Reticulum M. Faragher
Between The Borders
January 2024

Post Office Protocol - Reticulum

Abstract

This document describes a variant of the Post Office Protocol, Version 3, tailored for the LXMF and Reticulum. All feedback is welcome in any applicable Reticulum channel. This is not an official publication of Reticulum and is solely the responsibility of the author.

Contents

1 Introduction **2**

2 Basic Operation **2**

3 AUTHORIZATION state **3**

4 TRANSACTION state **3**
4.1 STAT 3
4.2 LIST [msg] 4
4.3 RETR [msg] 5
4.4 DELE [msg] 5
4.5 NOOP 6
4.6 RSET 6
4.7 UIDL [msg] 6
4.8 QUIT 7

1. Introduction

Reticulum faces many of the issues faced by ARPANET for precisely the same reasons. Mail to offline systems or access to mail from multiple systems was generally impossible using UNIX-style mail handlers. The Post Office Protocol was designed to remedy these shortcomings in a practical and efficient way. POP3 explicitly has a limited feature set when compared to IMAP and was designed to be simple and extensible. This following note, made on page 11 of RFC 1939, is the touchstone for this implementation.

> **In short, the philosophy of this memo is to put intelligence in the part of the POP3 client and not the POP3 server.**

Reticulum and LXMRouter take care of routing, feature sets, and offline delivery, but there is no multiple device access in LXMF clients due to the way identities, addresses, and paths are handled in Reticulum. The narrow and efficient feature set of Reticulum and POP3 are in alignment, and a modified Post Office Protocol can allow a multiple-access messaging system, either as a proxy for an lxmf.delivery address or a stand alone protocol (suggested: popr.delivery).

The feature set is limited by design, and only Reticulum authentication is acceptable. Due to the way these messages are handled, they will be ended up stored in the clear on the server, and therefore either require a trusted server or an encrypted message only decodable by the client application. As this protocol is meant to be run for a single address on a single server, this is deemed acceptable, as its security is similar to a Nomad Network instance.

2. Basic Operation

Initial connection begins with a Reticulum link and identification. Once this link is established, the server will send a greeting either noting a successful connection, or a reason for rejection (such as an unidentified link). All communication to the server is done via Reticulum requests, and responses (excepting RETR requests) are UTF-8 encoded strings starting with ‘‘+OK’’ for a positive response or ‘‘-ERR’’ for a negative response.

By using requests, the passed data is automatically encapsulated and requires less client or server controlled sanitization or logic. The returned string is handled similarly and can contain newlines to generate multi-line responses.

As with POP3, POPR passes through three states during operation, AUTHENTICATION, TRANSACTION, and UPDATE. AUTHORIZATION is brief compared to POP3, as the server either sees an identified link that matches a trusted identity and proceeds to the TRANSACTION stage, or it does not and rejects the connection. During the TRANSACTION phase the client can request information about the messages the server contains, request a specific message, tag a file for deletion, or other optional commands. No messages are deleted until the UPDATE phase, and that phase only follows a successful QUIT command, meaning a failed transaction will not affect the messages on the server.

Unlike POP3, an unimplemented command will not be responded to in any way, due to the nature of Reticulum requests, and handling failed requests are the duty of the client.

Any timeout should exceed ten minutes, with any command resetting the timer, independent of any Reticulum timeouts. A server MUST NOT enter the UPDATE state on a timeout or link closure.

3. AUTHORIZATION state

On establishment of a link, the server sends a single line greeting, confirming its status. Example:

```
S: +OK POPR GO
```

The server may, but does not need to identify itself. Due to its public key being on record with the client, it has already verified its identity. At this point the client should identify itself. If any command is entered or any message sent without identification, the server will disconnect with a negative status indicator. Example:

```
S: -ERR NO ID
```

If the server receives an identity that isn't authorized to access the address, it will disconnect with a negative status indicator. Example:

```
S: -ERR NO AUTH
```

Once the client has been authorized, the messages and their size is determined, and they are listed in order starting with '1' and listed in base 10. The message store is then locked, preventing other instances from accessing the mailbox. This lock is released either on a quit or an abnormal termination. Due to the nature of the Reticulum link, it is wise to check for a lock when the link closes and remove it if present.

The server MAY allow a currently logged in user to log in again, terminating the previous link without updating, on the assumption that the previous session terminated abnormally.

The dynamic updating of this list to reflect new messages MAY NOT be implemented to disincentivize keeping a session open rather than using the server in a transactional manner.

4. TRANSACTION state

Once the client is authenticated and the mailbox is locked, the session enters the TRANSACTION state. The client may issue any number of the following requests any number of times. After each command the server issues a response. When the client issues the QUIT command, the session enters the UPDATE state. A session terminated in any other way closes the connection without a response and without entering the UPDATE state.

NOTE: The following commands use a POP style text input, however, the command is the request path, and the argument is the data to be sent. The traditional format is retained for human readability.

4.1. STAT

Arguments: None

Restrictions:

TRANSACTION state only

Discussion:

Server replies with a positive state response, followed by the number of messages and total mailbox size in bytes, all separated with a single space separating the fields and a CRLF pair to end the message. In contrast to the POP3 standard, which strongly advises against additional information, the POPR standard dictates that the response MUST contain exactly this information and MUST NOT contain additional information.

Both message number and mailbox size are expressed in base 10, and the size is listed in bytes (as opposed to POP's octets)

Message number and mailbox size do not include those marked for deletion.

Possible Responses:

```
+OK nn mm
```

Example(s):

```
C: STAT
S: +OK 6 2048
```

4.2. LIST [msg]

Arguments:

An optional message number which cannot refer to a message marked for deletion.

Restrictions:

TRANSACTION state only

Discussion:

When a message number is specified, that message and its size are listed. If no number is specified, a list of all of the messages is provided. The format is the message number, followed by its size in bytes.

Possible Responses:

```
+OK nn mm


+OK
nn mm
nn mm
nn mm


-ERR
```

Example(s):

```
C: LIST 3
S: +OK 3 453


C: LIST
S:+OK
1 342
```

```
2 235
3 453
4 623
```

4.3. RETR [msg]

Arguments: A message number which cannot refer to a message marked for deletion.

Restrictions:

TRANSACTION state only

Discussion:

Retrieves a message from the server. A negative response is the usual -ERR. A successful response is an LXMessage packed to bytes using the packed_container() method.

Possible Responses:

<LXM packed container>

-ERR

Example(s):

```
C: RETR 2 S: <LXM packed container as bytes>


C: RETR 99 S: -ERR
```

4.4. DELE [msg]

Arguments: A message number which cannot refer to a message marked for deletion.

Restrictions:

TRANSACTION state only

Discussion:

Marks a message for deletion. Does not delete the message.

Possible Responses:

+OK

-ERR

Example(s):

```
C: DELE 1
S: +OK
```

4.5. NOOP

Arguments: None

Restrictions:

TRANSACTION state only

Discussion:

No Operation. Does nothing, but does refresh the session timer.

Possible Responses:

+OK

Example(s):

```
C: NOOP
S: +OK
```

4.6. RSET

Arguments: None

Restrictions:

TRANSACTION state only

Discussion:

Resets mailbox state. Removes the marks for deletion from all messages in the mailbox.

Possible Responses:

+OK

Example(s):

```
C: RSET
S: +OK
```

4.7. UIDL [msg]

Arguments: An optional unique identifier which cannot refer to a message marked for deletion.

Restrictions:

TRANSACTION state only

Discussion:

With no message specified, lists all messages in the mailbox with the format of ‘message-no UID’. LX Messages are identified with their hash. Specifying a hash will search the message store and return the message number and hash if found.

Possible Responses:

```
+OK
mm nn
mm nn
...
```

```
+OK nn mm
```

```
-ERR
```

Example(s):

```
C: UIDL
S: +OK
1 882fd83776434c4a2dfc861c8b1f063815f1c75a6e3c552ad06bfb5e35478a2b
2 e38dfe8919527053b76238092a9d37b378eef7d01b43ed7522d0969610a0d2e9
3 3b0d15ee1725a0b45e9e02b8fa404d143d402c7a0fcda7ce13cd84046f6e063b
```

```
C: UIDL e38dfe8919527053b76238092a9d37b378eef7d01b43ed7522d0969610a0d2e9
S: +OK 2 e38dfe8919527053b76238092a9d37b378eef7d01b43ed7522d0969610a0d2e9
```

4.8. QUIT

Arguments: None

Restrictions:

TRANSACTION state only

Discussion:

Terminate connection. Moves to UPDATE state and closes the link.

Possible Responses:

NONE

Example(s):

```
C: QUIT
S: -Link closed-
```