POST OFFICE PROTOCOL - VERSION 2


Status of this Memo

   This RFC suggests a simple method for workstations to dynamically
   access mail from a mailbox server.  This RFC specifies a proposed
   protocol for the ARPA-Internet community, and requests discussion and
   suggestions for improvement.  This memo is a revision of RFC 918.
   Distribution of this memo is unlimited.

Introduction

   The intent of the Post Office Protocol Version 2 (POP2) is to allow a
   user's workstation to access mail from a mailbox server.  It is
   expected that mail will be posted from the workstation to the mailbox
   server via the Simple Mail Transfer Protocol (SMTP).  For further
   information see RFC-821 [1] and RFC-822 [2].

   This protocol assumes a reliable data stream such as provided by TCP
   or any similar protocol.  When TCP is used, the POP2 server listens
   on port 109 [4].

System Model and Philosophy

   While we view the workstation as an Internet host in the sense that
   it implements IP, we do not expect the workstation to contain the
   user's mailbox.  We expect the mailbox to be on a server machine.

   We believe it is important for the mailbox to be on an "always up"
   machine and that a workstation may be frequently powered down, or
   otherwise unavailable as an SMTP server.

   POP2 is designed for an environment of workstations and servers on a
   low-delay, high-throughput, local networks (such as Ethernets).  POP2
   may be useful in other environments as well, but if the environment
   is substantially different, a different division of labor between the
   client and server may be appropriate, and a different protocol
   required.

   Suppose the user's real name is John Smith, the user's machine is
   called FIDO, and that the mailbox server is called DOG-HOUSE.  Then

we expect the user's mail to be addressed to JSmith@DOG-HOUSE.ARPA
(not JSmith@FIDO.ARPA).

That is, the destination of the mail is the mailbox on the server
machine.  The POP2 protocol and the workstation are merely a
mechanism for viewing the messages in the mailbox.

The user is not tied to any particular workstation for accessing his
mail.  The workstation does not appear as any part of the mailbox
address.

This is a very simple protocol.  This is not a user interface.  We
expect that there is a program in the workstation that is friendly to
the user.  This protocol is not "user friendly".  One basic rule of
this protocol is "if anything goes wrong close the connection".
Another basic rule is to have few options.

POP2 does not parse messages in any way.  It does not analyze message
headers (Date:, From:, To:, Cc:, or Subject:).  POP2 simply transmits
whole messages from a mailbox server to a client workstation.

The Protocol

The POP2 protocol is a sequence of commands and replies.  The design
draws from many previous protocols of the ARPA-Internet community.

The server must be listening for a connection.  When a connection
is opened the server sends a greeting message and waits for
commands.  When commands are received the server acts on them and
responds with replies.

The client opens a connection, waits for the greeting, then sends
the HELO command with the user name and password arguments to
establish authorization to access mailboxes.  The server returns
the number of messages in the default mailbox.

The client may read the default mailbox associated with the user
name or may select another mailbox by using the FOLD command.  The
server returns the number of messages in the mailbox selected.

The client begins a message reading transaction with a READ
command.  The read command may optionally indicate which message
number to read, the default is the current message (incremented
when a message is read and set to one when a new folder is
selected).  The server returns the number of characters in the
message.

The client asks for the content of the message to be sent with the
RETR command.  The server sends the message data.

When all the data has been received the client sends an
acknowledgment command.  This is one of ACKS, ACKD, and NACK.

   ACKS means "I've received the message successfully and please
   keep it in the mailbox".

   ACKD means "I've received the message successfully and please
   delete it from the mailbox".

   NACK means "I did not receive the message and please keep it in
   the mailbox".

In the case of ACKS or ACKD the server increments the current
message indicator.  In the case of NACK the current message
indicator stays the same.

In all cases the server returns the number of characters in the
(now) current message.

The client terminates the session with the QUIT command.  The
server returns an ok.

The Normal Scenario

```
        Client                     Server
        ------                     ------
                           Wait for Connection
   Open Connection  -->
                       <--  + POP2 Server Ready
                            Wait for Command
   HELO Fred Secret -->
                       <--  #13 messages for you
                            Wait for Command
   READ 13          -->
                       <--  =537 characters in that message
                            Wait for Command
   RETR             -->
                       <--  (send the message data)
                            Wait for Command
   ACKS             -->
                       <--  =0 no more messages
                            Wait for Command
   QUIT             -->
                       <--  + OK
   Close connection --> <--  Close connection
                            Wait for Connection (go back to start)
```

Conventions

   Arguments

      These arguments have system specific definitions.

         user - A login account name.

         password - The password for the login account.

         mailbox - A mailbox name (also called a mail folder).

Default Mailboxes

    TOPS-20

        MAIL.TXT.1 - from login directory

    UNIX

        both
            /usr/spool/mail/user
        and
            /usr/user/Mail/inbox/*

            where "user" is the user value supplied in the HELO command.

End of Line

    End of Line is Carriage Return (CR) followed by Line Feed (LF).
    This sequence is indicated by "CRLF" in this document.  This end
    of line convention must be used for commands and replies.

Message Length

    The reply to the READ command or an acknowledgment command (ACKS,
    ACKD, NACK) is the length (a character count) of the next message
    to be transmitted.  This includes all the characters in the data
    transmitted.  CRLF counts as two characters.  A length of zero
    means the message does not exist or is empty.  A request to
    transmit a message of zero length will result in the server
    closing the connection.  The message is transmitted in the
    standard internet format described in RFC-822 [2] and NVT-ASCII.
    This may be different from the storage format and may make
    computing the message length from the stored message non-trivial.

Message Numbers

    The reply to the HELO and FOLD commands is a count of the number
    of messages in a the selected mailbox.  The READ command has a
    message number as an optional argument.  These numbers are
    decimal, start at one, and computed with respect to the current
    mailbox.  That is, the first message in a mailbox is message
    number 1.

Numbers

    All numbers in this memo and protocol are decimal.

# DOCKET ALARM

# Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

## Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

## Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

## Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

## API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

### LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

### FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

### E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

fastcase®
Smarter legal research.