Mail Maintenance
Internet-Draft
Intended status: Informational
Expires: 15 February 2026

R. Signes
Fastmail

14 August 2025

IMAP Extensions Suggestions
draft-ietf-mailmaint-imap-extensions-suggestions-00

Abstract

This document presents a set of IMAP extensions, each of which is recommended as a priority for general-purpose IMAP client and server implementations.

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 15 February 2026.

Table of Contents

1. Introduction

IMAP4 (RFC 9051 et al.) is a complex protocol, and many extensions have been written over its decades of use. Programmers working on IMAP4 clients or servers today have very little guidance on which extensions have much support or have significant value. This document, prepared by a group of established IMAP4 client and server authors, is meant to serve as an overview of which standards should be targeted by living IMAP4 code. There are many trade-offs involved in implementing IMAP4. This document focuses on reducing round trips required to complete common tasks and on eliminating likely confusion for users or user agents.

2. Conventions and Definitions

This document does not define a formal specification, and words "should" and "should not" are used with specific meaning in the document, distinct from that described in RFC 2119.

* should - This word means that the action described is strongly recommended and will enhance interoperability or usability.

* should not - This phrase means that the action described is strongly recommended against, and might hurt interoperability or usability.

3. Core Protocol Version

The current IMAP protocol is IMAP4rev2, specified in RFC 9051.

New server implementations should target IMAP4rev2. This means some of the extensions listed below are required, and that the server is subject to certain further restrictions. Existing server implementations that target earlier revisions of IMAP4 should pursue the recommended extensions below. That is: we suggest IMAP4rev1 with all of the extensions below as a goal before full IMAP4rev2.

IMAP4rev2 and IMAP4rev1 are both in active use, so IMAP4 clients should be capable of interoperating with either protocol. Clients should avoid relying on behaviors removed in IMAP4rev2, and should be capable of operating without features mandated in IMAP4rev2. The extensions below were chosen with the goal of eliminating wasted effort. They are either commonly offered even by IMAP4rev1 servers or provide so much utility that they are worth implementing as optional best-case code paths.

For a list of changes between revisions, see Appendix E of the RFC (https://datatracker.ietf.org/doc/html/rfc9051#changesFromIMAP4rev1).

4. The most effective IMAP extensions

These are the IMAP extensions we believe have the most impact and are most widely implemented. They are listed in no particular order, and instead we’ve provided an explanation of their benefit, so that implementors can choose based on their product’s priorities.

4.1. COMPRESS, RFC 4978

With the COMPRESS extension enabled, the IMAP conversation in both directions is compressed using DEFLATE. In practice, this leads to a 20-40% (or more) reduction in bandwidth used. The deflate algorithm is widely implemented and adds very little computing cost to providing service.

4.2. CONDSTORE and QRESYNC, RFC 7162

These extensions provide efficient mechanisms for re-synchronization. When an IMAP client can make use of these features, it can efficiently update its local cache by only fetching new or changed data. Without these features, the client needs to rescan old mail for changes. Implementing these extensions reduces bandwidth requires and means an IMAP client is fully up to date much more quickly after coming back online. CONDSTORE and QRESYNC are probably the most important extensions to implement to improve IMAP efficiency.

4.3. ESEARCH, RFC 4731

The ESEARCH extension (not to be confused with the ESEARCH command provided by the MULTISEARCH extension) extends the core search mechanism to allow limited results, get a count of results, and get results in a more compact format. This reduces bandwidth and can allow server-side query optimizations.

This extension is required by IMAP4rev2.

4.4. ID, RFC 2971

The ID extension adds the ID command and response, through which the client identifies itself (by name and version) to the server, and the server does the same in return. When a client or server author digs into a bug or other problematic behavior, knowing what piece of software is at the other end of the connection makes it easier to reproduce and diagnose problems. It also means that there’s a path to contact the maintainers of the software involved.

4.5. MOVE, RFC 6851

The MOVE command provides a way to atomically move a message from one mailbox to another, combining a message copy and expunge. This eliminates the possibility of a copied-but-not-deleted message. Also, because there’s no intermediate state where the message exists twice, the command won’t fail due to quota limitations.

This extension is required by IMAP4rev2.

4.6. MULTISEARCH, RFC 7377

This extension adds a new command (ESEARCH, not to be confused with the ESEARCH capability) which can search multiple mailboxes at once. This makes searches both faster and more efficient. Without multisearch, searching multiple mailboxes will require the client pipeline a series of SELECT and SEARCH command. Combining these into as single command may also permit the server to perform a much more efficient search.

4.7. OBJECTID, RFC 8474

The OBJECTID extension provides unique identifiers to messages and threads. This has a number of distinct benefits:

* Mailbox renames can be synchronized extremely efficiently. Without a mailbox id, synchronizing a mailbox rename can be very expensive, especially if it has child mailboxes.

* Mailbox renames need not break Sieve scripts, which can target mailboxes by id in addition to name.

* Messages appearing in multiple mailboxes only need to be synchronized once, saving bandwidth and storage space.

* The object ids can be used to make JMAP calls, if JMAP is available for the target server.

4.8. SPECIAL-USE, RFC 6154

The SPECIAL-USE extension describes a way to add metadata to a mailbox to indicate that the mailbox is for a well-known purpose. For example, the trash mailbox or sent mailbox can be marked as such. This helps client authors meet user expectations consistently across implementations. On servers without this extension, clients are left to make their own mailboxes, and two clients may pick different names, leaving the user with both "Sent" and "Sent Messages".

At minimum servers and clients should most implement support for \Drafts, \Junk, \Sent, and \Trash.

In addition to the behavior from the specification:

* No one mailbox in a user’s personal namespace should have more than one special use attribute. Any combination of two special uses on a mailbox is likely to confuse both users and their user agent software.

* No one special use attribute should be present on two different mailboxes in a user's personal namespace. A client should not be forced to pick between two \Sent mailboxes, for example.

* Special use mailboxes should be at the root of the user's personal mailbox hierarchy, and servers should reject attempts to move special-use mailboxes elsewhere in the hierarchy. Usually, moving a special use mailbox from the top level is the result of a mistake, and leads to complications when users, no longer seeing a "Sent" mailbox, create a new one - which doesn't work.

This extension is required by IMAP4rev2.

5. Further effective IMAP extensions

These extensions are also useful and recommended, although they apply to more specific scenarios than the more general-use extensions listed in the previous section

5.1. NOTIFY, RFC 5465

Core IMAP4rev2 includes the IDLE command, which allows the client to switch into a passive mode and request that updates to the currently selected mailbox instead be pushed to client by the server. The NOTIFY extension provides an improved from of IDLE. It can instruct the server to provide updates for multiple mailboxes, and to send STATUS lines for mailboxes with updates.

This is especially valuable for servers that may deliver new mail to mailboxes other than the inbox, using mail rules or other routing.

5.2. PREVIEW, RFC 8970

The PREVIEW extension provides another property on messages, PREVIEW, which will store a short plain-text snippet of text that serves as a preview of the message content. If a server provides message previews, clients can show message previews without having to fetch body structure or body parts. When clients use server-provided preview text, all clients will have a consistent preview.

5.3. REPLACE, RFC 8508

The REPLACE command provides a way to atomically replace one message with another, combining an append and single-message expunge. It's primarily used for managing draft messages, but can also be used for editing messages "in place" in other ways.

Use of the REPLACE command eliminates the possibility of appending without deleting message. Also, because there's no intermediate state, a REPLACE command won't fail due to quota limitations as could happen with append-then-expunge.

6. Additional notes for client implementations

For one item, I have only a vague not here, "utf8 v utf8", which needs clarifying or to be dropped.

When possible, clients should use the UID command instead of the message sequence number commands it supersedes. For example, UID FETCH, not FETCH. UIDs persist between IMAP sessions, making offline operation simpler. In the future, the UIDONLY extension, RFC 9586, may permit sessions using only the UID form of commands to achieve better client and server performance

7. Additional notes for server implementations

Retrieving the BODYSTRUCTURE data item should be efficient. IMAP4 clients will expect fetching the BODYSTRUCTURE to cost no more than fetching the ENVELOPE. Slow implementations, like parsing the stored message for each request, are likely to result in a poor user experience.

8. Security Considerations

TODO Security

9. IANA Considerations

This document has no IANA actions.

Acknowledgments

TODO acknowledge.

Author's Address

Ricardo Signes
Fastmail
Email: rjbs@semiotic.systems