IMAP Extension for object identifiers
draft-ietf-extra-imap-objectid-08

Abstract

   This document updates RFC3501 (IMAP4rev1) with persistent identifiers
   on mailboxes and messages to allow clients to more efficiently re-use
   cached data when resources have changed location on the server.

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 February 3, 2019.

Table of Contents

1. Introduction

   IMAP stores are often used by many clients. Each client may cache
   data from the server so that they don't need to re-download
   information. [RFC3501] defines that a mailbox can be uniquely
   referenced by its name and UIDVALIDITY, and a message within that

mailbox can be uniquely referenced by its mailbox (name +
UIDVALIDITY) and UID. The triple of mailbox name, UIDVALIDITY and
UID is guaranteed to be immutable.

[RFC4315] defines a COPYUID response which allows a client which
copies messages to know the mapping between the UIDs in the source
and destination mailboxes, and hence update its local cache.

If a mailbox is successfully renamed by a client, that client will
know that the same messages exist in the destination mailbox name as
previously existed in the source mailbox name.

The result is that the client which copies (or [RFC6851] moves)
messages or renames a mailbox can update its local cache, but any
other client connected to the same store can not know with certainty
that the messages are identical, and so will re-download everything.

This extension adds new properties to a message (EMAILID) and mailbox
(MAILBOXID) which allow a client to quickly identify messages or
mailboxes which have been renamed by another client.

This extension also adds an optional thread identifier (THREADID) to
messages, which can be used by the server to indicate messages which
it has identified to be related. A server that does not implement
threading will return NIL to all requests for THREADID.

2. Conventions Used In This Document

In examples, "C:" indicates lines sent by a client that is connected
to a server. "S:" indicates lines sent by the server to the client.

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.

3. CAPABILITY Identification

IMAP servers that support this extension MUST include "OBJECTID" in
the response list to the CAPABILITY command.

4. MAILBOXID object identifier

The MAILBOXID is a server-allocated unique identifer for each
mailbox.

The server MUST return the same MAILBOXID for a mailbox with the same
name and UIDVALIDITY.

The server MUST NOT report the same MAILBOXID for two mailboxes at
the same time.

The server MUST NOT reuse the same MAILBOXID for a mailbox which does
not obey all the invariants that [RFC3501] defines for a mailbox
which does not change name or UIDVALIDITY.

The server MUST keep the same MAILBOXID for the source and
destination when renaming a mailbox in a way which keeps the same
messages (but see [RFC3501] for the special case regarding renaming
of INBOX, which is treated as creating a new mailbox and moving the
messages)

4.1. New response code for CREATE

This document extends the CREATE command to have the response code
MAILBOXID on successful mailbox creation.

A server advertising the OBJECTID capability MUST include the
MAILBOXID response code in the tagged OK response to all successful
CREATE commands.

Syntax: "MAILBOXID" SP "(" objectid ")"

     Response code in tagged OK for successful CREATE command.

Example:

```
C: 3 create foo
S: 3 OK [MAILBOXID (F2212ea87-6097-4256-9d51-71338625)] Completed
C: 4 create bar
S: 4 OK [MAILBOXID (F6352ae03-b7f5-463c-896f-d8b48ee3)] Completed
C: 5 create foo
S: 5 NO Mailbox already exists
```

4.2. New OK Untagged Response for SELECT and EXAMINE

This document adds a new untagged response code to the SELECT and
EXAMINE commands.

A server advertising the OBJECTID capability MUST return an untagged
OK response with the MAILBOXID response code on all successful SELECT
and EXAMINE commands.

Syntax: "OK" SP "[" "MAILBOXID" SP "(" objectid ")" "]" SP text

                   Untagged OK response to SELECT or EXAMINE.

   Example:

        C: 27 select "foo"
        [...]
        S: * OK [MAILBOXID (F2212ea87-6097-4256-9d51-71338625)] Ok
        [...]
        S: 27 OK [READ-WRITE] Completed

4.3. New attribute for STATUS

   This document adds the MAILBOXID attribute to the STATUS command
   using the extended syntax defined in [RFC4466].

   A server that advertises the OBJECTID capability MUST support the
   MAILBOXID status attribute.

   Syntax: "MAILBOXID"

                  The attribute in the STATUS command.

   Syntax: "MAILBOXID" SP "(" objectid ")"

      The response item in the STATUS response contains the objectid
      assigned by the server for this mailbox.

   Example:

    C: 6 status foo (mailboxid)
    S: * STATUS foo (MAILBOXID (F2212ea87-6097-4256-9d51-71338625))
    S: 6 OK Completed
    C: 7 status bar (mailboxid)
    S: * STATUS bar (MAILBOXID (F6352ae03-b7f5-463c-896f-d8b48ee3))
    S: 7 OK Completed
    C: 8 rename foo renamed
    S: * OK rename foo renamed
    S: 8 OK Completed
    C: 9 status renamed (mailboxid)
    S: * STATUS renamed (MAILBOXID (F2212ea87-6097-4256-9d51-71338625))
    S: 9 OK Completed
    C: 10 status bar (mailboxid)
    S: * STATUS bar (MAILBOXID (F6352ae03-b7f5-463c-896f-d8b48ee3))
    S: 10 OK Completed

   When the LIST-STATUS IMAP capability defined in [RFC5819] is also
   available, the STATUS command can be combined with the LIST command.

Example:

```
C: 11 list "" "*" return (status (mailboxid))
S: * LIST (\HasNoChildren) "." INBOX
S: * STATUS INBOX (MAILBOXID (Ff8e3ead4-9389-4aff-adb1-d8d89efd8cbf))
S: * LIST (\HasNoChildren) "." bar
S: * STATUS bar (MAILBOXID (F6352ae03-b7f5-463c-896f-d8b48ee3))
S: * LIST (\HasNoChildren) "." renamed
S: * STATUS renamed (MAILBOXID (F2212ea87-6097-4256-9d51-71338625))
S: 11 OK Completed (0.001 secs 3 calls)
```

5. EMAILID object identifier and THREADID correlator

5.1. EMAILID identifier for identical messages

The EMAILID data item is an objectid which uniquely identifies the
content of a single message. Anything which must remain immutable on
a {name, uidvalidity, uid} triple must also be the same between
messages with the same EMAILID.

The server MUST return the same EMAILID for the same triple, hence
EMAILID is immutable.

The server MUST return the same EMAILID as the source message for the
matching destination message in the COPYUID pairing after a COPY or
[RFC6851] MOVE command.

The server MAY assign the same EMAILID as an existing message upon
APPEND (e.g. if it detects that the new message has exactly identical
content to that of an existing message)

NOTE: EMAILID only identifies the immutable content of the message.
In particular, it is possible for different messages with the same
EMAILID to have different keywords. This document does not specify a
way to STORE by EMAILID.

5.2. THREADID identifer for related messages

The THREADID data item is an objectid which uniquely identifies a set
of messages which the server believes should be grouped together when
presented.

THREADID calculation is generally based on some combination of
References, In-Reply-To and Subject, but the exact logic is left up
to the server implementation. [RFC5256] describes some algorithms
that could be used, however this specfication does not mandate any
particular strategy.

The server MUST return the same THREADID for all messages with the
same EMAILID.

The server SHOULD return the same THREADID for related messages even
if they are in different mailboxes, e.g. messages which would appear
in the same thread if they were in the same mailbox SHOULD have the
same THREADID even if they are in different mailboxes.

The server MUST NOT change the THREADID of a message once reported.

THREADID is OPTIONAL; if the server doesn't support THREADID or is
unable to calculate relationships between messages, it MUST return
NIL to all FETCH responses for the THREADID data item, and a SEARCH
for THREADID MUST NOT match any messages.

The server MUST NOT use the same objectid value for both EMAILIDs and
THREADIDs. If they are stored with the same value internally, the
server can generate prefixed values (as shown in the examples below
with M and T prefixes) to avoid clashes.

5.3. New Message Data Items in FETCH and UID FETCH Commands

This document defines two FETCH items:

Syntax: "EMAILID"

  The EMAILID message data item causes the server to return EMAILID
  FETCH response data items.

Syntax: "THREADID"

 The THREADID message data item causes the server to return THREADID
 FETCH response data items.

And the following responses:

Syntax: "EMAILID" SP "(" objectid ")"

The EMAILID response data item contains the server-assigned objectid
for each message.

Syntax: "THREADID" SP "(" objectid ")"

The THREADID response data item contains the server-assigned objectid
for the set of related messages to which this message belongs.

Syntax: "THREADID" SP nil

The NIL value to the THREADID response data item is returned when
the server mailbox does not support THREADID calculation.

Example:

```
C: 5 append inbox "20-Mar-2018 03:07:37 +1100" {733}
[...]
Subject: Message A
Message-ID: <fake.1521475657.54797@example.com>
[...]
S: 5 OK [APPENDUID 1521475658 1] Completed

C: 11 append inbox "20-Mar-2018 03:07:37 +1100" {793}
[...]
Subject: Re: Message A
Message-ID: <fake.1521475657.21213@example.org>
References: <fake.1521475657.54797@example.com>
[...]
S: 11 OK [APPENDUID 1521475658 2] Completed

C: 17 append inbox "20-Mar-2018 03:07:37 +1100" {736}
[...]
Subject: Message C
Message-ID: <fake.1521475657.60280@example.com>
[...]
S: 17 OK [APPENDUID 1521475658 3] Completed

C: 22 fetch 1:* (emailid threadid)
S: * 1 FETCH (EMAILID (M6d99ac3275bb4e) THREADID (T64b478a75b7ea9))
S: * 2 FETCH (EMAILID (M288836c4c7a762) THREADID (T64b478a75b7ea9))
S: * 3 FETCH (EMAILID (M5fdc09b49ea703) THREADID (T11863d02dd95b5))
S: 22 OK Completed (0.000 sec)

C: 23 move 2 foo
S: * OK [COPYUID 1521475659 2 1] Completed
S: * 2 EXPUNGE
S: 23 OK Completed

C: 24 fetch 1:* (emailid threadid)
S: * 1 FETCH (EMAILID (M6d99ac3275bb4e) THREADID (T64b478a75b7ea9))
S: * 2 FETCH (EMAILID (M5fdc09b49ea703) THREADID (T11863d02dd95b5))
S: 24 OK Completed (0.000 sec)
C: 25 select "foo"

C: 25 select "foo"
[...]
S: 25 OK [READ-WRITE] Completed
C: 26 fetch 1:* (emailid threadid)
S: * 1 FETCH (EMAILID (M288836c4c7a762) THREADID (T64b478a75b7ea9))
S: 26 OK Completed (0.000 sec)
```

Example: (no THREADID support)

```
                  C: 26 fetch 1:* (emailid threadid)
                  S: * 1 FETCH (EMAILID (M00000001) THREADID NIL)
                  S: * 2 FETCH (EMAILID (M00000002) THREADID NIL)
                  S: 26 OK Completed (0.000 sec)
```


6. New Filters on SEARCH command

   This document defines filters EMAILID and THREADID on the SEARCH
   command.

   Syntax: "EMAILID" SP objectid

        Messages whose EMAILID is exactly the specified objectid.

   Syntax: "THREADID" SP objectid

        Messages whose THREADID is exactly the specified objectid.

   Example: (as if run before the MOVE above when the mailbox had 3
   messages)

```
                  C: 27 search emailid M6d99ac3275bb4e
                  S: * SEARCH 1
                  S: 27 OK Completed (1 msgs in 0.000 secs)
                  C: 28 search threadid T64b478a75b7ea9
                  S: * SEARCH 1 2
                  S: 28 OK Completed (2 msgs in 0.000 secs)
```

7. Formal syntax

   The following syntax specification uses the Augmented Backus-Naur
   Form (ABNF) [RFC5234] notation. Elements not defined here can be
   found in the formal syntax of the ABNF [RFC5234], IMAP [RFC3501], and
   IMAP ABNF extensions [RFC4466] specifications.

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

   Please note specifically that objectid values are case sensitive.

```
   capability =/ "OBJECTID"

   fetch-att =/ "EMAILID" / "THREADID"

   fetch-emailid-resp = "EMAILID" SP "(" objectid ")"
           ; follows tagged-ext production from [@!RFC4466]

   fetch-threadid-resp = "THREADID" SP ( "(" objectid ")" / nil )
           ; follows tagged-ext production from [@!RFC4466]

   msg-att-static =/ fetch-emailid-resp / fetch-threadid-resp

   objectid = 1*255(ALPHA / DIGIT / "_" / "-")
           ; characters in object identifiers are case
           ; significant

   resp-text-code =/ "MAILBOXID" SP "(" objectid ")"
           ; incorporated before the expansion rule of
           ;  atom [SP 1*<any TEXT-CHAR except "]">]
           ; that appears in [@!RFC3501]

   search-key =/ "EMAILID" SP objectid / "THREADID" SP objectid

   status-att =/ "MAILBOXID"

   status-att-value =/ "MAILBOXID" SP "(" objectid ")"
           ; follows tagged-ext production from [@!RFC4466]
```

8. Implementation considerations

8.1. Assigning object identifiers

   All objectid values are allocated by the server.

   In the interests of reducing the possibilities of encoding mistakes,
   objectids are restricted to a safe subset of possible byte values,
   and in order to allow clients to allocate storage, they are
   restricted in length.

   An objectid is a string of 1 to 255 characters from the following set
   of 64 codepoints: a-z, A-Z, 0-9, '_', '-'. These characters are safe
   to use in almost any context (e.g. filesystems, URIs, IMAP atoms).
   These are the same characters defined as base64url in [RFC4648].

   For maximum safety, servers should also follow defensive allocation
   strategies to avoid creating risks where glob completion or data type
   detection may be present (e.g. on filesystems or in spreadsheets).
   In particular it is wise to avoid:

o ids starting with -

o ids starting with digits

o ids which contain only digits

o ids which differ only by ASCII case (A vs a)

o the specific sequence of 3 characters NIL in any case (as this
   sequence can be confused with the IMAP protocol expression of the
   null value)

A good solution to these issues is to prefix every ID with a single
alphabetical character.

8.2. Interaction with special cases

The case of RENAME INBOX may need special handling, as it has special
behaviour as defined in [RFC3501] section 6.3.5.

It is advisable (though not required) to have MAILBOXID be globally
unique, but it is only required to be unique within messages offered
to a single client login to a single server hostname. For example, a
proxy which aggregates multiple independent servers MUST NOT
advertise the OBJECTID capability unless it can guarantee that
different objects will never use the same identifiers, even if
backend object identifiers collide.

8.3. Client usage

Servers that implement both RFC 6154 and this specification should
optimize their execution of command like UID SEARCH OR EMAILID 1234
EMAILID 4321.

Clients can assume that searching the all-mail mailbox using OR/
EMAILID or OR/THREADID is a fast way to find messages again if some
other client has moved them out of the mailbox where they were
previously seen.

Clients that cache data offline should fetch the EMAILID of all new
messages to avoid re-downloading already cached message details.

Clients should fetch the MAILBOXID for any new mailboxes before
discarding cache data for any mailbox which is no longer present on
the server, so that they can detect renames and avoid re-downloading
data.

9. Future considerations

This extension is intentionally defined to be compatible with the
data model in [I-D.ietf-jmap-mail].

A future extension could be proposed to give a way to SELECT a
mailbox by MAILBOXID rather than name.

A future extension to [RFC5228] could allow fileinto by MAILBOXID
rather than name.

An extension to allow fetching message content directly via EMAILID
and message listings by THREADID could be proposed.

10. IANA Considerations

IANA is requested to add "OBJECTID" to the "IMAP Capabilities"
registry located at <http://www.iana.org/assignments/imap-
capabilities> with a Reference of [[THIS RFC]].

IANA is requested to add "MAILBOXID" to the "IMAP Response Codes"
registry located at <https://www.iana.org/assignments/imap-response-
codes> with a Reference of [[THIS RFC]].

11. Security Considerations

It is strongly advised that servers generate OBJECTIDs which are safe
to use as filesystem names, and unlikely to be auto-detected as
numbers. See implementation considerations.

If a digest is used for ID generation, it must have a collision
resistent property, so server implementations are advised to monitor
current security research and choose secure digests. As the IDs are
generated by the server, it will be possible to migrate to a new hash
by just using the new algorith when creating new IDs. This is
particularly true if a prefix is used on each ID, which can be
changed when the algorithm changes.

The use of a digest for ID generation may be used as proof that a
particular sequence of bytes was seen by the server, however this is
only a risk if IDs are leaked to clients who don't have permission to
fetch the data directly. Servers that are expected to handle highly
sensitive data should consider this when choosing how to create IDs.

See also the security considerations in [RFC3501] section 11.

12. Changes

To be removed by the editor before publication

12.1. draft-ietf-extra-imap-objectid-08

o added reference to RFC4648 for base64url (Adam Roach review)

o less prescriptive instruction for digests (Adam Roach review)

o minor punctuation fix (Alissa Cooper review)

o clarified SHOULD around THREADID in different mailboxes (Alissa
   Cooper review)

o clarified that objectids are case sensitive (Eric Rescorla review)

o clarified why NIL is a bad objectid (Eric Rescorla review)

o upper case OPTIONAL for threadid (Pete Resnick genart review)

o missing word in object identifiers clash (Pete Resnick genart
   review)

12.2. draft-ietf-extra-imap-objectid-07

o updated boilerplate to RFC8174 (Benjamin Kaduk review)

o fixed spelling of invariants (Benjamin Kaduk review)

o block quoted ABNF for better text formatting (Benjamin Kaduk
   review)

o clarified that servers can just switch to a new digest without
   changing old IDs (Benjamin Kaduk review)

o changed use of folder to mailbox to avoid confusion (Warren Kumari
   review)

o made both IANA requests say "reference of this RFC" (Warren Kumari
   review)

12.3. draft-ietf-extra-imap-objectid-06

o fixed one more missing space in ABNF (ad review)

o made one more MUST for mailbox being retained on rename (genart
   review)

o updated ABNF to also extend msg-att-static (validator review)

o lowercased NIL => nil in ABNF (validator review)

12.4. draft-ietf-extra-imap-objectid-05

o changed some SHOULD to lower case in advice sections (genart
   review)

o clarified that THREADID MUST NOT change

12.5. draft-ietf-extra-imap-objectid-04

o described NIL THREADID in more detail (ad review)

o made RFC5256 a normative reference (ad review)

o fixed ABNF missing quote (ad review)

o documented hash upgrade process (ad review)

o referenced RFC3501 for INBOX rename (ad review)

o referenced RFC3501 security considerations (secdir review)

o turned mealy-mouthed "SHOULDs" in to "MUSTs" on immutability
   (genart review)

o remove suggested algorithms which are no longer legitimate (genart
   review)

o updated proxy advice to suggest rewriting ids (genart review)

o fixed minor gramatical issues (genart review)

o required that EMAILID and THREADID are not identical (own
   decision)

12.6. draft-ietf-extra-imap-objectid-03

o added RFC3501 to Abstract

o updated [[THIS RFC]] to not fail idnits

o changed jmap-mail to be informative rather than normative

o shortened IDs to stop wrapping and outdents in IMAP examples

12.7. **draft-ietf-extra-imap-objectid-02**

   o added "Client usage" section

12.8. **draft-ietf-extra-imap-objectid-01**

   o added "updates" for RFC3501

   o fixed domains in thread example

   o described threading in more detail

   o added IANA request for Response Code

   o clarified RFC2119 references

   o simplified some waffle in wording

   o added security consideration to choose good digest

   o added MAILBOXID-UID suggestion for EMAILID generation

   o updated ABNF normative reference to RFC5234

12.9. **draft-ietf-extra-imap-objectid-00**

   o renamed draft to be objectid rather than uniqueid

   o renamed UNIQUEID (capability) to OBJECTID

   o restricted objectid to 64 safe characters

   o added security considerations and advice about choosing objectid

   o wrapped all responses in () for RFC4466 compatibility

   o signifiant rewrite of all sections

12.10. **draft-ietf-extra-imap-uniqueid-00**

   o renamed draft to be an EXTRA document

   o added example for LIST RETURN STATUS

   o started work on ABNF

   o attempted to add response codes for EMAILID and THREADID

12.11. draft-gondwana-imap-uniqueid-01

o   renamed UNIQUEID (status item) to MAILBOXID

o   renamed MSGID to EMAILID

o   renamed THRID to THREADID

o   added TODO section

12.12. draft-gondwana-imap-uniqueid-00

o   initial upload with names UNIQUEID/MSGID/THRID

13. Acknowledgments

The EXTRA working group at IETF. In particular feedback from Arnt
Gulbrandsen, Brandon Long, Chris Newman and Josef Sipek.

The Gmail X-GM-THRID and X-GM-MSGID implementation as currently
defined at <https://developers.google.com/gmail/imap/imap-
extensions>.

Dovecot X-GUID implementation.

13.1. Appendix 1: ideas for implementing object identifiers

Ideas for calculating MAILBOXID:

o   [RFC4122] UUID

o   Server assigned sequence number (guaranteed not to be reused)

Ideas for implementing EMAILID:

o   Digest of message content (RFC822 bytes) - expensive unless cached

o   [RFC4122] UUID

o   Server assigned sequence number (guaranteed not to be reused)

Ideas for implementing THREADID:

o   Derive from EMAILID of first seen message in the thread.

o   [RFC4122] UUID

o   Server assigned sequence number (guaranteed not to be reused)

There is a need to index and look up reference/in-reply-to data at
message creation to efficiently find matching messages for threading.
Threading may be either across mailboxes, or within each mailbox
only. The server has significant leeway here.

14. References

14.1. Normative References

[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>.

[RFC4315] Crispin, M., "Internet Message Access Protocol (IMAP) -
           UIDPLUS extension", RFC 4315, DOI 10.17487/RFC4315,
           December 2005, <https://www.rfc-editor.org/info/rfc4315>.

[RFC4466] Melnikov, A. and C. Daboo, "Collected Extensions to IMAP4
           ABNF", RFC 4466, DOI 10.17487/RFC4466, April 2006,
           <https://www.rfc-editor.org/info/rfc4466>.

[RFC5228] Guenther, P., Ed. and T. Showalter, Ed., "Sieve: An Email
           Filtering Language", RFC 5228, DOI 10.17487/RFC5228,
           January 2008, <https://www.rfc-editor.org/info/rfc5228>.

[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>.

[RFC5256] Crispin, M. and K. Murchison, "Internet Message Access
           Protocol - SORT and THREAD Extensions", RFC 5256,
           DOI 10.17487/RFC5256, June 2008,
           <https://www.rfc-editor.org/info/rfc5256>.

[RFC5819] Melnikov, A. and T. Sirainen, "IMAP4 Extension for
           Returning STATUS Information in Extended LIST", RFC 5819,
           DOI 10.17487/RFC5819, March 2010,
           <https://www.rfc-editor.org/info/rfc5819>.

   [RFC6851] Gulbrandsen, A. and N. Freed, Ed., "Internet Message
              Access Protocol (IMAP) - MOVE Extension", RFC 6851,
              DOI 10.17487/RFC6851, January 2013,
              <https://www.rfc-editor.org/info/rfc6851>.

   [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>.

14.2. Informative References

   [I-D.ietf-jmap-mail]
              Jenkins, N., "JMAP for Mail", draft-ietf-jmap-mail-06
              (work in progress), July 2018.

   [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally
              Unique IDentifier (UUID) URN Namespace", RFC 4122,
              DOI 10.17487/RFC4122, July 2005,
              <https://www.rfc-editor.org/info/rfc4122>.

   [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data
              Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006,
              <https://www.rfc-editor.org/info/rfc4648>.

Author's Address

   Bron Gondwana (editor)
   FastMail
   Level 2, 114 William St
   Melbourne VIC 3000
   Australia

   Email: brong@fastmailteam.com
   URI: https://www.fastmail.com