IMAP4 Extension: Message Preview Generation
draft-ietf-extra-imap-fetch-preview-10

Abstract

   This document specifies an Internet Message Access Protocol (IMAP)
   protocol extension allowing a client to request a server-generated
   abbreviated text representation of message data useful as a
   contextual preview of the entire message.

Status of This Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at https://datatracker.ietf.org/drafts/current/.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   This Internet-Draft will expire on April 3, 2021.

Copyright Notice

Table of Contents

1. Introduction

   Many modern mail clients display small extracts of the body text as
   an aid to allow a user to quickly decide whether they are interested
   in viewing the full message contents. Mail clients implementing the
   Internet Message Access Protocol [RFC3501] would benefit from a
   standardized, consistent way to generate these brief textual previews
   of messages.

   Generation of a preview on the server has several benefits. First,
   it allows consistent representation of previews across all clients.
   This standardized display can reduce user confusion when using
   multiple clients, as abbreviated message representations in clients
   will show identical message contents.

   Second, server-side preview generation is more efficient. A client-
   based algorithm needs to issue, at a minimum, a FETCH BODYSTRUCTURE
   command in order to determine which MIME [RFC2045] body part(s)
   should be represented in the preview. Subsequently, at least one
   FETCH BODY command may be needed to retrieve body data used in
   preview generation. These FETCH commands cannot be pipelined since
   the BODYSTRUCTURE query must be parsed on the client before the list
   of parts to be retrieved via the BODY command(s) can be determined.

Additionally, it may be difficult to predict the amount of body data
that must be retrieved to adequately represent the part via a
preview, therefore requiring inefficient fetching of excessive data
in order to account for this uncertainty. For example, a preview
algorithm to display data contained in a text/html [RFC2854] part
will likely strip the markup tags to obtain textual content.
However, without fetching the entire content of the part, there is no
way to guarantee that sufficient non-tag content will exist unless
either 1) the entire part is retrieved or 2) an additional partial
FETCH is executed when the client determines that it does not possess
sufficient data from a previous partial FETCH to display an adequate
representation of the preview.

Finally, server generation allows caching in a centralized location.
Using server-generated previews allows global generation once per
message, and that preview can be cached for the retention period of
the source message. Retrieval of message data may be expensive
within a server, for example, so a server can be configured to reduce
its storage retrieval load by pre-generating preview data.

A server indicates support for this extension via the "PREVIEW"
capability name.

2. Conventions Used In This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
"OPTIONAL" in this document are to be interpreted as described in BCP
14 [RFC2119] [RFC8174] when, and only when, they appear in all
capitals, as shown here.

"User" is used to refer to a human user, whereas "client" refers to
the software being run by the user.

In examples, "C:" and "S:" indicate lines sent by the client and
server respectively. If a single "C:" or "S:" label applies to
multiple lines, then the line breaks between those lines are for
editorial clarity only and are not part of the actual protocol
exchange.

As with all IMAP extension documents, the case used in writing IMAP
protocol elements herein is chosen for editorial clarity, and
implementations must pay attention to the numbered rules at the
beginning of [RFC3501] Section 9.

3. FETCH Data Item

3.1. Command

To retrieve a preview for a message, the "PREVIEW" FETCH attribute is
used when issuing a FETCH command.

3.2. Response

The server returns a variable-length string that is the generated
preview for that message. This string is intended to be viewed by
the user as a contextual preview of the entire message, and is not
intended to be interpreted in any way by the client software.

Example: Retrieving preview information in a SELECTed mailbox

```
C: A1 FETCH 1 (PREVIEW)
S: * 1 FETCH (PREVIEW "Preview text!")
S: A1 OK FETCH complete.
```

A server SHOULD strive to generate the same string for a given
message for each request. However, since previews are understood to
be an approximation of the message data and not a canonical view of
its contents, a client MUST NOT assume that a message preview is
immutable for a given message. This relaxed requirement permits a
server to offer previews as an option without requiring potentially
burdensome storage and/or processing requirements to guarantee
immutability for a use case that does not require this strictness.
For example, the underlying IMAP server may change due to a system
software upgrade; an account's state information may be retained in
the migration but the new server may generate different PREVIEW text
than the old server.

It is possible that the server has determined that no meaningful
preview text can be generated for a particular message. Examples of
this involve encrypted messages, content types the server does not
support previews of, and other situations where the server is not
able to extract information for a preview. In such cases, the server
MUST return a zero-length string. Clients SHOULD NOT send another
FETCH for a preview for such messages. (As discussed previously,
preview data is not immutable so there is chance that at some point
in the future the server would be able to generate meaningful text.
However, this scenario is expected to be rare so a client should not
continually send out requests to try to capture this infrequent
occurrence.)

If the LAZY modifier is used, the server MAY return NIL for the
preview response, indicating that preview generation could not be

completed without causing undue delay. A server MUST NOT return NIL
to a FETCH PREVIEW request made without the LAZY modifier.

3.3. Preview Text Format

The generated preview text MUST be treated as text/plain [RFC2046]
media type data by the client.

The generated string MUST NOT be content transfer encoded and MUST be
encoded in UTF-8 [RFC3629]. The server SHOULD remove any formatting
markup and do whatever processing might be useful in rendering the
preview as plain text.

For purposes of this section, a "preview character" is defined as a
single UCS character encoded in UTF-8. Note: a single preview
character may compromise multiple octets, so any buffers implemented
to conform to the string limitations identified in this document
should be sized to prevent possible overflow errors.

The server SHOULD limit the length of the preview text to 200 preview
characters. This length should provide sufficient data to generally
support both various languages (and their different average word
lengths) and diverse client display size requirements.

The server MUST NOT output preview text longer than 256 preview
characters.

If the preview is not generated based on the body content of the
message, and the LANGUAGE [RFC5255] extension is supported by the
server, the preview text SHOULD be generated according to the
language rules that apply to human-readable text. For example, a
message that consists of a single image MIME part has no human-
readable text from which to generate preview information. Instead,
the server may wish to output a description that the message contains
an image and describe some attributes of the image, such as image
format, size, and filename. This descriptive text is not a product
of the message body itself but is rather auto-generated data by the
server, and should thus use the rules defined for human-readable text
described in the LANGUAGE extension (if supported on the server).

4. LAZY Priority Modifier

4.1. LAZY

The LAZY modifier directs the server to return the preview
representation only if that data can be returned without undue delay
to the client.

If this modifier is used, and the server is unable to return preview
data without undue delay, the server MUST return NIL as the preview
response.

The LAZY modifier MUST be implemented by any server that supports the
PREVIEW extension.

4.2. Client Implementation Advice

Upon opening a mailbox, a client generally performs a FETCH of
message details in order to create a listing to present to the user
(e.g. ENVELOPE data). Using this extension, a client may want to
additionally display preview information as part of this listing.
Quickly providing the base mailbox listing, with basic message
details, is the primary goal of this command as this is required to
allow the user to begin interacting with the mailbox. Preview data
is likely to be of secondary importance; it provides useful context,
but it is not necessary to perform message actions. A client can
load unavailable previews in the background and display them
asynchronously to the user as the preview data is provided by the
server.

In this scenario, the client would add the PREVIEW data item, with
the LAZY modifier, to the list of FETCH items needed to generate the
mailbox listing. This allows the server to advantageously return
preview data without blocking the primary goal of quickly returning
the basic message details used to generate the mailbox listing.

Once this initial FETCH is complete, the client can then issue FETCH
requests, without the LAZY modifier, to load the PREVIEW data item
for the messages in which preview data was not returned. It is
RECOMMENDED that these FETCH requests be issued in small batches,
e.g., 50 messages per FETCH command, since preview generation may be
expensive and a single large request may exceed server resource
limits.

See Example 2 for an implementation of this strategy.

A client SHOULD NOT continually issue LAZY PREVIEW FETCH commands in
a selected mailbox as the server is under no requirement to return
preview information for this command, which could lead to an
unnecessary waste of system and network resources.

5. Examples

   Example 1: Requesting PREVIEW without LAZY modifier.

      C: A1 CAPABILITY
      S: * CAPABILITY IMAP4rev1 PREVIEW
      S: A1 OK Capability command completed.
      [...a mailbox is SELECTed...]
      C: A2 FETCH 1 (RFC822.SIZE PREVIEW)
      S: * 1 FETCH (RFC822.SIZE 5647 PREVIEW {200}
      S: Lorem ipsum dolor sit amet, consectetur adipiscing elit.
      S: Curabitur aliquam turpis et ante dictum, et pulvinar dui congue.
      S: Maecenas hendrerit, lorem non imperdiet pellentesque, nulla
      S: ligula nullam
      S:)
      S: A2 OK FETCH complete.

   Example 2: Requesting PREVIEW with LAZY modifier, to obtain previews
   during initial mailbox listing if readily available; otherwise, load
   previews in background.

      C: D1 FETCH 1:4 (ENVELOPE PREVIEW (LAZY))
      S: * 1 FETCH (ENVELOPE ("Wed, 23 Sep 2020 15:03:11 +0000" [...])
         PREVIEW "Preview text for message 1.")
      S: * 2 FETCH (PREVIEW "" ENVELOPE
         ("Thu, 24 Sep 2020 12:17:23 +0000" [...]))
      S: * 3 FETCH (ENVELOPE ("Fri, 25 Sep 2020 09:13:45 +0000" [...])
         PREVIEW NIL)
      S: * 4 FETCH (ENVELOPE ("Sat, 26 Sep 2020 07:11:18 +0000" [...])
         PREVIEW NIL)
      S: D1 OK FETCH completed.
      [...Client has preview for message 1 and knows that message 2 has
          a preview that is empty; only need to request preview of
          messages 3 & 4 (e.g. in background)...]
      C: D2 FETCH 3:4 (PREVIEW)
      S: * 3 FETCH (PREVIEW {30}
      S: Message data from message 3.
      S:)
      S: * 4 FETCH (PREVIEW "Message 4 preview")
      S: D2 OK Fetch completed.

Example 3: Retrieve preview information for search results within a
single mailbox. Use SEARCHRES [RFC5182] extension to save a round-
trip.

```
C: E1 CAPABILITY
S: * CAPABILITY IMAP4rev1 PREVIEW SEARCHRES
S: E1 OK Capability command completed.
[...a mailbox is SELECTed...]
C: E2 SEARCH RETURN (SAVE) FROM "FOO"
C: E3 FETCH $ (UID PREVIEW (LAZY))
S: E2 OK SEARCH completed.
S: * 5 FETCH (UID 13 PREVIEW "Preview!")
S: * 9 FETCH (UID 23 PREVIEW NIL)
S: E3 OK FETCH completed.
[...Retrieve message 9 preview in background...]
C: E4 UID FETCH 23 (PREVIEW)
S: * 9 FETCH (UID 23 PREVIEW "Another preview!")
S: E4 OK FETCH completed.
```

6. Formal Syntax

The following syntax specification uses the augmented Backus-Naur
Form (BNF) as described in ABNF [RFC5234]. It includes definitions
from IMAP [RFC3501].

```
capability        =/ "PREVIEW"

fetch-att         =/ "PREVIEW" [SP "(" preview-mod *(SP
                      preview-mod) ")"]

msg-att-dynamic   =/ "PREVIEW" SP nstring

preview-mod       =  "LAZY"
```

7. IANA Considerations

IMAP4 [RFC3501] capabilities are registered by publishing a standards
track or IESG-approved experimental RFC. The registry is currently
located at:

   http://www.iana.org/assignments/imap-capabilities

This document requests that IANA adds the "PREVIEW" capability to the
IMAP4 [RFC3501] capabilities registry.

8. Security Considerations

   Implementation of this extension might enable denial-of-service
   attacks against server resources, due to excessive memory or CPU
   usage during preview generation or increased storage usage if preview
   results are stored on the server after generation. In order to
   mitigate such attacks, servers SHOULD log the client authentication
   identity on FETCH PREVIEW operations in order to facilitate tracking
   of abusive clients.

   Servers MAY limit the resources that preview generation uses. Such
   resource limitations might, in an extreme example, cause a server to
   return a preview that is the empty string for a message that
   otherwise would have had a non-empty preview. However, it is
   recommended that at least some preview text be provided in this
   situation, even if the quality of the preview is degraded.

   Just as the messages they summarize, preview data may contain
   sensitive information. If generated preview data is stored on the
   server, e.g. for caching purposes, these previews MUST be protected
   with equivalent authorization and confidentiality controls as the
   source message.

9. References

9.1. Normative References

   [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail
              Extensions (MIME) Part Two: Media Types", RFC 2046,
              DOI 10.17487/RFC2046, November 1996,
              <https://www.rfc-editor.org/info/rfc2046>.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119,
              DOI 10.17487/RFC2119, March 1997,
              <https://www.rfc-editor.org/info/rfc2119>.

   [RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION
              4rev1", RFC 3501, DOI 10.17487/RFC3501, March 2003,
              <https://www.rfc-editor.org/info/rfc3501>.

   [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
              10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November
              2003, <https://www.rfc-editor.org/info/rfc3629>.

   [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax
              Specifications: ABNF", STD 68, RFC 5234,
              DOI 10.17487/RFC5234, January 2008,
              <https://www.rfc-editor.org/info/rfc5234>.

   [RFC5255] Newman, C., Gulbrandsen, A., and A. Melnikov, "Internet
              Message Access Protocol Internationalization", RFC 5255,
              DOI 10.17487/RFC5255, June 2008,
              <https://www.rfc-editor.org/info/rfc5255>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

   [RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail
              Extensions (MIME) Part One: Format of Internet Message
              Bodies", RFC 2045, DOI 10.17487/RFC2045, November 1996,
              <https://www.rfc-editor.org/info/rfc2045>.

   [RFC2854] Connolly, D. and L. Masinter, "The 'text/html' Media
              Type", RFC 2854, DOI 10.17487/RFC2854, June 2000,
              <https://www.rfc-editor.org/info/rfc2854>.

   [RFC5182] Melnikov, A., "IMAP Extension for Referencing the Last
              SEARCH Result", RFC 5182, DOI 10.17487/RFC5182, March
              2008, <https://www.rfc-editor.org/info/rfc5182>.

Appendix A. Change History (To be removed by RFC Editor before publication)

   Changes from draft-slusarz-imap-fetch-snippet-00:

   o Added standardized language to Section 2 regarding IMAP ABNF
      conventions

   o Changed draft name to draft-ietf-extra-imap-fetch-snippet-##

   Changes from draft-ietf-extra-imap-fetch-snippet-00:

   o Changed nomenclature from "snippet" to "preview"

   o Changed draft name to draft-ietf-extra-imap-fetch-preview-##

   o Update to RFC 8174 boilerplate

   o Updated length requirements for PREVIEW=FUZZY

o Added preview-atom ABNF to limit use of "=" character

o UTF-8 is a normative reference

o Clarify that characters for purpose of length limitations are
   defined as UCS characters as encoded by UTF-8

o Fix some incorrect literal lengths in examples

Changes from draft-ietf-extra-imap-fetch-preview-00:

o Updated postal address

o Added example to FETCH response section

o Added example on how LANGUAGE extension may influence preview
   generation

o Added recommendation that only one LAZY FETCH be executed for a
   message per mailbox

o Added request to create algorithm and modifier registries

o Added requirement that algorithm and modifier names conform to RFC
   6648

o Added DoS attack info to security considerations

o Distinguish between NIL response and zero-length string

o Don't use deprecated "X-" convention in example

o Spelling and nits

Changes from draft-ietf-extra-imap-fetch-preview-01:

o Fix capability ABNF

o Removed CAPABILITY string for examples where it did not add
   valuable context

o Altered preview data in examples to cover a variety of potential
   server return scenarios

o Added "SHOULD be registered" language to algorithm names and
   priority modifiers

Changes from draft-ietf-extra-imap-fetch-preview-02:

o Move Acknowledgments to un-numbered appendix

o Improved abstract text

o Consistently use "priority modifiers" instead of "modifiers"

o Update example to conform with RFC 3501 UID FETCH requirements

Changes from draft-ietf-extra-imap-fetch-preview-03:

o Remove preview modifier registry request

o Improve instructions for registration of algorithms

o Add storage information to security considerations

o Clarify parsing of algorithm list in FETCH command

o Clarify difference between NIL response and zero-length string

o Add normative reference for text/plain

o Add warning regarding buffers and multiple octet preview
   characters

o Clarify how to handle preview data return when using an explicit
   algorithm list

o Various editorial fixes

Changes from draft-ietf-extra-imap-fetch-preview-04:

o Make clear that preview caching is tied to retention period of the
   source message

Changes from draft-ietf-extra-imap-fetch-preview-05:

o Clarify "zero-length string" preview data vs. NIL preview data

o MIME data -> media type

o Capability registration should not include the algorithm name

o Give example of how PREVIEW data might change over time

Changes from draft-ietf-extra-imap-fetch-preview-06:

o Change algorithm names to media types

o FUZZY algorithm changed to text/imap-fetch-preview

o Remove server broadcast of PREVIEW algorithm extensions from
   capability

o Default, fallback algorithm in absence of client selection now
   MUST be text/imap-fetch-preview

o LAZY modifier should work on default algorithm if no specific
   algorithm is provided as an argument

Changes from draft-ietf-extra-imap-fetch-preview-07:

o Remove algorithm selection; PREVIEW always returns text in format
   defined in Section 3.3

Changes from draft-ietf-extra-imap-fetch-preview-08:

o FETCH PREVIEW without LAZY modifier MUST NOT return NIL

o Improve client implementation advice for LAZY modifier

Changes from draft-ietf-extra-imap-fetch-preview-09:

o Clarified that string response is to be interpreted by user, not
   the client

o Give example behavior of resource limitation

o Various editorial fixes

Acknowledgments

   The author would like to thank the following people for their
   comments and contributions to this document: Stephan Bosch, Bron
   Gondwana, Teemu Huovila, Neil Jenkins, Steffen Lehmann, Barry Leiba,
   Alexey Melnikov, Chris Newman, Pete Resnick, Jeff Sipek, Timo
   Sirainen, Steffen Templin, and Aki Tuomi.

Author's Address

   Michael M. Slusarz
   Open-Xchange Inc.
   530 Lytton Avenue
   Palo Alto, California 94301
   US

   Email: michael.slusarz@open-xchange.com