Network Working Group
Request for Comments: 5464
Category: Standards Track

C. Daboo
Apple, Inc.

February 2009

The IMAP METADATA Extension

Status of This Memo

Abstract

The METADATA extension to the Internet Message Access Protocol permits clients and servers to maintain "annotations" or "metadata" on IMAP servers. It is possible to have annotations on a per-mailbox basis or on the server as a whole. For example, this would allow comments about the purpose of a particular mailbox to be "attached" to that mailbox, or a "message of the day" containing server status information to be made available to anyone logging in to the server.

Table of Contents

1. Introduction and Overview

The goal of the METADATA extension is to provide a means for clients to set and retrieve "annotations" or "metadata" on an IMAP server. The annotations can be associated with specific mailboxes or the server as a whole. The server can choose to support only server annotations or both server and mailbox annotations.

A server that supports both server and mailbox annotations indicates the presence of this extension by returning "METADATA" as one of the supported capabilities in the CAPABILITY command response.

A server that supports only server annotations indicates the presence of this extension by returning "METADATA-SERVER" as one of the supported capabilities in the CAPABILITY command response.

A server that supports unsolicited annotation change responses MUST support the "ENABLE" [RFC5161] extension to allow clients to turn that feature on.

The METADATA extension adds two new commands and one new untagged response to the IMAP base protocol.

This extension makes the following changes to the IMAP protocol:

- adds a new SETMETADATA command
- adds a new GETMETADATA command
- adds a new METADATA untagged response
- adds a new METADATA response code

The rest of this document describes the data model and protocol changes more rigorously.

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 [RFC2119].

Whitespace and line breaks have been added to the examples in this document to promote readability.

3. Data Model

3.1. Overview

Mailboxes or the server as a whole may have zero or more annotations associated with them. An annotation contains a uniquely named entry, which has a value. Annotations can be added to mailboxes when a mailbox name is provided as the first argument to the SETMETADATA command, or to the server as a whole when the empty string is provided as the first argument to the command.

For example, a general comment being added to a mailbox may have an entry name of "/comment" and a value of "Really useful mailbox".

The protocol changes to IMAP described below allow a client to access or change the values of any annotation entry, assuming it has sufficient access rights to do so.

3.2. Namespace of Entries

Each annotation is an entry that has a hierarchical name, with each component of the name separated by a slash ("/"). An entry name MUST NOT contain two consecutive "/" characters and MUST NOT end with a "/" character.

The value of an entry is NIL (has no value), or a string or binary data of zero or more octets. A string MAY contain multiple lines of text. Clients MUST use the CRLF (0x0D 0x0A) character octet sequence to represent line ends in a multi-line string value.

Entry names MUST NOT contain asterisk ("*") or percent ("%") characters and MUST NOT contain non-ASCII characters or characters with octet values in the range 0x00 to 0x19. Invalid entry names result in a BAD response in any IMAP command in which they are used.

Entry names are case-insensitive.

Use of control or punctuation characters in entry names is strongly discouraged.

This specification defines an initial set of entry names available for use with mailbox and server annotations. In addition, an extension mechanism is described to allow additional names to be added for extensibility.

The first component in entry names defines the scope of the annotation. Currently, only the prefixes "/private" or "/shared" are defined. These prefixes are used to indicate whether an annotation

is stored on a per-user basis ("/private") and not visible to other users, or whether an annotation is shared between authorized users ("/shared") with a single value that can be read and changed by authorized users with appropriate access. See Section 3.3 for details.

Entry names can have any number of components starting at 2, unless they fall under the vendor namespaces (i.e., have a /shared/vendor/<vendor-token> or /private/vendor/<vendor-token> prefix as described below), in which case they have at least 4 components.

3.2.1. Entry Names

Entry names MUST be specified in a Standards Track or IESG-approved Experimental RFC, or fall under the vendor namespace. See Section 6.1 for the registration template.

3.2.1.1. Server Entries

These entries are set or retrieved when the mailbox name argument to the new SETMETADATA or GETMETADATA command is the empty string.

/shared/comment

Defines a comment or note that is associated with the server and that is shared with authorized users of the server.

/shared/admin

Indicates a method for contacting the server administrator. The value MUST be a URI (e.g., a mailto: or tel: URL). This entry is always read-only -- clients cannot change it. It is visible to authorized users of the system.

/shared/vendor/<vendor-token>

Defines the top level of shared entries associated with the server, as created by a particular product of some vendor. This entry can be used by vendors to provide server- or client-specific annotations. The vendor-token MUST be registered with IANA, using the Application Configuration Access Protocol (ACAP) [RFC2244] vendor subtree registry.

/private/vendor/<vendor-token>

Defines the top level of private entries associated with the server, as created by a particular product of some vendor. This entry can be used by vendors to provide server- or client-specific

annotations. The vendor-token MUST be registered with IANA, using the ACAP [RFC2244] vendor subtree registry.

3.2.1.2. Mailbox Entries

These entries are set or retrieved when the mailbox name argument to the new SETMETADATA or GETMETADATA command is not the empty string.

/shared/comment

Defines a shared comment or note associated with a mailbox.

/private/comment

Defines a private (per-user) comment or note associated with a mailbox.

/shared/vendor/<vendor-token>

Defines the top level of shared entries associated with a specific mailbox, as created by a particular product of some vendor. This entry can be used by vendors to provide client-specific annotations. The vendor-token MUST be registered with IANA, using the ACAP [RFC2244] vendor subtree registry.

/private/vendor/<vendor-token>

Defines the top level of private entries associated with a specific mailbox, as created by a particular product of some vendor. This entry can be used by vendors to provide client-specific annotations. The vendor-token MUST be registered with IANA, using the ACAP [RFC2244] vendor subtree registry.

3.3. Private versus Shared and Access Control

In the absence of the ACL (Access Control List) extension [RFC4314], users can only set and retrieve private or shared mailbox annotations on a mailbox that exists and is returned to them via a LIST or LSUB command, and on which they have either read or write access to the actual message content of the mailbox (as determined by the READ-ONLY and READ-WRITE response codes as described in Section 5.2 of [RFC4314]).

When the ACL extension [RFC4314] is present, users can only set and retrieve private or shared mailbox annotations on a mailbox on which they have the "l" right and any one of the "r", "s", "w", "i", or "p" rights.

If a client attempts to set or retrieve annotations on mailboxes that do not satisfy the conditions above, the server MUST respond with a NO response.

Users can always retrieve private or shared server annotations if they exist. Servers MAY restrict the creation of private or shared server annotations as appropriate. When restricted, the server MUST return a NO response when the SETMETADATA command is used to try to create a server annotation.

If the METADATA extension is present, support for shared annotations is REQUIRED, whilst support for private annotations is OPTIONAL. This recognizes the fact that support for private annotations may introduce significantly more complexity to a server in terms of tracking ownership of the annotations, how quota is determined for users based on their own annotations, etc.

4. IMAP Protocol Changes

4.1. General Considerations

The new SETMETADATA command and the METADATA response each have a mailbox name argument. An empty string is used for the mailbox name to signify server annotations. A non-empty string is used to signify mailbox annotations attached to the corresponding mailbox.

Servers SHOULD ensure that mailbox annotations are automatically moved when the mailbox they refer to is renamed, i.e., the annotations follow the mailbox. This applies to a rename of the INBOX, with the additional behavior that the annotations are copied from the original INBOX to the renamed mailbox, i.e., mailbox annotations are preserved on the INBOX when it is renamed.

Servers SHOULD delete annotations for a mailbox when the mailbox is deleted, so that a mailbox created with the same name as a previously existing mailbox does not inherit the old mailbox annotations.

Servers SHOULD allow annotations on all 'types' of mailboxes, including ones reporting \Noselect for their LIST response. Servers can implicitly remove \Noselect mailboxes when all child mailboxes are removed, and, at that time any annotations associated with the \Noselect mailbox SHOULD be removed.

The server is allowed to impose limitations on the size of any one annotation or the total number of annotations for a single mailbox or for the server as a whole. However, the server MUST accept an annotation data size of at least 1024 bytes, and an annotation count per server or mailbox of at least 10.

Some annotations may be "read-only" -- i.e., they are set by the server and cannot be changed by the client. Also, such annotations may be "computed" -- i.e., the value changes based on underlying properties of the mailbox or server. For example, an annotation reporting the total size of all messages in the mailbox would change as messages are added or removed. Or, an annotation containing an IMAP URL for the mailbox would change if the mailbox was renamed.

Servers MAY support sending unsolicited responses for use when annotations are changed by some "third-party" (see Section 4.4). In order to do so, servers MUST support the ENABLE command [RFC5161] and MUST only send unsolicited responses if the client used the ENABLE command [RFC5161] extension with the capability string "METADATA" or "METADATA-SERVER" earlier in the session, depending on which of those capabilities is supported by the server.

4.2. GETMETADATA Command

This extension adds the GETMETADATA command. This allows clients to retrieve server or mailbox annotations.

This command is only available in authenticated or selected state [RFC3501].

```
Arguments:  mailbox-name
            options
            entry-specifier

Responses:  required METADATA response

Result:     OK - command completed
            NO - command failure: can't access annotations on
                 the server
            BAD - command unknown or arguments invalid
```

When the mailbox name is the empty string, this command retrieves server annotations. When the mailbox name is not empty, this command retrieves annotations on the specified mailbox.

Options MAY be included with this command and are defined below.

Example:

```
C: a GETMETADATA "" /shared/comment
S: * METADATA "" (/shared/comment "Shared comment")
S: a OK GETMETADATA complete
```

In the above example, the contents of the value of the "/shared/comment" server entry is requested by the client and returned by the server.

Example:

```
C: a GETMETADATA "INBOX" /private/comment
S: * METADATA "INBOX" (/private/comment "My own comment")
S: a OK GETMETADATA complete
```

In the above example, the contents of the value of the "/private/comment" mailbox entry for the mailbox "INBOX" is requested by the client and returned by the server.

Entry specifiers can be lists of atomic specifiers, so that multiple annotations may be returned in a single GETMETADATA command.

Example:

```
C: a GETMETADATA "INBOX" (/shared/comment /private/comment)
S: * METADATA "INBOX" (/shared/comment "Shared comment"
                       /private/comment "My own comment")
S: a OK GETMETADATA complete
```

In the above example, the values of the two server entries "/shared/comment" and "/private/comment" on the mailbox "INBOX" are requested by the client and returned by the server.

4.2.1. MAXSIZE GETMETADATA Command Option

When the MAXSIZE option is specified with the GETMETADATA command, it restricts which entry values are returned by the server. Only entry values that are less than or equal in octet size to the specified MAXSIZE limit are returned. If there are any entries with values larger than the MAXSIZE limit, the server MUST include the METADATA LONGENTRIES response code in the tagged OK response for the GETMETADATA command. The METADATA LONGENTRIES response code returns the size of the biggest entry value requested by the client that exceeded the MAXSIZE limit.

Example:

```
C: a GETMETADATA "INBOX" (MAXSIZE 1024)
                         (/shared/comment /private/comment)
S: * METADATA "INBOX" (/private/comment "My own comment")
S: a OK [METADATA LONGENTRIES 2199] GETMETADATA complete
```

In the above example, the values of the two server entries "/shared/comment" and "/private/comment" on the mailbox "INBOX" are requested by the client, which wants to restrict the size of returned values to 1024 octets. In this case, the "/shared/comment" entry value is 2199 octets and is not returned.

4.2.2. DEPTH GETMETADATA Command Option

When the DEPTH option is specified with the GETMETADATA command, it extends the list of entry values returned by the server. For each entry name specified in the GETMETADATA command, the server returns the value of the specified entry name (if it exists), plus all entries below the entry name up to the specified DEPTH. Three values are allowed for DEPTH:

"0" - no entries below the specified entry are returned
"1" - only entries immediately below the specified entry are returned
"infinity" - all entries below the specified entry are returned

Thus, "depth 1" for an entry "/a" will match "/a" as well as its children entries (e.g., "/a/b"), but will not match grandchildren entries (e.g., "/a/b/c").

If the DEPTH option is not specified, this is the same as specifying "DEPTH 0".

Example:

```
C: a GETMETADATA "INBOX" (DEPTH 1)
                         (/private/filters/values)
S: * METADATA "INBOX" (/private/filters/values/small
     "SMALLER 5000" /private/filters/values/boss
     "FROM \"boss@example.com\"")
S: a OK GETMETADATA complete
```

In the above example, 2 entries below the /private/filters/values entry exist on the mailbox "INBOX": "/private/filters/values/small" and "/private/filters/values/boss".

4.3. SETMETADATA Command

This extension adds the SETMETADATA command. This allows clients to set annotations.

This command is only available in authenticated or selected state [RFC3501].

```
      Arguments:  mailbox-name
                  entry
                  value
                  list of entry, values

      Responses:  no specific responses for this command

      Result:     OK - command completed
                  NO - command failure: can't set annotations,
                       or annotation too big or too many
                  BAD - command unknown or arguments invalid
```

This command sets the specified list of entries by adding or replacing the specified values provided, on the specified existing mailboxes or on the server (if the mailbox argument is the empty string). Clients can use NIL for the value of entries it wants to remove. The server SHOULD NOT return a METADATA response containing the updated annotation data. Clients MUST NOT assume that a METADATA response will be sent, and MUST assume that if the command succeeds, then the annotation has been changed.

If the server is unable to set an annotation because the size of its value is too large, the server MUST return a tagged NO response with a "[METADATA MAXSIZE NNN]" response code when NNN is the maximum octet count that it is willing to accept.

If the server is unable to set a new annotation because the maximum number of allowed annotations has already been reached, the server MUST return a tagged NO response with a "[METADATA TOOMANY]" response code.

If the server is unable to set a new annotation because it does not support private annotations on one of the specified mailboxes, the server MUST return a tagged NO response with a "[METADATA NOPRIVATE]" response code.

When any one annotation fails to be set, resulting in a tagged NO response from the server, then the server MUST NOT change the values for other annotations specified in the SETMETADATA command.

Example:

```
C: a SETMETADATA INBOX (/private/comment {33}
S: + ready for data
My new comment across
two lines.
)
S: a OK SETMETADATA complete
```

In the above example, the entry "/private/comment" for the mailbox "INBOX" is created (if not already present) and the value set to a multi-line string.

Example:

```
C: a SETMETADATA INBOX (/private/comment NIL)
S: a OK SETMETADATA complete
```

In the above example, the entry "/private/comment" is removed from the mailbox "INBOX".

Multiple entries can be set in a single SETMETADATA command by listing entry-value pairs in the list.

Example:

```
C: a SETMETADATA INBOX (/private/comment "My new comment"
                        /shared/comment "This one is for you!")
S: a OK SETMETADATA complete
```

In the above example, the entries "/private/comment" and "/shared/comment" for the mailbox "INBOX" are created (if not already present) and the values set as specified.

Example:

```
C: a SETMETADATA INBOX (/private/comment "My new comment")
S: a NO [METADATA TOOMANY] SETMETADATA failed
```

In the above example, the server is unable to set the requested (new) annotation as it has reached the limit on the number of annotations it can support on the specified mailbox.

4.4. METADATA Response

The METADATA response displays results of a GETMETADATA command, or can be returned as an unsolicited response at any time by the server in response to a change in a server or mailbox annotation.

When unsolicited responses are activated by the ENABLE [RFC5161] command for this extension, servers MUST send unsolicited METADATA responses if server or mailbox annotations are changed by a third-party, allowing servers to keep clients updated with changes.

Unsolicited METADATA responses MUST only contain entry names, not the values. If the client wants to update any cached values, it must explicitly retrieve those using a GETMETADATA command.

The METADATA response can contain multiple entries in a single response, but the server is free to return multiple responses for each entry or group of entries, if it desires.

This response is only available in authenticated or selected state [RFC3501].

4.4.1. METADATA Response with Values

The response consists of a list of entry-value pairs.

Example:

```
C: a GETMETADATA "" /shared/comment
S: * METADATA "" (/shared/comment "My comment")
S: a OK GETMETADATA complete
```

In the above example, a single entry with its value is returned by the server.

Example:

```
C: a GETMETADATA "INBOX" /private/comment /shared/comment
S: * METADATA "INBOX" (/private/comment "My comment"
                       /shared/comment "Its sunny outside!")
S: a OK GETMETADATA complete
```

In the above example, two entries and their values are returned by the server.

Example:

```
C: a GETMETADATA "INBOX" /private/comment /shared/comment
S: * METADATA "INBOX" (/private/comment "My comment")
S: * METADATA "INBOX" (/shared/comment "Its sunny outside!")
S: a OK GETMETADATA complete
```

In the above example, the server returns two separate responses for each of the two entries requested.

4.4.2. Unsolicited METADATA Response without Values

The response consists of a list of entries, each of which have changed on the server or mailbox.

Example:

```
C: a NOOP
S: * METADATA "" /shared/comment
S: a OK NOOP complete
```

In the above example, the server indicates that the "/shared/comment" server entry has been changed.

Example:

```
C: a NOOP
S: * METADATA "INBOX" /shared/comment /private/comment
S: a OK NOOP complete
```

In the above example, the server indicates a change to two mailbox entries.

5. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [RFC5234].

Non-terminals referenced but not defined below are as defined by [RFC3501], with the new definitions in [RFC4466] superseding those in [RFC3501].

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.

```
capability        =/ "METADATA" / "METADATA-SERVER"
                     ; defines the capabilities for this extension.

command-auth      =/ setmetadata / getmetadata
                     ; adds to original IMAP command

entries           = entry /
                     "(" entry *(SP entry) ")"
                     ; entry specifiers

entry             = astring
                     ; slash-separated path to entry
                     ; MUST NOT contain "*" or "%"

entry-value       = entry SP value

entry-values      = "(" entry-value *(SP entry-value) ")"
```

```
      entry-list        = entry *(SP entry)
                          ; list of entries used in unsolicited
                          ; METADATA response

      getmetadata       = "GETMETADATA" [SP getmetadata-options]
                                        SP mailbox SP entries
                          ; empty string for mailbox implies
                          ; server annotation.

      getmetadata-options = "(" getmetadata-option
                               *(SP getmetadata-option) ")"

      getmetadata-option = tagged-ext-label [SP tagged-ext-val]
                          ; tagged-ext-label and tagged-ext-val
                          ; are defined in [RFC4466].

      maxsize-opt       = "MAXSIZE" SP number
                          ; Used as a getmetadata-option

      metadata-resp     = "METADATA" SP mailbox SP
                          (entry-values / entry-list)
                          ; empty string for mailbox implies
                          ; server annotation.

      response-payload  =/ metadata-resp
                          ; adds to original IMAP data responses

      resp-text-code    =/ "METADATA" SP "LONGENTRIES" SP number
                            ; new response codes for GETMETADATA

      resp-text-code    =/ "METADATA" SP ("MAXSIZE" SP number /
                                          "TOOMANY" / "NOPRIVATE")
                          ; new response codes for SETMETADATA
                          ; failures

      scope-opt         =  "DEPTH" SP ("0" / "1" / "infinity")
                          ; Used as a getmetadata-option

      setmetadata       = "SETMETADATA" SP mailbox
                                        SP entry-values
                          ; empty string for mailbox implies
                          ; server annotation.

      value             = nstring / literal8
```

6. IANA Considerations

All entries MUST have either "/shared" or "/private" as a prefix. Entry names MUST be specified in a Standards Track or IESG-approved Experimental RFC, or fall under the vendor namespace (i.e., use /shared/vendor/<vendor-token> or /private/vendor/<vendor-token> as the prefix).

Each entry registration MUST include a content-type that is used to indicate the nature of the annotation value. Where applicable, a charset parameter MUST be included with the content-type.

6.1. Entry and Attribute Registration Template

```
To: iana@iana.org
Subject: IMAP METADATA Entry Registration

Type:         [Either "Mailbox" or "Server"]

Name:         [the name of the entry]

Description:  [a description of what the entry is for]

Content-type: [MIME Content-Type and charset for the entry value]

RFC Number:   [for entries published as RFCs]

Contact:      [email and/or physical address to contact for
               additional information]
```

6.2. Server Entry Registrations

The following templates specify the IANA registrations of annotation entries specified in this document.

6.2.1. /shared/comment

```
To: iana@iana.org
Subject: IMAP METADATA Entry Registration

Type:         Server

Name:         /shared/comment

Description:  Defines a comment or note that is associated
              with the server and that is shared with
              authorized users of the server.

Content-type: text/plain; charset=utf-8

RFC Number:   RFC 5464

Contact:      IMAP Extensions mailto:ietf-imapext@imc.org
```

6.2.2. /shared/admin

```
To: iana@iana.org
Subject: IMAP METADATA Entry Registration

Type:         Server

Name:         /shared/admin

Description:  Indicates a method for contacting the server
              administrator.  The value MUST be a URI (e.g., a
              mailto: or tel: URL).  This entry is always
              read-only -- clients cannot change it.  It is visible
              to authorized users of the system.

Content-type: text/plain; charset=utf-8

RFC Number:   RFC 5464

Contact:      IMAP Extensions mailto:ietf-imapext@imc.org
```

6.3. Mailbox Entry Registrations

The following templates specify the IANA registrations of annotation entries specified in this document.

6.3.1. /shared/comment

```
To: iana@iana.org
Subject: IMAP METADATA Entry Registration

Type:         Mailbox

Name:         /shared/comment

Description:  Defines a shared comment or note associated with a
              mailbox.

Content-type: text/plain; charset=utf-8

RFC Number:   RFC 5464

Contact:      IMAP Extensions mailto:ietf-imapext@imc.org
```

6.3.2. /private/comment

```
To: iana@iana.org
Subject: IMAP METADATA Entry Registration

Type:         Mailbox

Name:         /private/comment

Description:  Defines a private comment or note associated with a
              mailbox.

Content-type: text/plain; charset=utf-8

RFC Number:   RFC 5464

Contact:      IMAP Extensions mailto:ietf-imapext@imc.org
```

7. Security Considerations

The security considerations in Section 11 of [RFC3501] apply here with respect to protecting annotations from snooping. Servers MAY choose to only support the METADATA and/or METADATA-SERVER extensions after a privacy layer has been negotiated by the client.

Annotations can contain arbitrary data of varying size. As such, servers MUST ensure that size limits are enforced to prevent a user from using up all available space on a server and preventing use by others. Clients MUST treat annotation data values as an "untrusted" source of data as it is possible for it to contain malicious content.

Annotations whose values are intended to remain private MUST be stored only in entries that have the "/private" prefix on the entry name.

Excluding the above issues, the METADATA extension does not raise any security considerations that are not present in the base IMAP protocol, and these issues are discussed in [RFC3501].

8. Normative References

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

[RFC2244] Newman, C. and J. Myers, "ACAP -- Application Configuration Access Protocol", RFC 2244, November 1997.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

[RFC4314] Melnikov, A., "IMAP4 Access Control List (ACL) Extension", RFC 4314, December 2005.

[RFC4466] Melnikov, A. and C. Daboo, "Collected Extensions to IMAP4 ABNF", RFC 4466, April 2006.

[RFC5161] Gulbrandsen, A. and A. Melnikov, "The IMAP ENABLE Extension", RFC 5161, March 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

Appendix A. Acknowledgments

The ideas expressed in this document are based on the message annotation document that was co-authored by Randall Gellens. The author would like to thank the following individuals for contributing their ideas and support for writing this specification: Dave Cridland, Arnt Gulbrandsen, Dan Karp, Alexey Melnikov, Ken Murchison, Chris Newman, and Michael Wener.

Author's Address

Cyrus Daboo
Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
USA

EMail: cyrus@daboo.name
URI: http://www.apple.com/

Full Copyright Statement

Intellectual Property