Network Working Group
Request for Comments: 5034
Obsoletes: 1734
Updates: 2449
Category: Standards Track

R. Siemborski
Google, Inc.
A. Menon-Sen
Oryx Mail Systems GmbH

July 2007

The Post Office Protocol (POP3) Simple Authentication and Security Layer (SASL) Authentication Mechanism

Status of This Memo

Copyright Notice

Abstract

This document defines a profile of the Simple Authentication and Security Layer (SASL) for the Post Office Protocol (POP3). This extension allows a POP3 client to indicate an authentication mechanism to the server, perform an authentication protocol exchange, and optionally negotiate a security layer for subsequent protocol interactions during this session.

This document seeks to consolidate the information related to POP3 AUTH into a single document. To this end, this document obsoletes and replaces RFC 1734, and updates the information contained in Section 6.3 of RFC 2449.

1. Introduction

The POP3 (see [RFC1939]) AUTH command (see [RFC1734]) has suffered several problems in its specification. The first is that it was very similar to a SASL framework defined by [RFC4422], but pre-dated the initial SASL specification. It was therefore missing some key components, such as a way to list the available authentication mechanisms.

Later, [RFC2449] attempted to remedy this situation by adding the CAPA command and allowing an initial client response with the AUTH command, but problems remained in the clarity of the specification of how the initial client response was to be handled.

Together, this means creating a full POP3 AUTH implementation requires an understanding of material in at least five different documents (and [RFC3206] provides additional response codes that are useful during authentication).

This document attempts to combine the information in [RFC1734] and [RFC2449] to simplify this situation. Additionally, it aims to clarify and update the older specifications where appropriate.

2. Conventions Used in This Document

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 [RFC2119].

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

Formal syntax is defined by [RFC4234].

3. The SASL Capability

This section supersedes the definition of the SASL Capability in section 6.3 of [RFC2449].

CAPA tag:
   SASL

Arguments:
   Supported SASL Mechanisms

Added commands:
   AUTH

Standard commands affected:
None

Announced states / possible differences:
both / no

Commands valid in states:
AUTHORIZATION

Specification reference:
This document and [RFC4422]

Discussion:
The SASL capability permits the use of the AUTH command (as defined in Section 4 of this document) to begin a SASL negotiation (as defined in [RFC4422]). The argument to the SASL capability is a space-separated list of SASL mechanisms that are supported.

If a server either does not support the CAPA command or does not advertise the SASL capability, clients SHOULD NOT attempt the AUTH command. If a client does attempt the AUTH command in such a situation, it MUST NOT supply the client initial response parameter (for backwards compatibility with [RFC1734]).

Note that the list of available mechanisms MAY change after a successful STLS command (see [RFC2595]). However, as required by [RFC2449], implementations MUST continue to include the SASL capability even after a successful AUTH command has been completed (even though no further AUTH commands may be issued).

Example
```
S: +OK pop.example.com BlurdyBlurp POP3 server ready
C: CAPA
S: +OK List of capabilities follows
S: SASL PLAIN DIGEST-MD5 GSSAPI ANONYMOUS
S: STLS
S: IMPLEMENTATION BlurdyBlurp POP3 server
S: .
```

4. The AUTH Command

AUTH mechanism [initial-response]

Arguments:

mechanism: A string identifying a SASL authentication mechanism.

initial-response: An optional initial client response, as defined in Section 3 of [RFC4422]. If present, this response MUST be encoded as Base64 (specified in Section 4 of [RFC4648]), or consist only of the single character "=", which represents an empty initial response.

Restrictions:

After an AUTH command has been successfully completed, no more AUTH commands may be issued in the same session. After a successful AUTH command completes, a server MUST reject any further AUTH commands with an -ERR reply.

The AUTH command may only be given during the AUTHORIZATION state.

Discussion:

The AUTH command initiates a SASL authentication exchange between the client and the server. The client identifies the SASL mechanism to use with the first parameter of the AUTH command. If the server supports the requested authentication mechanism, it performs the SASL exchange to authenticate the user. Optionally, it also negotiates a security layer for subsequent protocol interactions during this session. If the requested authentication mechanism is not supported, the server rejects the AUTH command with an -ERR reply.

The authentication protocol exchange consists of a series of server challenges and client responses that are specific to the chosen SASL mechanism.

A server challenge is sent as a line consisting of a "+" character, followed by a single space and a string encoded using Base64, as specified in Section 4 of [RFC4648]. This line MUST NOT contain any text other than the BASE64-encoded challenge.

A client response consists of a line containing a string encoded as Base64. If the client wishes to cancel the authentication exchange, it issues a line with a single "*". If the server receives such a response, it MUST reject the AUTH command by sending an -ERR reply.

The optional initial-response argument to the AUTH command is used to save a round trip when using authentication mechanisms that support an initial client response. If the initial response argument is omitted and the chosen mechanism requires

an initial client response, the server MUST proceed by issuing an empty challenge, as defined in Section 3 of [RFC4422]. In POP3, an empty server challenge is defined as a line with only a "+", followed by a single space. It MUST NOT contain any other data.

For the purposes of the initial client response, the 255-octet limit on the length of a single command, defined in Section 4 of [RFC2449], still applies. If specifying an initial response would cause the AUTH command to exceed this length, the client MUST NOT use the initial-response parameter (and must proceed instead by sending its initial response after an empty challenge from the server, as in Section 3 of [RFC4422]).

If the client needs to send a zero-length initial response, it MUST transmit the response as a single equals sign ("="). This indicates that the response is present, but contains no data.

If the client uses an initial-response argument to the AUTH command with a SASL mechanism that does not support an initial client send, the server MUST reject the AUTH command with an -ERR reply.

If the server cannot Base64 decode a client response, it MUST reject the AUTH command with an -ERR reply. If the client cannot Base64 decode any of the server's challenges, it MUST cancel the authentication using the "*" response. In particular, servers and clients MUST reject (and not ignore) any character not explicitly allowed by the Base64 alphabet, and MUST reject any sequence of Base64 characters that contains the pad character ('=') anywhere other than the end of the string (e.g., "=AAA" and "AAA=BBB" are not allowed).

Excepting the initial client response, these BASE64 strings may be of arbitrary length, depending on the authentication mechanism in use. Clients and servers MUST be able to handle the largest encoded challenges and responses generated by the authentication mechanisms they support. This requirement is independent of any line-length limitations the client or server may have in other parts of its protocol implementation.

If the server is unable to authenticate the client, it MUST reject the AUTH command with an -ERR reply. Should the client successfully complete the exchange, the server issues a +OK reply. Additionally, upon success, the POP3 session enters the TRANSACTION state.

The authorization identity generated by the SASL exchange is a simple username, and SHOULD use the SASLprep profile (see [RFC4013]) of the StringPrep algorithm (see [RFC3454]) to prepare these names for matching. If preparation of the authorization identity fails or results in an empty string (unless it was transmitted as the empty string), the server MUST fail the authentication.

If a security layer is negotiated during the SASL exchange, it takes effect for the client on the octet immediately following the CRLF that concludes the last response generated by the client. For the server, it takes effect immediately following the CRLF of its success reply.

When a security layer takes effect, the server MUST discard any knowledge previously obtained from the client, which was not obtained from the SASL negotiation itself. Likewise, the client MUST discard any knowledge obtained from the server, such as the list of available POP3 service extensions.

When both Transport Layer Security (TLS) (see [RFC4346]) and SASL security layers are in effect, the TLS encoding MUST be applied after the SASL encoding when sending data. (According to [RFC2595], STLS can only be issued before AUTH in any case.)

Note that POP3 does not allow for additional data to be sent with a message indicating a successful outcome (see Section 3.6 of [RFC4422]).

The service name specified by this protocol's profile of SASL is "pop".

If an AUTH command fails, the client may try another authentication mechanism or present different credentials by issuing another AUTH command (or by using one of the other POP3 authentication mechanisms). Likewise, the server MUST behave as if the client had not issued the AUTH command.

To ensure interoperability, client and server implementations of this extension MUST implement the PLAIN SASL mechanism [RFC4616] running over TLS [RFC2595].

A server implementation MUST implement a configuration in which it does NOT advertise or permit any plaintext password mechanisms, unless the STLS command has been used to negotiate a TLS session (see [RFC2595]). As described by RFC 4616, this configuration SHOULD be the default configuration. Before using a plaintext password mechanism over a TLS session, client

implementations MUST verify the TLS server certificate as required by RFC 2595, Section 2.4. Client and server implementations SHOULD implement additional SASL mechanisms that do not send plaintext passwords, such as the GSSAPI [RFC4752] mechanism.

5. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form notation as specified in [RFC4234]. The rules CRLF, ALPHA, and DIGIT are imported from [RFC4234]. The sasl-mech rule is from [RFC4422].

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper- or lower-case characters to define token strings is for editorial clarity only. Implementations MUST accept these strings in a case-insensitive fashion.

```
auth-command     = "AUTH" SP sasl-mech [SP initial-response]
                   *(CRLF [base64]) [CRLF cancel-response] CRLF

initial-response = base64 / "="

cancel-response  = "*"

base64           = base64-terminal /
                   ( 1*(4base64-CHAR) [base64-terminal] )

base64-char      = ALPHA / DIGIT / "+" / "/"
                   ;; Case-sensitive

base64-terminal  = (2base64-char "==") / (3base64-char "=")

continue-req     = "+" SP [base64] CRLF
```

Additionally, the ABNF specified in [RFC2449] is updated as follows:

```
response         =/ continue-req
```

6. Examples

Here is an example of a client attempting AUTH PLAIN (see [RFC4616]) under TLS and making use of the initial client response:

```
S: +OK pop.example.com BlurdyBlurp POP3 server ready
C: CAPA
S: +OK List of capabilities follows
S: SASL DIGEST-MD5 GSSAPI ANONYMOUS
S: STLS
S: IMPLEMENTATION BlurdyBlurp POP3 server
S: .
C: STLS
S: +OK Begin TLS negotiation now
    (TLS negotiation proceeds, further commands protected by TLS
    layer)
C: CAPA
S: +OK List of capabilities follows
S: SASL PLAIN DIGEST-MD5 GSSAPI ANONYMOUS
S: IMPLEMENTATION BlurdyBlurp POP3 server
S: .
C: AUTH PLAIN dGVzdAB0ZXN0AHRlc3Q=
S: +OK Maildrop locked and ready
```

Here is another client that is attempting AUTH PLAIN under a TLS layer, this time without the initial response. Parts of the negotiation before the TLS layer was established have been omitted:

```
    (TLS negotiation proceeds, further commands protected by TLS
    layer)
C: CAPA
S: +OK List of capabilities follows
S: SASL PLAIN DIGEST-MD5 GSSAPI ANONYMOUS
S: IMPLEMENTATION BlurdyBlurp POP3 server
S: .
C: AUTH PLAIN
    (note that there is a space following the '+' on the
    following line)
S: +
C: dGVzdAB0ZXN0AHRlc3Q=
S: +OK Maildrop locked and ready
```

Here is an example using a mechanism in which the exchange begins with a server challenge (the long lines are broken for editorial clarity only):

```
S: +OK pop.example.com BlurdyBlurp POP3 server ready
C: CAPA
S: +OK List of capabilities follows
S: SASL DIGEST-MD5 GSSAPI ANONYMOUS
S: STLS
S: IMPLEMENTATION BlurdyBlurp POP3 server
S: .
C: AUTH DIGEST-MD5
S: + cmVhbG09ImVsd29vZC5pbm5vc29mdC5jb20iLG5vbmNlPSJPQTZNRzl0
     RVFHbTJoaCIscW9wPSJhdXRoIixhbGdvcml0aG09bWQ1LXNlc3MsY2hh
     cnNldD11dGYtOA==
C: Y2hhcnNldD11dGYtOCx1c2VybmFtZT0iY2hyaXMiLHJlYWxtPSJlbHdvb2
   QuaW5ub3NvZnQuY29tIixub25jZT0iT0E2TUc5dEVRR20yaGgiLG5jPTAw
   MDAwMDAxLGNub25jZT0iT0E2TUhYaDZWcVRyUmsiLGRpZ2VzdC11cmk9In
   BvcC9lbHdvb2QuaW5ub3NvZnQuY29tIixyZXNwb25zZT1iMGQ1NmQyZjA1
   NGMyNGI2MjA3MjMyMjEwNjQ2OGRiOSxxb3A9YXV0aA==
S: + cnNwYXV0aD0wYjk3MTQ2MmNlZjVlOGY5MzBkYjlhMzNiMDJmYzlhMA==
C:
S: +OK Maildrop locked and ready
```

7. Security Considerations

Security issues are discussed throughout this document.

8. IANA Considerations

The IANA has updated its site to refer to this RFC instead of [RFC1734] in http://www.iana.org/assignments/pop3-extension-mechanism (the POP3 extension registry), and also in http://www.iana.org/assignments/gssapi-service-names (the GSSAPI/SASL service name registry).

9. Acknowledgments

The authors would like to acknowledge the contributions of John Myers, Randall Gellens, Chris Newman, Laurence Lundblade, and other contributors to RFC 1734 and RFC 2554, on which this document draws heavily.

The authors would also like to thank Ken Murchison, Randall Gellens, Alexey Melnikov, Mark Crispin, Arnt Gulbrandsen, Lisa Dusseault, Frank Ellermann, and Philip Guenther for their reviews of this document.

10. Changes From RFC 1734, RFC 2449.

1. Updated references to newer versions of various specifications, particularly RFC 4422.

2. The SASL-based semantics defined in RFC 2449 are now normative for the AUTH extension.

3. The proper behaviour and handling of initial client responses is defined, with examples and references to SASL.

4. New minimum requirement of support for TLS+PLAIN.

5. The SASLprep profile SHOULD be used to prepare authorization identities.

6. Clarify that the TLS encoding should be applied after any encoding applied by SASL security layers.

7. Note that the mechanism list can change after STLS.

8. Explicitly mention that "=" means a zero-length initial response.

9. Note that POP3 doesn’t allow additional data to be sent with +OK.

11. Normative References

[RFC1939] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996.

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

[RFC2449] Gellens, R., Newman, C., and L. Lundblade, "POP3 Extension Mechanism", RFC 2449, November 1998.

[RFC2595] Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595, June 1999.

[RFC3454] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")", RFC 3454, December 2002.

[RFC4013] Zeilenga, K., "SASLprep: Stringprep Profile for User Names and Passwords", RFC 4013, February 2005.

[RFC4234] Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[RFC4422] Melnikov, A., Ed., and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC4616] Zeilenga, K., Ed., "The PLAIN Simple Authentication and Security Layer (SASL) Mechanism", RFC 4616, August 2006.

12. Informative References

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

[RFC3206] Gellens, R., "The SYS and AUTH POP Response Codes", RFC 3206, February 2002.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC4752] Melnikov, A., Ed., "The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism", RFC 4752, November 2006.

Authors' Addresses

Robert Siemborski
Google, Inc.
1600 Ampitheatre Parkway
Mountain View, CA 94043

Phone: +1 650 623 6925
EMail: robsiemb@google.com

Abhijit Menon-Sen
Oryx Mail Systems GmbH

EMail: ams@oryx.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.