in octets. Methods for calculating the exact size of the message are described in the "Message Format" section below. This memo makes no requirement on what follows the message size in the scan listing. Minimal implementations should just end that line of the response with a CRLF pair. More advanced implementations may include other information, as parsed from the message.

> NOTE: This memo STRONGLY discourages implementations from supplying additional information in the scan listing. Other, optional, facilities are discussed later on which permit the client to parse the messages in the maildrop.

Note that messages marked as deleted are not listed.

Possible Responses:
+OK scan listing follows
-ERR no such message

Examples:
C: LIST
S: +OK 2 messages (320 octets)
S: 1 120

```
S: 2 200
S: .
  ...
C: LIST 2
S: +OK 2 200
  ...
C: LIST 3
S: -ERR no such message, only 2 messages in maildrop
```

RETR msg

Arguments:
a message-number (required) which may NOT refer to a message marked as deleted

Restrictions:
may only be given in the TRANSACTION state

Discussion:
If the POP3 server issues a positive response, then the response given is multi-line. After the initial +OK, the POP3 server sends the message corresponding to the given message-number, being careful to byte-stuff the termination character (as with all multi-line responses).

Possible Responses:
+OK message follows
-ERR no such message

Examples:

```
C: RETR 1
S: +OK 120 octets
S: <the POP3 server sends the entire message here>
S: .
```

DELE msg

Arguments:
a message-number (required) which may NOT refer to a message marked as deleted

Restrictions:
may only be given in the TRANSACTION state

Discussion:
The POP3 server marks the message as deleted. Any future reference to the message-number associated with the message in a POP3 command generates an error. The POP3 server does not actually delete the message until the POP3 session enters the UPDATE state.

Possible Responses:
+OK message deleted
-ERR no such message

Examples:
C: DELE 1
S: +OK message 1 deleted
...
C: DELE 2
S: -ERR message 2 already deleted

NOOP

Arguments: none

Restrictions:
may only be given in the TRANSACTION state

Discussion:
The POP3 server does nothing, it merely replies with a positive response.

Possible Responses:
+OK

Examples:
C: NOOP
S: +OK

RSET

Arguments: none

Restrictions:
may only be given in the TRANSACTION state

Discussion:
If any messages have been marked as deleted by the POP3 server, they are unmarked. The POP3 server then replies

with a positive response.

Possible Responses:
+OK

Examples:
C: RSET
S: +OK maildrop has 2 messages (320 octets)

6. The UPDATE State

When the client issues the QUIT command from the TRANSACTION state, the POP3 session enters the UPDATE state. (Note that if the client issues the QUIT command from the AUTHORIZATION state, the POP3 session terminates but does NOT enter the UPDATE state.)

If a session terminates for some reason other than a client-issued QUIT command, the POP3 session does NOT enter the UPDATE state and MUST not remove any messages from the maildrop.

QUIT

Arguments: none

Restrictions: none

Discussion:
The POP3 server removes all messages marked as deleted from the maildrop and replies as to the status of this operation. If there is an error, such as a resource shortage, encountered while removing messages, the maildrop may result in having some or none of the messages marked as deleted be removed. In no case may the server remove any messages not marked as deleted.

Whether the removal was successful or not, the server then releases any exclusive-access lock on the maildrop and closes the TCP connection.

Possible Responses:
+OK
-ERR some deleted messages not removed

Examples:
C: QUIT
S: +OK dewey POP3 server signing off (maildrop empty)
...
C: QUIT

```
S: +OK dewey POP3 server signing off (2 messages left)
   ...
```

7. Optional POP3 Commands

The POP3 commands discussed above must be supported by all minimal implementations of POP3 servers.

The optional POP3 commands described below permit a POP3 client greater freedom in message handling, while preserving a simple POP3 server implementation.

NOTE: This memo STRONGLY encourages implementations to support these commands in lieu of developing augmented drop and scan listings. In short, the philosophy of this memo is to put intelligence in the part of the POP3 client and not the POP3 server.

TOP msg n

Arguments:
a message-number (required) which may NOT refer to to a message marked as deleted, and a non-negative number of lines (required)

Restrictions:
may only be given in the TRANSACTION state

Discussion:
If the POP3 server issues a positive response, then the response given is multi-line. After the initial +OK, the POP3 server sends the headers of the message, the blank line separating the headers from the body, and then the number of lines of the indicated message's body, being careful to byte-stuff the termination character (as with all multi-line responses).

Note that if the number of lines requested by the POP3 client is greater than than the number of lines in the body, then the POP3 server sends the entire message.

Possible Responses:
+OK top of message follows
-ERR no such message

Examples:

```
C: TOP 1 10
S: +OK
```

```
S: <the POP3 server sends the headers of the
   message, a blank line, and the first 10 lines
   of the body of the message>
S: .
   ...
C: TOP 100 3
S: -ERR no such message
```

UIDL [msg]

Arguments:
a message-number (optional), which, if present, may NOT refer to a message marked as deleted

Restrictions:
may only be given in the TRANSACTION state.

Discussion:
If an argument was given and the POP3 server issues a positive response with a line containing information for that message. This line is called a "unique-id listing" for that message.

If no argument was given and the POP3 server issues a positive response, then the response given is multi-line. After the initial +OK, for each message in the maildrop, the POP3 server responds with a line containing information for that message. This line is called a "unique-id listing" for that message.

In order to simplify parsing, all POP3 servers are required to use a certain format for unique-id listings. A unique-id listing consists of the message-number of the message, followed by a single space and the unique-id of the message. No information follows the unique-id in the unique-id listing.

The unique-id of a message is an arbitrary server-determined string, consisting of one to 70 characters in the range 0x21 to 0x7E, which uniquely identifies a message within a maildrop and which persists across sessions. This persistence is required even if a session ends without entering the UPDATE state. The server should never reuse an unique-id in a given maildrop, for as long as the entity using the unique-id exists.

Note that messages marked as deleted are not listed.

While it is generally preferable for server implementations to store arbitrarily assigned unique-ids in the maildrop,

this specification is intended to permit unique-ids to be calculated as a hash of the message. Clients should be able to handle a situation where two identical copies of a message in a maildrop have the same unique-id.

Possible Responses:
+OK unique-id listing follows
-ERR no such message

Examples:
```
C: UIDL
S: +OK
S: 1 whqtswO00WBw418f9t5JxYwZ
S: 2 QhdPYR:00WBw1Ph7x7
S: .
   ...
C: UIDL 2
S: +OK 2 QhdPYR:00WBw1Ph7x7
   ...
C: UIDL 3
S: -ERR no such message, only 2 messages in maildrop
```

USER name

Arguments:
a string identifying a mailbox (required), which is of significance ONLY to the server

Restrictions:
may only be given in the AUTHORIZATION state after the POP3 greeting or after an unsuccessful USER or PASS command

Discussion:
To authenticate using the USER and PASS command combination, the client must first issue the USER command. If the POP3 server responds with a positive status indicator ("+OK"), then the client may issue either the PASS command to complete the authentication, or the QUIT command to terminate the POP3 session. If the POP3 server responds with a negative status indicator ("-ERR") to the USER command, then the client may either issue a new authentication command or may issue the QUIT command.

The server may return a positive response even though no such mailbox exists. The server may return a negative response if mailbox exists, but does not permit plaintext

password authentication.

Possible Responses:
+OK name is a valid mailbox
-ERR never heard of mailbox name

Examples:
C: USER frated
S: -ERR sorry, no mailbox for frated here
...
C: USER mrose
S: +OK mrose is a real hoopy frood

PASS string

Arguments:
a server/mailbox-specific password (required)

Restrictions:
may only be given in the AUTHORIZATION state immediately after a successful USER command

Discussion:
When the client issues the PASS command, the POP3 server uses the argument pair from the USER and PASS commands to determine if the client should be given access to the appropriate maildrop.

Since the PASS command has exactly one argument, a POP3 server may treat spaces in the argument as part of the password, instead of as argument separators.

Possible Responses:
+OK maildrop locked and ready
-ERR invalid password
-ERR unable to lock maildrop

Examples:
C: USER mrose
S: +OK mrose is a real hoopy frood
C: PASS secret
S: -ERR maildrop already locked
...
C: USER mrose
S: +OK mrose is a real hoopy frood
C: PASS secret
S: +OK mrose's maildrop has 2 messages (320 octets)

APOP name digest

Arguments:
a string identifying a mailbox and a MD5 digest string (both required)

Restrictions:
may only be given in the AUTHORIZATION state after the POP3 greeting or after an unsuccessful USER or PASS command

Discussion:
Normally, each POP3 session starts with a USER/PASS exchange. This results in a server/user-id specific password being sent in the clear on the network. For intermittent use of POP3, this may not introduce a sizable risk. However, many POP3 client implementations connect to the POP3 server on a regular basis -- to check for new mail. Further the interval of session initiation may be on the order of five minutes. Hence, the risk of password capture is greatly enhanced.

An alternate method of authentication is required which provides for both origin authentication and replay protection, but which does not involve sending a password in the clear over the network. The APOP command provides this functionality.

A POP3 server which implements the APOP command will include a timestamp in its banner greeting. The syntax of the timestamp corresponds to the 'msg-id' in [RFC822], and MUST be different each time the POP3 server issues a banner greeting. For example, on a UNIX implementation in which a separate UNIX process is used for each instance of a POP3 server, the syntax of the timestamp might be:

<process-ID.clock@hostname>

where 'process-ID' is the decimal value of the process's PID, clock is the decimal value of the system clock, and hostname is the fully-qualified domain-name corresponding to the host where the POP3 server is running.

The POP3 client makes note of this timestamp, and then issues the APOP command. The 'name' parameter has identical semantics to the 'name' parameter of the USER command. The 'digest' parameter is calculated by applying the MD5 algorithm [RFC1321] to a string consisting of the timestamp (including angle-brackets) followed by a shared

secret. This shared secret is a string known only to the POP3 client and server. Great care should be taken to prevent unauthorized disclosure of the secret, as knowledge of the secret will allow any entity to successfully masquerade as the named user. The 'digest' parameter itself is a 16-octet value which is sent in hexadecimal format, using lower-case ASCII characters.

When the POP3 server receives the APOP command, it verifies the digest provided. If the digest is correct, the POP3 server issues a positive response, and the POP3 session enters the TRANSACTION state. Otherwise, a negative response is issued and the POP3 session remains in the AUTHORIZATION state.

Note that as the length of the shared secret increases, so does the difficulty of deriving it. As such, shared secrets should be long strings (considerably longer than the 8-character example shown below).

Possible Responses:
+OK maildrop locked and ready
-ERR permission denied

Examples:
S: +OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>
C: APOP mrose c4c9334bac560ecc979e58001b3e22fb
S: +OK maildrop has 1 message (369 octets)

In this example, the shared secret is the string 'tanstaaf'. Hence, the MD5 algorithm is applied to the string

<1896.697170952@dbc.mtview.ca.us>tanstaaf

which produces a digest value of

c4c9334bac560ecc979e58001b3e22fb

8. Scaling and Operational Considerations

Since some of the optional features described above were added to the POP3 protocol, experience has accumulated in using them in large-scale commercial post office operations where most of the users are unrelated to each other. In these situations and others, users and vendors of POP3 clients have discovered that the combination of using the UIDL command and not issuing the DELE command can provide a weak version of the "maildrop as semi-permanent repository" functionality normally associated with IMAP. Of course the other capabilities of

IMAP, such as polling an existing connection for newly arrived messages and supporting multiple folders on the server, are not present in POP3.

When these facilities are used in this way by casual users, there has been a tendency for already-read messages to accumulate on the server without bound. This is clearly an undesirable behavior pattern from the standpoint of the server operator. This situation is aggravated by the fact that the limited capabilities of the POP3 do not permit efficient handling of maildrops which have hundreds or thousands of messages.

Consequently, it is recommended that operators of large-scale multi-user servers, especially ones in which the user's only access to the maildrop is via POP3, consider such options as:

* Imposing a per-user maildrop storage quota or the like.

  A disadvantage to this option is that accumulation of messages may result in the user's inability to receive new ones into the maildrop. Sites which choose this option should be sure to inform users of impending or current exhaustion of quota, perhaps by inserting an appropriate message into the user's maildrop.

* Enforce a site policy regarding mail retention on the server.

  Sites are free to establish local policy regarding the storage and retention of messages on the server, both read and unread. For example, a site might delete unread messages from the server after 60 days and delete read messages after 7 days. Such message deletions are outside the scope of the POP3 protocol and are not considered a protocol violation.

  Server operators enforcing message deletion policies should take care to make all users aware of the policies in force.

  Clients must not assume that a site policy will automate message deletions, and should continue to explicitly delete messages using the DELE command when appropriate.

  It should be noted that enforcing site message deletion policies may be confusing to the user community, since their POP3 client may contain configuration options to leave mail on the server which will not in fact be supported by the server.

  One special case of a site policy is that messages may only be downloaded once from the server, and are deleted after this has been accomplished. This could be implemented in POP3 server

software by the following mechanism: "following a POP3 login by a client which was ended by a QUIT, delete all messages downloaded during the session with the RETR command". It is important not to delete messages in the event of abnormal connection termination (ie, if no QUIT was received from the client) because the client may not have successfully received or stored the messages. Servers implementing a download-and-delete policy may also wish to disable or limit the optional TOP command, since it could be used as an alternate mechanism to download entire messages.

9. POP3 Command Summary

Minimal POP3 Commands:

USER name	valid in the AUTHORIZATION state
PASS string	
QUIT	
STAT	valid in the TRANSACTION state
LIST [msg]	
RETR msg	
DELE msg	
NOOP	
RSET	
QUIT	

Optional POP3 Commands:

APOP name digest	valid in the AUTHORIZATION state
TOP msg n	valid in the TRANSACTION state
UIDL [msg]	

POP3 Replies:

+OK
-ERR

Note that with the exception of the STAT, LIST, and UIDL commands, the reply given by the POP3 server to any command is significant only to "+OK" and "-ERR". Any text occurring after this reply may be ignored by the client.

10. Example POP3 Session

```
S: <wait for connection on TCP port 110>
C: <open connection>
S:    +OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>
C:    APOP mrose c4c9334bac560ecc979e58001b3e22fb
S:    +OK mrose's maildrop has 2 messages (320 octets)
C:    STAT
S:    +OK 2 320
C:    LIST
S:    +OK 2 messages (320 octets)
S:    1 120
S:    2 200
S:    .
C:    RETR 1
S:    +OK 120 octets
S:    <the POP3 server sends message 1>
S:    .
C:    DELE 1
S:    +OK message 1 deleted
C:    RETR 2
S:    +OK 200 octets
S:    <the POP3 server sends message 2>
S:    .
C:    DELE 2
S:    +OK message 2 deleted
C:    QUIT
S:    +OK dewey POP3 server signing off (maildrop empty)
C:  <close connection>
S:  <wait for next connection>
```

11. Message Format

All messages transmitted during a POP3 session are assumed to conform to the standard for the format of Internet text messages [RFC822].

It is important to note that the octet count for a message on the server host may differ from the octet count assigned to that message due to local conventions for designating end-of-line. Usually, during the AUTHORIZATION state of the POP3 session, the POP3 server can calculate the size of each message in octets when it opens the maildrop. For example, if the POP3 server host internally represents end-of-line as a single character, then the POP3 server simply counts each occurrence of this character in a message as two octets. Note that lines in the message which start with the termination octet need not (and must not) be counted twice, since the POP3 client will remove all byte-stuffed termination characters when it receives a multi-line response.

12. 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, University of Delaware, August 1982.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, MIT Laboratory for Computer Science, April 1992.

[RFC1730] Crispin, M., "Internet Message Access Protocol - Version 4", RFC 1730, University of Washington, December 1994.

[RFC1734] Myers, J., "POP3 AUTHentication command", RFC 1734, Carnegie Mellon, December 1994.

13. Security Considerations

It is conjectured that use of the APOP command provides origin identification and replay protection for a POP3 session. Accordingly, a POP3 server which implements both the PASS and APOP commands should not allow both methods of access for a given user; that is, for a given mailbox name, either the USER/PASS command sequence or the APOP command is allowed, but not both.

Further, note that as the length of the shared secret increases, so does the difficulty of deriving it.

Servers that answer -ERR to the USER command are giving potential attackers clues about which names are valid.

Use of the PASS command sends passwords in the clear over the network.

Use of the RETR and TOP commands sends mail in the clear over the network.

Otherwise, security issues are not discussed in this memo.

14. Acknowledgements

The POP family has a long and checkered history. Although primarily a minor revision to RFC 1460, POP3 is based on the ideas presented in RFCs 918, 937, and 1081.

In addition, Alfred Grimstad, Keith McCloghrie, and Neil Ostroff provided significant comments on the APOP command.

15. Authors' Addresses

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

EMail: jgm+@cmu.edu

Marshall T. Rose
Dover Beach Consulting, Inc.
420 Whisman Court
Mountain View, CA 94043-2186

EMail: mrose@dbc.mtview.ca.us

Appendix A. Differences from RFC 1725

This memo is a revision to RFC 1725, a Draft Standard. It makes the following changes from that document:

- clarifies that command keywords are case insensitive.
- specifies that servers must send "+OK" and "-ERR" in upper case.
- specifies that the initial greeting is a positive response, instead of any string which should be a positive response.
- clarifies behavior for unimplemented commands.
- makes the USER and PASS commands optional.
- clarified the set of possible responses to the USER command.
- reverses the order of the examples in the USER and PASS commands, to reduce confusion.
- clarifies that the PASS command may only be given immediately after a successful USER command.
- clarified the persistence requirements of UIDs and added some implementation notes.
- specifies a UID length limitation of one to 70 octets.
- specifies a status indicator length limitation of 512 octets, including the CRLF.
- clarifies that LIST with no arguments on an empty mailbox returns success.
- adds a reference from the LIST command to the Message Format section
- clarifies the behavior of QUIT upon failure
- clarifies the security section to not imply the use of the USER command with the APOP command.
- adds references to RFCs 1730 and 1734
- clarifies the method by which a UA may enter mail into the transport system.

- clarifies that the second argument to the TOP command is a number of lines.

- changes the suggestion in the Security Considerations section for a server to not accept both PASS and APOP for a given user from a "must" to a "should".

- adds a section on scaling and operational considerations

Appendix B. Command Index