Network Working Group
Request for Comments: 2449
Updates: 1939
Category: Standards Track

R. Gellens
Qualcomm
C. Newman
Innosoft
L. Lundblade
Qualcomm

November 1998

POP3 Extension Mechanism

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.

Copyright Notice

IESG Note

This extension to the POP3 protocol is to be used by a server to express policy descisions taken by the server administrator. It is not an endorsement of implementations of further POP3 extensions generally. It is the general view that the POP3 protocol should stay simple, and for the simple purpose of downloading email from a mail server. If more complicated operations are needed, the IMAP protocol [RFC 2060] should be used. The first paragraph of section 7 should be read very carefully.

Table of Contents

1. Introduction

The Post Office Protocol version 3 [POP3] is very widely used. However, while it includes some optional commands (and some useful protocol extensions have been published), it lacks a mechanism for advertising support for these extensions or for behavior variations.

Currently these optional features and extensions can only be detected by probing, if at all. This is at best inefficient, and possibly worse. As a result, some clients have manual configuration options for POP3 server capabilities.

Because one of the most important characteristics of POP3 is its simplicity, it is desirable that extensions be few in number (see section 7). However, some extensions are necessary (such as ones that provide improved security [POP-AUTH]), while others are very desirable in certain situations. In addition, a means for discovering server behavior is needed.

This memo updates RFC 1939 [POP3] to define a mechanism to announce support for optional commands, extensions, and unconditional server behavior. Included is an initial set of currently deployed capabilities which vary between server implementations, and several new capabilities (SASL, RESP-CODES, LOGIN-DELAY, PIPELINING, EXPIRE and IMPLEMENTATION). This document also extends POP3 error messages so that machine parsable codes can be provided to the client. An initial set of response codes is included. In addition, an [ABNF] specification of POP3 commands and responses is defined.

Public comments should be sent to the IETF POP3 Extensions mailing list, <ietf-pop3ext@imc.org>. To subscribe, send a message containing SUBSCRIBE to <ietf-pop3ext-request@imc.org>.

2. Conventions Used in this Document

The key words "REQUIRED", "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [KEYWORDS].

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

3. General Command and Response Grammar

The general form of POP3 commands and responses is described using [ABNF]:

POP3 commands:

```
command      =  keyword *(SP param) CRLF    ;255 octets maximum
keyword      =  3*4VCHAR
param        =  1*VCHAR
```

POP3 responses:

```
response     =  greeting / single-line / capa-resp / multi-line
capa-resp    =  single-line *capability "." CRLF
capa-tag     =  1*cchar
capability   =  capa-tag *(SP param) CRLF   ;512 octets maximum
cchar        =  %x21-2D / %x2F-7F
                    ;printable ASCII, excluding "."
dot-stuffed  =  *CHAR CRLF                  ;must be dot-stuffed
gchar        =  %x21-3B / %x3D-7F
                    ;printable ASCII, excluding "<"
greeting     =  "+OK" [resp-code] *gchar [timestamp] *gchar CRLF
                    ;512 octets maximum
multi-line   =  single-line *dot-stuffed "." CRLF
rchar        =  %x21-2E / %x30-5C / %x5E-7F
                    ;printable ASCII, excluding "/" and "]"
resp-code    =  "[" resp-level *("/" resp-level) "]"
resp-level   =  1*rchar
schar        =  %x21-5A / %x5C-7F
                    ;printable ASCII, excluding "["
single-line  =  status [SP text] CRLF       ;512 octets maximum
status       =  "+OK" / "-ERR"
text         =  *schar / resp-code *CHAR
timestamp    =  "<" *VCHAR ">"
                    ;MUST conform to RFC-822 msg-id
```

4. Parameter and Response Lengths

This specification increases the length restrictions on commands and parameters imposed by RFC 1939.

The maximum length of a command is increased from 47 characters (4 character command, single space, 40 character argument, CRLF) to 255 octets, including the terminating CRLF.

Servers which support the CAPA command MUST support commands up to 255 octets. Servers MUST also support the largest maximum command length specified by any supported capability.

The maximum length of the first line of a command response (including the initial greeting) is unchanged at 512 octets (including the terminating CRLF).

5. The CAPA Command

The POP3 CAPA command returns a list of capabilities supported by the POP3 server. It is available in both the AUTHORIZATION and TRANSACTION states.

A capability description MUST document in which states the capability is announced, and in which states the commands are valid.

Capabilities available in the AUTHORIZATION state MUST be announced in both states.

If a capability is announced in both states, but the argument might differ after authentication, this possibility MUST be stated in the capability description.

(These requirements allow a client to issue only one CAPA command if it does not use any TRANSACTION-only capabilities, or any capabilities whose values may differ after authentication.)

If the authentication step negotiates an integrity protection layer, the client SHOULD reissue the CAPA command after authenticating, to check for active down-negotiation attacks.

Each capability may enable additional protocol commands, additional parameters and responses for existing commands, or describe an aspect of server behavior. These details are specified in the description of the capability.

Section 3 describes the CAPA response using [ABNF]. When a capability response describes an optional command, the <capa-tag> SHOULD be identical to the command keyword. CAPA response tags are case-insensitive.

CAPA

Arguments:
none

Restrictions:
none

Discussion:
An -ERR response indicates the capability command is not implemented and the client will have to probe for capabilities as before.

An +OK response is followed by a list of capabilities, one per line. Each capability name MAY be followed by a single space and a space-separated list of parameters. Each capability line is limited to 512 octets (including the CRLF). The capability list is terminated by a line containing a termination octet (".") and a CRLF pair.

Possible Responses:
+OK -ERR

Examples:

```
C: CAPA
S: +OK Capability list follows
S: TOP
S: USER
S: SASL CRAM-MD5 KERBEROS_V4
S: RESP-CODES
S: LOGIN-DELAY 900
S: PIPELINING
S: EXPIRE 60
S: UIDL
S: IMPLEMENTATION Shlemazle-Plotz-v302
S: .
```

6. Initial Set of Capabilities

This section defines an initial set of POP3 capabilities. These include the optional POP3 commands, already published POP3 extensions, and behavior variations between POP3 servers which can impact clients.

Note that there is no APOP capability, even though APOP is an optional command in [POP3]. Clients discover server support of APOP by the presence in the greeting banner of an initial challenge enclosed in angle brackets ("<>"). Therefore, an APOP capability would introduce two ways for a server to announce the same thing.

6.1. TOP capability

CAPA tag:
TOP

Arguments:
none

Added commands:
TOP

Standard commands affected:
none

Announced states / possible differences:
both / no

Commands valid in states:
TRANSACTION

Specification reference:
[POP3]

Discussion:
The TOP capability indicates the optional TOP command is available.

6.2. USER capability

CAPA tag:
USER

Arguments:
none

Added commands:
USER PASS

Standard commands affected:
none

Announced states / possible differences:
both / no

Commands valid in states:
AUTHENTICATION

Specification reference:
[POP3]

Discussion:
The USER capability indicates that the USER and PASS commands are supported, although they may not be available to all users.

6.3. SASL capability

CAPA tag:
SASL

Arguments:
Supported SASL mechanisms

Added commands:
AUTH

Standard commands affected:
none

Announced states / possible differences:
both / no

Commands valid in states:
AUTHENTICATION

Specification reference:
[POP-AUTH, SASL]

Discussion:
The POP3 AUTH command [POP-AUTH] permits the use of [SASL] authentication mechanisms with POP3. The SASL capability indicates that the AUTH command is available and that it supports an optional base64 encoded second argument for an initial client response as described in the SASL specification. The argument to the SASL capability is a space separated list of SASL mechanisms which are supported.

6.4. RESP-CODES capability

CAPA tag:
RESP-CODES

Arguments:
none

Added commands:
none

Standard commands affected:
none

Announced states / possible differences:
both / no

Commands valid in states:
n/a

Specification reference:
this document

Discussion:
The RESP-CODES capability indicates that any response text issued by this server which begins with an open square bracket ("[") is an extended response code (see section 8).

6.5. LOGIN-DELAY capability

CAPA tag:
LOGIN-DELAY

Arguments:
minimum seconds between logins; optionally followed by USER in AUTHENTICATION state.

Added commands:
none

Standard commands affected:
USER PASS APOP AUTH

Announced states / possible differences:
both / yes

Commands valid in states:
n/a

Specification reference:
this document

Discussion:
POP3 clients often login frequently to check for new mail. Unfortunately, the process of creating a connection, authenticating the user, and opening the user's maildrop can be very resource intensive on the server. A number of deployed POP3 servers try to reduce server load by requiring a delay between logins. The LOGIN-DELAY capability includes an integer argument which indicates the number of seconds after an "+OK" response to a PASS, APOP, or AUTH command before another authentication will be accepted. Clients which permit the user to configure a mail check interval SHOULD use this capability to determine the minimum permissible interval. Servers which advertise LOGIN-DELAY SHOULD enforce it.

If the minimum login delay period could differ per user (that is, the LOGIN-DELAY argument might change after authentication), the server MUST announce in AUTHENTICATION state the largest value which could be set for any user. This might be the largest value currently in use for any user (so only one value per server), or even the largest value which the server permits to be set for any user. The server SHOULD append the token "USER" to the LOGIN-DELAY parameter in AUTHENTICATION state, to inform the client that a more accurate value is available after authentication. The server SHOULD announce the more accurate value in TRANSACTION state. (The "USER" token allows the client to decide if a second CAPA command is needed or not.)

Servers enforce LOGIN-DELAY by rejecting an authentication command with or without the LOGIN-DELAY error response. See section 8.1.1 for more information.

6.6. PIPELINING capability

CAPA tag:
PIPELINING

Arguments:
none

Added commands:
none

Standard commands affected:
all

Announced states / possible differences:
both / no

Commands valid in states:
n/a

Specification reference:
this document

Discussion:
The PIPELINING capability indicates the server is capable of accepting multiple commands at a time; the client does not have to wait for the response to a command before issuing a subsequent command. If a server supports PIPELINING, it MUST process each command in turn. If a client uses PIPELINING, it MUST keep track of which commands it has outstanding, and match server responses to commands in order. If either the client or server uses blocking writes, it MUST not exceed the window size of the underlying transport layer.

Some POP3 clients have an option to indicate the server supports "Overlapped POP3 commands." This capability removes the need to configure this at the client.

This is roughly synonymous with the ESMTP PIPELINING extension [PIPELINING], however, since SMTP [SMTP] tends to have short commands and responses, the benefit is in grouping multiple commands and sending them as a unit. While there are cases of this in POP (for example, USER and PASS could be batched, multiple RETR and/or DELE commands could be sent as a group), because POP has short commands and sometimes lengthy responses, there is also an advantage is sending new commands while still receiving the response to an earlier command (for example, sending RETR and/or DELE commands while processing a UIDL reply).

6.7. EXPIRE capability

CAPA tag:
EXPIRE

Arguments:
server-guaranteed minimum retention days, or NEVER; optionally followed by USER in AUTHENTICATION state

Added commands:
none

Standard commands affected:
none

Announced states / possible differences:
both / yes

Commands valid in states:
n/a

Specification reference:
this document

Discussion:
While POP3 allows clients to leave messages on the server, RFC 1939 [POP3] warns about the problems that may arise from this, and allows servers to delete messages based on site policy.

The EXPIRE capability avoids the problems mentioned in RFC 1939, by allowing the server to inform the client as to the policy in effect. The argument to the EXPIRE capability indicates the minimum server retention period, in days, for messages on the server.

EXPIRE 0 indicates the client is not permitted to leave mail on the server; when the session enters the UPDATE state the server MAY assume an implicit DELE for each message which was downloaded with RETR.

EXPIRE NEVER asserts that the server does not delete messages.

The concept of a "retention period" is intentionally vague. Servers may start counting days to expiration when a message is added to a maildrop, when a client becomes aware of the existence of a message through the LIST or UIDL commands, when a message has been acted upon in some way (for example, TOP or RETR), or at some other event. The EXPIRE capability cannot provide a precise indication as to exactly when any specific message will expire. The capability is intended to make it easier for clients to behave in ways which conform to site policy and user wishes. For example, a client might display a warning for attempts to configure a "leave mail on server" period which is greater than or equal to some percentage of the value announced by the server.

If a site uses any automatic deletion policy, it SHOULD use the EXPIRE capability to announce this.

The EXPIRE capability, with a parameter other than 0 or NEVER, is intended to let the client know that the server does permit mail to be left on the server, and to present a value which is the smallest which might be in force.

Sites which permit users to retain messages indefinitely SHOULD announce this with the EXPIRE NEVER response.

If the expiration policy differs per user (that is, the EXPIRE argument might change after authentication), the server MUST announce in AUTHENTICATION state the smallest value which could be set for any user. This might be the smallest value currently in use for any user (so only one value per server), or even the smallest value which the server permits to be set for any user. The server SHOULD append the token "USER" to the EXPIRE parameter in AUTHENTICATION state, to inform the client that a more accurate value is available after authentication. The server SHOULD announce the more accurate value in TRANSACTION state. (The "USER" token allows the client to decide if a second CAPA command is needed or not.)

A site may have a message expiration policy which treats messages differently depending on which user actions have been performed, or based on other factors. For example, a site might delete unseen messages after 60 days, and completely- or partially-seen messages after 15 days.

The announced EXPIRE value is the smallest retention period which is or might be used by any category or condition of the current site policy, for any user (in AUTHENTICATION state) or the specific user (in TRANSACTION state). That is, EXPIRE informs the client of the minimum number of days messages may remain on the server under any circumstances.

Examples:

```
EXPIRE 5 USER
EXPIRE 30
EXPIRE NEVER
EXPIRE 0
```

The first example indicates the server might delete messages after five days, but the period differs per user, and so a more accurate value can be obtained by issuing a second CAPA command in TRANSACTION state. The second example indicates the server could delete messages after 30 days. In the third example, the server announces it does not delete messages. The fourth example specifies that the site does not permit messages to be left on the server.

6.8. UIDL capability

CAPA tag:
UIDL

Arguments:
none

Added commands:
UIDL

Standard commands affected:
none

Announced states / possible differences:
both / no

Commands valid in states:
TRANSACTION

Specification reference:
[POP3]

Discussion:
The UIDL capability indicates that the optional UIDL command is supported.

6.9. IMPLEMENTATION capability

CAPA tag:
IMPLEMENTATION

Arguments:
string giving server implementation information

Added commands:
none

Standard commands affected:
none

Announced states / possible differences:
both (optionally TRANSACTION only) / no

Commands valid in states:
n/a

Specification reference:
this document

Discussion:
It is often useful to identify an implementation of a particular server (for example, when logging). This is commonly done in the welcome banner, but one must guess if a string is an implementation ID or not.

The argument to the IMPLEMENTATION capability consists of one or more tokens which identify the server. (Note that since CAPA response tag arguments are space-separated, it may be convenient for the IMPLEMENTATION capability argument to not contain spaces, so that it is a single token.)

Normally, servers announce IMPLEMENTATION in both states. However, a server MAY chose to do so only in TRANSACTION state.

A server MAY include the implementation identification both in the welcome banner and in the IMPLEMENTATION capability.

Clients MUST NOT modify their behavior based on the server implementation. Instead the server and client should agree on a private extension.

7. Future Extensions to POP3

Future extensions to POP3 are in general discouraged, as POP3's usefulness lies in its simplicity. POP3 is intended as a download-and-delete protocol; mail access capabilities are available in IMAP [IMAP4]. Extensions which provide support for additional mailboxes, allow uploading of messages to the server, or which deviate from POP's download-and-delete model are strongly discouraged and unlikely to be permitted on the IETF standards track.

Clients MUST NOT require the presence of any extension for basic functionality, with the exception of the authentication commands (APOP, AUTH [section 6.3] and USER/PASS).

Section 9 specifies how additional capabilities are defined.

8. Extended POP3 Response Codes

Unextended POP3 is only capable of indicating success or failure to most commands. Unfortunately, clients often need to know more information about the cause of a failure in order to gracefully recover. This is especially important in response to a failed login

(there are widely-deployed clients which attempt to decode the error text of a PASS command result, to try and distinguish between "unable to get maildrop lock" and "bad login").

This specification amends the POP3 standard to permit an optional response code, enclosed in square brackets, at the beginning of the human readable text portion of an "+OK" or "-ERR" response. Clients supporting this extension MAY remove any information enclosed in square brackets prior to displaying human readable text to the user. Immediately following the open square bracket "[" character is a response code which is interpreted in a case-insensitive fashion by the client.

The response code is hierarchical, with a "/" separating levels of detail about the error. Clients MUST ignore unknown hierarchical detail about the response code. This is important, as it could be necessary to provide further detail for response codes in the future.

Section 3 describes response codes using [ABNF].

If a server supports extended response codes, it indicates this by including the RESP-CODES capability in the CAPA response.

Examples:

```
C: APOP mrose c4c9334bac560ecc979e58001b3e22fb
S: -ERR [IN-USE] Do you have another POP session running?
```

8.1. Initial POP3 response codes

This specification defines two POP3 response codes which can be used to determine the reason for a failed login. Section 9 specifies how additional response codes are defined.

8.1.1. The LOGIN-DELAY response code

This occurs on an -ERR response to an AUTH, USER (see note), PASS or APOP command and indicates that the user has logged in recently and will not be allowed to login again until the login delay period has expired.

NOTE: Returning the LOGIN-DELAY response code to the USER command avoids the work of authenticating the user but reveals to the client that the specified user exists. Unless the server is operating in an environment where user names are not secret (for example, many popular email clients advertise the POP server and user name in an outgoing mail header), or where server access is restricted, or the server can verify that the connection is to the same user, it is

strongly recommended that the server not issue this response code to the USER command. The server still saves the cost of opening the maildrop, which in some environments is the most expensive step.

8.1.2. The IN-USE response code

This occurs on an -ERR response to an AUTH, APOP, or PASS command. It indicates the authentication was successful, but the user's maildrop is currently in use (probably by another POP3 client).

9. IANA Considerations

This document requests that IANA maintain two new registries: POP3 capabilities and POP3 response codes.

New POP3 capabilities MUST be defined in a standards track or IESG approved experimental RFC, and MUST NOT begin with the letter "X".

New POP3 capabilities MUST include the following information:

- CAPA tag
- Arguments
- Added commands
- Standard commands affected
- Announced states / possible differences
- Commands valid in states
- Specification reference
- Discussion

In addition, new limits for POP3 command and response lengths may need to be included.

New POP3 response codes MUST be defined in an RFC or other permanent and readily available reference, in sufficient detail so that interoperability between independent implementations is possible. (This is the "Specification Required" policy described in [IANA]).

New POP3 response code specifications MUST include the following information: the complete response code, for which responses (+OK or -ERR) and commands it is valid, and a definition of its meaning and expected client behavior.

10. Security Considerations

A capability list can reveal information about the server's authentication mechanisms which can be used to determine if certain attacks will be successful. However, allowing clients to automatically detect availability of stronger mechanisms and alter their configurations to use them can improve overall security at a site.

Section 8.1 discusses the security issues related to use of the LOGIN-DELAY response code with the USER command.

11. Acknowledgments

This document has been revised in part based on comments and discussions which took place on and off the IETF POP3 Extensions mailing list. The help of those who took the time to review this memo and make suggestions is appreciated, especially that of Alexey Melnikov, Harald Alvestrand, and Mike Gahrns.

12. References

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

[IANA] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[IMAP4] Crispin, M., "Internet Message Access Protocol -- Version 4rev1", RFC 2060, December 1996.

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

[PIPELINING] Freed, N., "SMTP Service Extension for Command Pipelining", RFC 2197, September 1997.

[POP3] Myers, J. and M. Rose, "Post Office Protocol -- Version 3", STD 53, RFC 1939, May 1996.

[POP-AUTH] Myers, J., "POP3 AUTHentication command", RFC 1734, December 1994.

[SASL] Myers, J., "Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

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

13. Authors' Addresses

Randall Gellens
QUALCOMM Incorporated
6455 Lusk Blvd.
San Diego, CA 92121-2779
USA

Phone: +1 619 651 5115
Fax: +1 619 845 7268
EMail: randy@qualcomm.com

Chris Newman
Innosoft International, Inc.
1050 Lakes Drive
West Covina, CA 91790
USA

EMail: chris.newman@innosoft.com

Laurence Lundblade
QUALCOMM Incorporated
6455 Lusk Blvd.
San Diego, Ca, 92121-2779
USA

Phone: +1 619 658 3584
Fax: +1 619 845 7268
EMail: lgl@qualcomm.com

14. Full Copyright Statement