Network Working Group
Internet Draft
Intended status: Informational
Expires: July 2018

M. Holecek
University of Derby
January 11, 2018

# MAIL RETRIEVAL PROTOCOL

## Status of this Memo

Distribution of this memo is unlimited.

## Copyright Notice



## Abstract

The mail retrieval protocol, Version 1 allows a client to access and manipulate electronic mail messages on a server.

MRP includes operations for creating mailboxes, authentication, checking for new messages, permanently removing messages, setting and changing flags, searching, and selective fetching of messages. Messages in MRP are accessed using numbers. These numbers are either message sequence numbers (ID) or unique identifiers (UID).

MRP does not specify a means of posting mail; this function is handled by a mail transfer protocol such as RFC 5321.

Table of Contents

# 1. Protocol Overview

## 1.1. Link Level

The Mail Retrieval Protocol assumes a reliable data stream such as that provided by TCP. When TCP is used, an MR server listens on port 5,000.

## 1.2. Commands and Responses

An MRP connection consists of the establishment of a client/server network connection, an interaction. These client/server interactions consist of a client command, server data, and a server completion result response.

All interactions transmitted by client and server are in the form of lines, that is, strings that ands with a CRLF.

# 2. Conventions used in this document

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

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].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

# 3. Message Numbers

Messages in Mail Retrieval Protocol are accessed by one of two numbers; the identifier number (ID) or unique identifiers (UID).

## 3.1. Message Identifier (ID)

A 32-bit value assigned to each message, which refers to specific message in a database. This number MUST NOT refer to any other message in the database NOT only in single mailbox, but for all mailboxes stored in the database.

The message identifiers are assigned in a strictly ascending fashion in the database; as each message is added to the database it is assigned a higher ID than the messages which were added previously.

The message ID of a message MUST NOT change during the session or between the session.

### 3.2. Message Unique Identifier (UID)

A relative position from 1 to the number of messages in the mailbox. This position is assigned in a strictly ascending fashion in the mailbox. As each new message is added, it is assigned a message unique identifier (UID) that is 1 higher than the number of messages in the mailbox before that new message was added.

The message unique identifiers (UID) can be reassigned during the session. For example, when a message is permanently removed (EXPUNGE Command) from the mailbox, the message unique identifiers for all messages MUST be recalculate.

### 3.3. Message Multiline Replies

The reply text may be longer than single line; in these cases, the complete text must be marked so the client knows when it can stop reading the reply. This requires special format to indicate a multiple line reply.

The format for multiline replies requires that every line, except the last, begin with the star symbol "*" followed by space and by the text. The last line will begin with the OK response, followed by some text and <CRLF>.

Example:

C: SELECT

S: * 13 EXISTS

S: * 1 RECENT

S: * 4 SENT

S: * 2 DRAFT

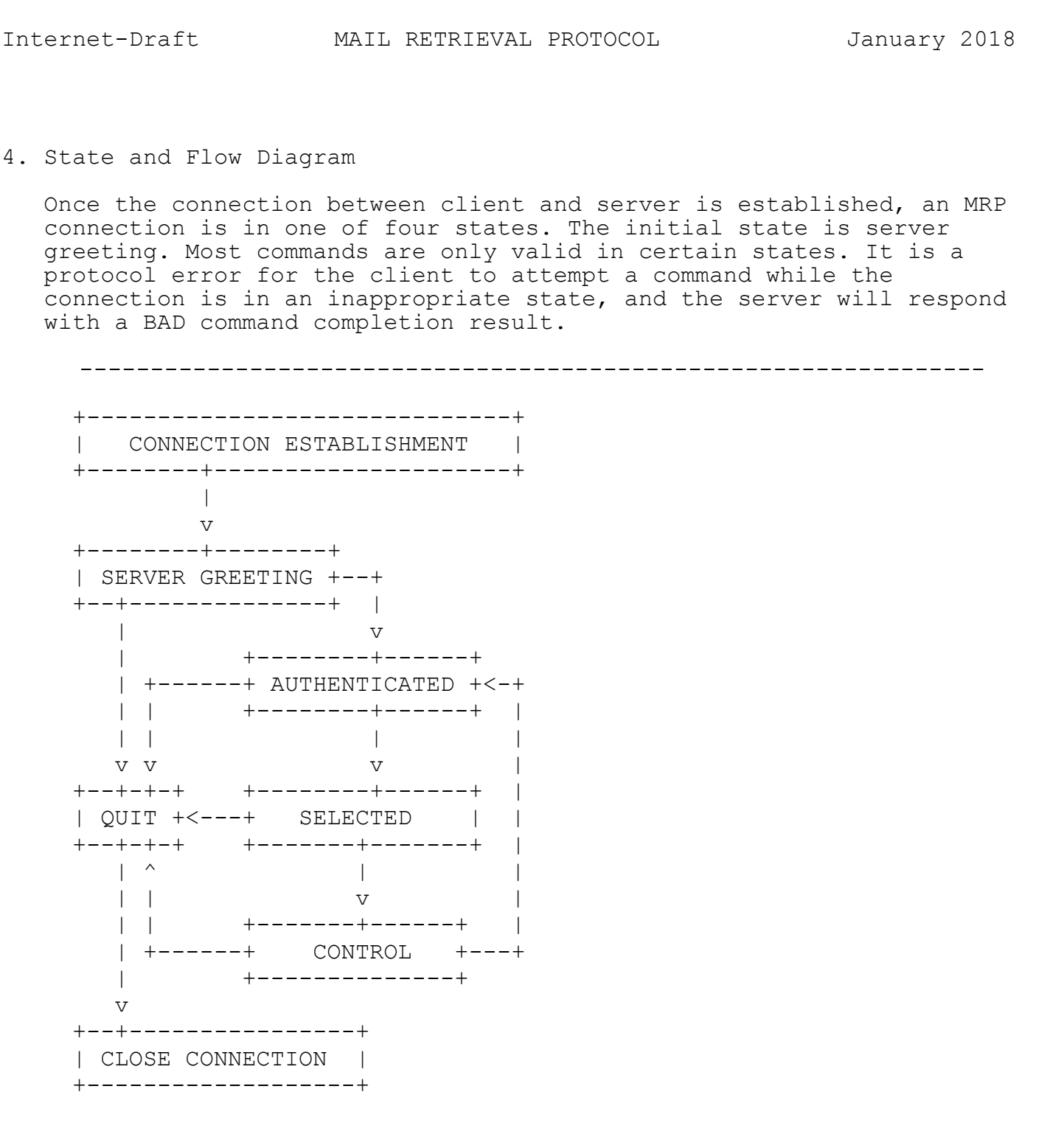
S: * 6 SEEN

S: * 0 DELETED

S: OK SELECT Completed

## 4. State and Flow Diagram

Once the connection between client and server is established, an MRP connection is in one of four states. The initial state is server greeting. Most commands are only valid in certain states. It is a protocol error for the client to attempt a command while the connection is in an inappropriate state, and the server will respond with a BAD command completion result.



### 4.1. Authenticated State

In the not authenticated state, the client MUST supply authentication credentials before most commands will be permitted. This state is entered when a connection starts and allows client to select encryption mode.

### 4.2. Selected State

In a selected state, the client is authenticated and MUST select a mailbox to access before commands that affect messages will be permitted. This state will reset unique identifiers (UID) of all messages starting from 1 (the first received message) to the number of the most recent message.

### 4.3. Control State

In control state, a mailbox has been selected to access. This state is entered when a mailbox has been successfully selected and client is able to fetch, search and manipulate messages belongs to the selected mailbox.

### 4.4. Quit State

In the quit state, the connection is being terminated. This state can be entered as a result of a client request (via the QUIT Command) or by unilateral action on the part of either the client or server.

If the client requests the quit state, the server MUST send an OK response to the client before the server closes the connection; and client MUST read the OK response to the QUIT Command before the client closes the connection.

A server MUST NOT unilaterally close the connection without sending OK response. A client should not unilaterally close the connection, and instead should issue a QUIT Command. If the server detects that the client has unilaterally closed the connection, the server will omit the OK response and simply close its connection.

# 5. Operational Considerations

## 5.1. Mailbox Naming

Mailbox names are 7-bit ASCII format and client implementations MUST NOT attempt to create 8-bit mailbox names. Mailbox names are case-insensitive, and server MUST NOT attempt to create new mailbox with same name.

## 5.2. Autologout Timer

If a server has an inactivity autologout timer, the duration of that timer MUST be at least 10 minutes. The recipient of any command from the client during that interval will reset timer.

# 6. Client Commands

MRP commands are described in this section. Command are organized by the state in which the command is permitted.

## 6.1. Client Commands - Any State

The following commands are valid in any state: NOOP, HELP and QUIT.

### 6.1.1. NOOP Command

The NOOP Command always succeeds. It does nothing. It can be used to reset any inactivity autologout timer on the server.

Example:

C: NOOP

S: OK NOOP Completed

### 6.1.2. HELP Command

The HELP Command causes a server to send helpful information to the client. The command MAY take an argument (any command name) and return more specific information as a response.

Example:

C: HELP

S: * Server works in three STATES:

S: * AUTHENTICATED, SELECTED, CONTROL

S: * AUTHENTICATED STATE COMMANDS:

S: * AUTH, LOGIN, TOKEN and CREATE

S: * SELECTED STATE COMMANDS: SELECT

S: * CONTROL STATE COMMANDS: FETCH, EXPUNGE, CHANGE, SEARCH, LOGOUT

S: * COMMANDS PERMITTED IN ALL STATES:

S: * NOOP, HELP and QUIT

S: * For more info use HELP<SP><COMMAND>

S: OK HELP Completed

### 6.1.3. QUIT Command

The QUIT Command informs the server that the client is done with the connection. The server MUST send an OK response, and then close the network connection.

Example:

C: QUIT

S: OK QUIT Completed

### 6.2. Client Commands – Authenticated State

In addition to the universal commands (NOOP, HELP and QUIT), the following commands are valid in Authenticated state: AUTH, LOGIN, CREATE and TOKEN.

#### 6.2.1. AUTH Command

The AUTH Command allows client to set exchange messages either as a plain text or by using encryption algorithms such as AES or Triple DES.

This command will initiate key exchange by using Diffie-Hellman method. When keys are exchanged, the server and client MUST send randomly generated shared secret which both sides will use for encryption of the messages. More detailed explanation of the Diffie-Hellman method is described in RFC 2631.

There are several encryption modes which client can chose from:

AES/CBC - AES Encryption with Cipher Block Chaining mode

AES/ECB - AES Encryption with Electronic CodeBook mode

DES/CBC - DESede Encryption with Cipher Block Chaining mode

DES/ECB - DESede Encryption with Electronic CodeBook mode

Plain text – messages will be send unencrypted over the network

Example:

C: AUTH AES/CBC

S: * AES/CBC Encryption is established

S: OK AUTH Completed

#### 6.2.2. LOGIN Command

The LOGIN Command identifies the client to the server and carries the plaintext password authenticating this user. In the case a client did not set encryption mode (by AUTH Command) before it sends the LOGIN command, the communication will be unencrypted (plain text) over the network.

Note: Use of the LOGIN Command over an insecure network (such as internet) without using AUTH Command in advance is a security, because anyone monitoring network traffic cab obtain plaintext passwords.

The successful login will generate token which can be used to authenticate client in the future. This token will be override every time the client sends successful LOGIN Command.

Example:

C: LOGIN SMITH PASSWORD123

S: * TOKEN fd725648-932b-45d4-b16a-83c2c5cc2474

S: OK LOGIN Completed

### 6.2.3. CREATE Command

The CREATE Command creates a mailbox with the given name. An OK response is returned only if a new mailbox with that name has been created. It is an error to attempt to create mailbox with a name that refers to an extant mailbox. Any error in creation will return a BAD response.

There are some criteria for creating mailbox:

a. Mailbox name MUST be minimal 8 and maximal 30 characters long
b. Mailbox name MUST contain only 7-bits ASCII characters
c. Password MUST be minimal 8 and maximal 30 characters long

This command will send OK response to the client with randomly generated token which can be used to authenticate client in the future.

Example:

C: CREATE SMITHEEE PASSWORD123

S: * TOKEN fd725648-932b-45d4-b16a-83c2c5cc2474

S: OK CREATE Completed

#### 6.2.4. TOKEN Command

The TOKEN Command identifies the client to the server and carries randomly generated token authenticating this user.

This token is generated after user is successfully authenticated by LOGIN or CREATE Command and MUST be send to the client in the server response. Client MAY store this token and use it for authentication to the server in the future without using LOGIN Command.

Token is stored in database and it can be reuse after closing network connection or after sending LOGOUT Command.

Example:

C: TOKEN SMITHEEE fd725648-932b-45d4-b16a-83c2c5cc2474

S: OK TOKEN Completed

### 6.3. Client Commands – Select State

In addition to the universal commands (NOOP, HELP and QUIT), the following command is valid in Select State: SELECT.

#### 6.3.1. SELECT Command

The SELECT Command selects a mailbox so that messages in the mailbox can be accessed. Before returning an OK to the client, the server MUST reset unique identifiers (UID) of all messages.

The server response MUST return following tags: EXISTS, RECENT, SENT, DRAFT, SEEN and DELETED.

In front of each of these tags MUST be the number of messages which corresponds to those flags. These will be discussed further in this document.

The SELECT Command MUST be called before any command from Control State otherwise the server MUST response with BAD Syntax error.

Example:

C: SELECT

S: * 13 EXISTS

S: * 1 RECENT

S: * 4 SENT

S: * 2 DRAFT

S: * 6 SEEN

S: * 0 DELETED

S: OK SELECT Completed

### 6.4. Client Commands – Control State

In the Control State, commands that manipulate messages in a mailbox are permitted.

In addition to the universal commands (NOOP, HELP and QUIT), the following commands are valid in Control State: FETCH, SEARCH, CHANGE, EXPUNGE and LOGOUT.

#### 6.4.1. FETCH Command

The FETCH Command retrieves data associated with a message in the mailbox.

There are three ways of use this command to fetch specific messages:

SEQUENCE

This will FETCH messages in the range of two numbers separated by colon which represents unique identifiers (UID) of the messages.

SINGLE

This will FETCH only one message corresponds to the number which represent UID of that message.

FLAGS

This will FETCH messages which are matching to the individual flags that are set for each message.

The fetch command allows of chaining, for example the client can retrieve SEQUENCE of messages with corresponding FLAGS or the client can retrieve messages with two or more flags.

In the case when client uses the chaining, the client MUST write SEQUENCE or SINGLE message UID before writing a FLAGS.

However, in case that the client wants to FETCH only messages corresponding to the FLAGS and not specific messages then using only FLAGS is permitted.

The following FLAGS are valid as the arguments of the FETCH Command. Refer to the Formal Syntax section for the precise syntactic definitions of the arguments.

ALL

All messages in the mailbox will be retrieved.

RECENT

Messages that have the RECENT flag set.

SENT

Messages that have the SENT flag set.

DRAFT

Messages that have the DRAFT flag set.

SEEN

Messages that have the SEEN flag set.

DELETED

Messages that have the DELETED flag set.

The FETCH Command response returns data about a message to the client. The response starts with multiline star symbol “*” and space, followed by message ID and SIZE of the message. Every message send to the client will start with same response. However, the data of every message are NOT prefixed with multiline symbol (“*”) which allows more effective separation of messages. The client can search for star symbol at the beginning during the parsing of all messages.

Example:

C: FETCH RECENT

S: * FETCH ID 48 SIZE 135

S: UID: 1

S: Sender: martin.holecek@icloud.com

S: Recipient: c.windmill@derby.ac.uk

S: Subject: Empty Subject

S: Date: 2007-12-08

S: This is the message

S: OK FETCH Completed

### 6.4.2. SEARCH Command

The SEARCH Command searches the mailbox for messages that match the given searching criteria. The searching criteria consist of a search key and search value.

The defined search keys are as follows. Refer to the Formal Syntax section for the precise syntactic definitions of the arguments.

ALL

Search for specific string in the body, subject, sender, or recipient of the message.

BODY

Search for specific string only in the body of the message.

SUBJECT

Search for specific string only in the subject of the message.

SENDER

Search for specific string only in the sender of the message.

RECIPIENT

Search for specific string only in the recipient of the message.

SINCE

Search for specific date where the date of the message is within or later than the specified date.

UNTIL

Search for specific date where the date of the message is earlier than the specified date.

Search value uses a string which MUST consist of characters from the ASCII character set. The matching string are case sensitive.

SEARCH Command Response

The SEARCH Command response from the server contains a listing of message sequence numbers corresponding to those messages that match the searching criteria. These numbers are unique identifiers (UID) of the messages. Each number is delimited by a space. If the SEARCH results in no matches, the server MUST return SEARCH FOUND NO RESULTS followed by OK Response.

Example:

C: SEARCH BODY=[My name is Tony]

S: * SEARCH 5 9 13

S: OK SEARCH Completed

C: SEARCH SENDER=[bobby]

S: * SEARCH FOUND NO RESULTS

S: OK SEARCH Completed

### 6.4.3. CHANGE Command

The CHANGE Command alters FLAG associated with a message in the mailbox.

The message which the client wants to alter MUST be specified by the ID of this message NOT by the unique identifiers (UID. If the ID does not match any of the stored messages of the mailbox then server MUST reply with BAD response.

The defined FLAGS are as follows. Refer to the Formal Syntax section for the precise syntactic definitions of the arguments.

RECENT

Replace the flag for the message with the RECENT flag.

SENT

Replace the flag for the message with the SENT flag.

DRAFT

Replace the flag for the message with the DRAFT flag.

SEEN

Replace the flag for the message with the SEEN flag.

DELETED

Replace the flag for the message with the DELETED flag.

Example:

C: CHANGE 41 RECENT

S: OK CHANGE Completed

C: CHANGE 34324 RECENT

S: BAD Message ID is not valid!

### 6.4.4. EXPUNGE Command

The EXPUNGE Command permanently removes all messages that have the DELETED flag set from the currently selected mailbox.

The server will reply with OK response if the messages has been deleted or it will replay with BAD in case of syntax error.

Example:

C: EXPUNGE

S: OK EXPUNGE Completed

#### 6.4.5. LOGOUT Command

The LOGOUT Command informs the server that the client is done with the mailbox. This will close current mailbox and it allows user to login with different credentials. The internal state will be changed to the authenticated state.

Example:

C: LOGOUT

S: OK LOGOUT Completed

## **7.** Formal Syntax

The commands consist of a command code followed by an argument or multiple arguments. Command codes are case-insensitive and consists only alphabetic characters. Arguments are mostly case-insensitive, however there are some exceptions to this rule which were already discussed in Client Commands section.

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

Note: [ABNF] rules MUST be followed strictly.

```
auth           =  “AUTH” SP encryption

change         =  “CHANGE” SP message-id SP flag

command        =  "AUTH" / "LOGIN" / "TOKEN" / "CREATE" / “SELECT” /
                  "FETCH" /   "EXPUNGE" / "CHANGE" / "SEARCH" /
                  "LOGOUT" / "NOOP" / "QUIT" / "HELP"

command-any    =  “NOOP” / “HELP” / “QUIT”
                  ; Valid in all states

create         =  “CREATE” SP mailbox SP password

date           =  date-year “-“ month “-“ day

date-year      =  4DIGIT

day            =  2DIGIT
                  ; Day of month

digit-nz       =  %x31-39
                  ; 1-9

encryption     =  “AES/ECB” / “AES/CBC” / “DES/ECB” / “DES/CBC”

expunge        =  “EXPUNGE”
```

```
   fetch          =  "FETCH" 1*([SP sequence-set] *([SP flag]))
   flag           =  "ALL" / "RECENT" / "SENT" / "DRAFT" / "SEEN" /
                     "DELETED"
                     ; Flags are case-insensitive
   help           =  "HELP" [SP command]
   login          =  "LOGIN" SP mailbox SP password
   logout         =  "LOGOUT"
   mailbox        =  string
   message-data   =  string
   message-id     =  number
                     ; The message ID is a specific number which
                     ; corresponds to individual message in the
                     ; database NOT only specific mailbox, but all
                     ; mailboxes stored in database. This number is
                     ; unique to all messages.
   month          =  2DIGIT
                     ; month of the year
   number         =  1*DIGIT
                     ; Non-zero unsigned 32-bit integer
                     ; (0 < n < 4,294,967,296)
   number-nz      =  digit-nz *DIGIT
                     ; Non-zero unsigned 32-bit integer
                     ; (0 < n < 4,294,967,296)
   password       =  string
```

```
   response       =  [response-data] response-done
   response-data  =  "*" SP response-text / response-token /
                     message-data
   response-done  =  ("OK" / "BAD") SP string
   response-text  =  string / number-nz SP "EXISTS" /
                     number-nz SP "RECENT" / number-nz SP "SENT" /
                     number-nz SP "DRAFT" / number-nz SP "SEEN" /
                     number-nz SP "DELETED"
   response-token =  "TOKEN" SP token-string
   search         =  "SEARCH" SP search-key "=" "[" search-value "]"
   search-key     =  "ALL" / "BODY" / "SUBJECT" / "SENDER" /
                     "RECIPIENT" / "SINCE" / "UNTIL"
   search-value   =  1*CHAR / date
                     ; any one of the 128 ASCII characters except "]"
   select         =  "SELECT"
   sequence-set   =  seq-number / seq-range
                     ; The server should respond with a BAD response to
                     ; a command that uses a message sequence number
                     ; greater than the number of messages in the
                     ; selected mailbox.
   seq-number     =  number-nz
                     ; This sequence corresponds to unique identifiers
                     ; (UID) of the messages which are specific to the
                     ; mailbox and strictly ascending
```

```
seq-range      =  seq-number ":" seq-number
                  ; Two seq-number values and all values between
                  ; these two regardless of order.
                  ; Example 2:4 indicate values 2, 3 and 4
token          =  "TOKEN" SP mailbox SP token-string
token-string   =  string
                  ; Randomly generated string of 36 characters
```

## 8. Security Considerations

Message Retrieval Protocol transaction, including electronic mail data, are sent in the clear text over the network unless protection from snooping is negotiated. This can be accomplished using AUTH Command which MUST provide mechanism for key exchange using Diffie-Hellman method as described in RFC 2631.

## 9. References

The following documents contain definitions or specifications that are necessary to understand this document properly:

[ABNF] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

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

[RFC2631] W. Diffie and M. E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644-654.

[RFC5321] Klensin, J., “Simple mail transfer protocol (SMTP)”, RFC 5321, October 2008.