

JMAP
Internet-Draft
Intended status: Standards Track
Expires: 2 February 2026

N.M. Jenkins, Ed.
Fastmail
1 August 2025

JSON Meta Application Protocol (JMAP) Email Delivery Push Notifications
draft-ietf-jmap-emailpush-01

Abstract

This document specifies an extension to the JSON Meta Application Protocol (JMAP) that allows clients to receive an object via the JMAP push channel whenever a new email is delivered that matches a client-defined filter. The object can also include properties from the email, to allow a user notification to be displayed without having to make a further network request.

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 2 February 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
1.1. Notational Conventions	3
1.2. Terminology	3
2. Addition to the Capabilities Object	3
2.1. urn:ietf:params:jmap:emailpush	3
3. The EmailPush object	4
4. Extension to the PushSubscription object	4
5. Example	5
6. Security Considerations	8
7. IANA Considerations	8
7.1. JMAP Capability Registration for "emailpush"	8
8. Normative References	8
9. Informative References	9
Author's Address	9

1. Introduction

The JSON Meta Application Protocol (JMAP) [RFC8620] defines a mechanism for creating a push subscription, allowing a client to be efficiently notified whenever the state on the server changes for a data type the client is interested in. This can be used to trigger a resync of that data in the client.

Many email clients wish to show the user a notification when a new email arrives. Email clients implementing JMAP for Mail [RFC8621] may subscribe to changes in the pseudo-type "EmailDelivery" to be notified just whenever a new email is delivered to the store (as opposed to whenever changes are made to the Email data in general, which may be caused by user actions such as reading, organising, or deleting their mail). However, this does not meet the needs of some email clients in constrained environments, particularly on mobile:

1. Data is pushed for all email deliveries, but the client needs to receive them only when a message arrives for which it intends to show the user a notification. For example, this is commonly only shown for messages delivered to the inbox by default, not messages filtered to other mailboxes.
2. The client needs to make at least one HTTP request, and sometimes more, to resynchronise its datastore before it can show a visible notification to the user. This can sometimes fail when there is poor connectivity. In such circumstances, the client is only able to tell the user a new message has arrived, not any of the details such as who sent it, or the subject.

To address these issues, this document defines a way to receive a new EmailPush object via the JMAP push subscription. Clients can request a filter be applied to precisely restrict which messages they are notified about. Clients may request certain properties of the email sent in the EmailPush object, so it has sufficient information to show the user a notification having to make any further requests to the server.

1.1. Notational Conventions

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.

Type signatures, examples, and property descriptions in this document follow the conventions established in Section 1.1 of [RFC8620].

1.2. Terminology

The same terminology is used in this document as in the core JMAP specification, see Section 1.6 of [RFC8620].

The term Email (with this specific capitalization) is used to refer to the data type defined in Section 4 of [RFC8621].

2. Addition to the Capabilities Object

The capabilities object is returned as part of the JMAP Session object; see Section 2 of [RFC8620]. This document defines an additional capability URI.

2.1. urn:ietf:params:jmap:emailpush

The urn:ietf:params:jmap:emailpush capability support for the extensions to PushSubscription described in this document. The value of this property in the JMAP Session "capabilities" property is an empty object.

Accounts for which email delivery push is supported MUST have this capability in the account's "accountCapabilities" property. The value is an empty object.

Note, this capability may be supported in servers or accounts that do not support the urn:ietf:params:jmap:mail capability — in such servers, notification of new emails is supported but any further interaction will require use of a different protocol, such as IMAP [RFC9051].

3. The EmailPush object

An **EmailPush** object represents a new message that has been delivered to the store. It has the following properties:

@type: String

This MUST be the string "EmailPush".

accountId: Id

The id of the Account this Email is in.

email: Email

An object with the properties of the Email object (as defined in Section 4.1 of [RFC8621]) that were requested by the client in the EmailPushConfig. Note, push channels often have strict restrictions on the size of data that can be sent. Properties MUST be added to this object in the order defined in the properties array of the associated EmailPushConfig. If adding a property would exceed the size limit for the push channel, the property MUST be omitted. Clients are RECOMMENDED to only request the properties they need for the visible notification, plus the "id" property if they want to then fetch further information.

4. Extension to the PushSubscription object

The PushSubscription object is defined in Section 7.2 of [RFC8620]. This document defines an extra property for this object:

emailPush: Id[EmailPushConfig]|null

If not null, this is a map. The keys are Account ids for accounts with support for the urn:ietf:params:jmap:emailpush capability. The server will push EmailPush objects for these accounts when a new message arrives, in accordance with the associated EmailPushConfig options for the account.

An **EmailPushConfig** object has the following properties:

filter: FilterOperator|FilterCondition|null

A filter to apply to determine which new messages to push an EmailPush object for. The FilterCondition is as defined in Section 4.4.1 of [RFC8621].

If urn:ietf:params:jmap:mail capability is not supported for an account, support for the "inMailbox" and "inMailboxOtherThan" filter conditions is OPTIONAL. However, if the server supports IMAP access to the same account with the ObjectID extension [RFC8474], it MUST support using mailbox ids discovered via IMAP with these filter conditions.

properties: String[]

The list of properties to include in the "email" value of the EmailPush object. This is the same as the properties argument in "Email/get", as defined in Section 4.2 of [RFC8621].

If urn:ietf:params:jmap:mail capability is supported the server MUST support all the properties it supports for the "Email/get" method. Otherwise, it MUST support the following properties:

- * keywords
- * from
- * to
- * cc
- * subject
- * preview
- * receivedAt
- * sentAt

And if the server supports IMAP ObjectID [RFC8474], it MUST also support the following properties:

- * id
- * threadId
- * mailboxIds

urgency: String (default: "normal")

The urgency with which to send push notifications containing the EmailPush object when sending over Web Push [RFC8030]. The value must be an urgency-option, as defined in Section 5.3 of [RFC8030].

5. Example

An email client connects to a server over IMAP ([RFC9051]) and after authenticating, sees the OBJECTID [RFC8474] and JMAPACCESS [RFC9698] capabilities advertised.

The JMAPACCESS IMAP capability gives it the URL for the JMAP session resource, and guarantees the client can use the same credentials to authenticate that it used for IMAP. The client fetches the JMAP Session object and finds the following JMAP capabilities present:

```
{
  "urn:ietf:params:jmap:core": {
    "maxCallsInRequest": 50,
    "maxConcurrentUpload": 10,
    "maxObjectsInGet": 4096,
    "maxObjectsInSet": 4096,
    "collationAlgorithms": [
      "i;ascii-numeric",
      "i;ascii-casemap",
      "i;octet"
    ],
    "maxSizeUpload": 250000000,
    "maxSizeRequest": 10000000,
    "maxConcurrentRequests": 10
  },
  "urn:ietf:params:jmap:emailpush": {},
}
```

Figure 1: The "capabilities" property of a JMAP Session object

The server doesn't support `urn:ietf:params:jmap:mail` yet, so the client can't switch over to using JMAP fully instead of IMAP, but it does support `urn:ietf:params:jmap:emailpush`, so the client can now use JMAP to receive push notifications.

The user has asked the client to notify it of any message delivered to the inbox, plus any message delivered to another folder that has the `$notify` keyword set. The client discovers the mailbox id of the inbox over IMAP using `ObjectId`, then creates a push subscription by making the following API request:

```

[[ "PushSubscription/set", {
  "create": {
    "4f29": {
      "deviceClientId": "a889-ffea-910",
      "url": "https://push.example.com/WmYD3enMWsSOE0ndiBZfYeg54j4eWHfI",
      "keys": {
        "p256dh": "BM0HH37xD8sEh8NeiHulhFuwxqeeruxPlX5S7XGwGjHY5UT5yZFQKAiihFQLP6GuEQP
QOtD9z28HRYV7elnndh8",
        "auth": "uwPFKACOLBipGPTVz4UCDg"
      },
      "types": [],
      "emailPush": {
        "A1412": {
          "filter": {
            "operator": "OR",
            "conditions": [
              { "inMailbox": "674cc240-95db-49ce-a8a2-054f4f733095" },
              { "hasKeyword": "$notify" }
            ]
          },
          "properties": ["from", "subject", "id"]
        }
      }
    }
  },
  "0" ]]
```

Figure 2: "methodCalls" Property of a JMAP Request

The server creates the push subscription and immediately pushes a PushVerification object to the client. The client updates the PushSubscription object with this value to validate it as per Section 7.2.2 of [RFC8620]. With this done, the server will now push an EmailPush object whenever a new message arrives that's delivered to the mailbox with id "674cc240-95db-49ce-a8a2-054f4f733095", or has the "\$notify" keyword. As the "types" property is the empty array, the server will not push any StateChange objects. For example, a message arrives and the server pushes the following:

```

{
  "@type": "EmailPush",
  "accountId": "A1412",
  "email": {
    "from": [{ "name": "John Smith", "email": "john@example.com" }],
    "subject": "Surprise birthday party for Jane tonight",
    "id": "M699bb151ca4cbf1c24364a68"
  }
}
```

Figure 3: An EmailPush object

6. Security Considerations

All security considerations of JMAP [RFC8620] apply to this specification. Additional considerations are detailed below.

With the push defined in JMAP Core, no user data travels over the push channel, only state strings. The EmailPush object on the other hand contains sensitive user data. As specified in Section 8.7 of [RFC8620], to ensure confidentiality and integrity of the user's data, the client MUST specify encryption keys when establishing the PushSubscription and ignore any push notification received that is not encrypted with those keys.

7. IANA Considerations

7.1. JMAP Capability Registration for "emailpush"

IANA will register the "emailpush" JMAP Capability as follows:

Capability Name: urn:ietf:params:jmap:emailpush
Specification document: this document
Intended use: common
Change Controller: IETF
Security and privacy considerations: this document, Section 6

8. 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>>.
- [RFC8030] Thomson, M., Damaggio, E., and B. Raymor, Ed., "Generic Event Delivery Using HTTP Push", RFC 8030, DOI 10.17487/RFC8030, December 2016, <<https://www.rfc-editor.org/info/rfc8030>>.
- [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>>.
- [RFC8620] Jenkins, N. and C. Newman, "The JSON Meta Application Protocol (JMAP)", RFC 8620, DOI 10.17487/RFC8620, July 2019, <<https://www.rfc-editor.org/info/rfc8620>>.

- [RFC8621] Jenkins, N. and C. Newman, "The JSON Meta Application Protocol (JMAP) for Mail", RFC 8621, DOI 10.17487/RFC8621, August 2019, <<https://www.rfc-editor.org/info/rfc8621>>.

9. Informative References

- [RFC8474] Gondwana, B., Ed., "IMAP Extension for Object Identifiers", RFC 8474, DOI 10.17487/RFC8474, September 2018, <<https://www.rfc-editor.org/info/rfc8474>>.
- [RFC9051] Melnikov, A., Ed. and B. Leiba, Ed., "Internet Message Access Protocol (IMAP) - Version 4rev2", RFC 9051, DOI 10.17487/RFC9051, August 2021, <<https://www.rfc-editor.org/info/rfc9051>>.
- [RFC9698] Gulbrandsen, A. and B. Gondwana, "The JMAPACCESS Extension for IMAP", RFC 9698, DOI 10.17487/RFC9698, January 2025, <<https://www.rfc-editor.org/info/rfc9698>>.

Author's Address

Neil Jenkins (editor)
Fastmail
PO Box 234, Collins St West
Melbourne VIC 8007
Australia
Email: neilj@fastmailteam.com
URI: <https://www.fastmail.com>