

# JMAP

IETF 102 / 16 Jul 2018

# Note Well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (<https://www.ietf.org/contact/ombudsteam/>) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

- BCP 9 (Internet Standards Process)
- BCP 25 (Working Group processes)
- BCP 25 (Anti-Harassment Procedures)
- BCP 54 (Code of Conduct)
- BCP 78 (Copyright)
- BCP 79 (Patents, Participation)
- <https://www.ietf.org/privacy-policy/> (Privacy Policy)

# Agenda

- JMAP Core: 15 min
- JMAP Mail: 15 min
- JMAP Extensions: 20 min
- Other business: 10 min

# JMAP Core

- registry of error types
- push subscription
- explicit query totals
- steps to last call!

# Registry of error types

- Q to settle today: do we need one?
- Only useful when different client behaviours can be articulated based on the distinction between those errors.
- Unlikely whole new classes of errors will appear; we have experience from previous protocols.
- But, if new errors are added, useful to have them all documented in one place.
- **<https://github.com/jmapio/jmap/issues/89>**

# Push subscriptions

- Quite a few changes since IETF101:
  - API is more like a standard object type
  - Can restrict pushes to just types you are interested in
- Remaining Q: add ability to request pushes only if before or after state of changed record matches conditions?
  - Good for mobile clients to reduce (possibly by a lot) the number of pushes they receive.
  - Could be expensive for server to calculate.
  - Might have to be optional for servers to implement?
- **<https://github.com/jmapio/jmap/issues/222>**

# Explicitly query totals

- See recent discussion on mailing list
- Philosophically
  - server shouldn't do work that the client doesn't need (see: \Recent, UNSEEN on SELECT)
  - client should explicitly ask for what it needs, rather than server doing speculative work or having to be told what not to do (see: CalDAV implicit scheduling)
- Disadvantage: complexity

# Last Call!

- No other open issues remain.
- Draft has been reviewed by various people (at various revision stages).
- There is sufficient implementation experience to prove the protocol is both implementable and efficient to use.
- Plan to go to last call following IETF 102; any comments?

# JMAP Mail

- Email groups in addresses
- Things that make sense as extensions removed
- steps to last call!

# Email groups

To: Friends: tom@example.com, Richard Smith  
<dick@example.com>; harry@example.com



Parse "asAddresses"

```
[{
  name: "Friends"
  email: null,
}, {
  name: null,
  email: "tom@example.com"
}, {
  name: "Richard Smith"
  email: "dick@example.com"
}, {
  name: null,
  email: null
}, {
  name: null,
  email: "harry@example.com"
}]
```

**Group name**

**Group end**

# Things to make into extensions

- Vacation response
  - Making it part of the same spec made it impossible (within spec) to expose a shared mail account without sharing access to setting vacation response to that account.
  - There are use cases for JMAP Mail where vacation response makes no sense (e.g. mailing list archive access).
- Search within attachments
  - Was always optional.
  - Moving this to an extension means the server advertises whether it supports it, which is better.

# Last Call!

- No other open issues remain.
- Draft has been reviewed by various people (at various revision stages).
- There is sufficient implementation experience to prove the protocol is both implementable and efficient to use.
- Plan to go to last call following IETF 102; any comments?

# JMAP Extensions

- Do we need to update the charter?
- draft-murchison-jmap-websocket
- vacation response - neilj
- full sieve - who?
- MDN - who?
- single HTML body - who?
- search inside attachments - who?

# Do we need to update the charter?

- Our deliverables only currently include core and mail, plus some supporting documents and tools.
- We always planned to re-charter for Calendar and Contacts, but maybe a general “extensions to JMAP” extra point in the deliverables for this working group makes sense.

draft-murchison-jmap-websocket



# draft-murchison-imap-websocket

- JMAP binding over a WebSocket transport layer
- “jmap” WebSocket Sub-Protocol
- JMAP API requests only - no upload or download
- JMAP over WebSocket messages are JSON-only
- Bi-directional compression of messages via WebSocket Per-Message Compression

# draft-murchison-imap-websocket

- Less overhead than JMAP over HTTP – each request doesn't have to be authenticated
- HTTP Authentication done once as part of the WebSocket handshake
- Standard WebSocket handshake over HTTP/1.1 (RFC 6455) or HTTP/2 (draft-ietf-httpbis-h2-websockets)

# draft-murchison-imap-websocket

- Discovering Support for JMAP over WebSocket:
  - 'wsUrl' property in JMAP Session Object
  - Are there any other parameters/limits required?
  - Do we need a 'urn:ietf:params:jmap:websocket' capability object?

# draft-murchison-imap-websocket

- Example HTTP/1.1 Discovery:

Client >>

GET /jmap/ HTTP/1.1

Host: server.example.com

Authorization: Basic Zm9vOmJhcg==

# draft-murchison-imap-websocket

<< Server

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

```
{  
  "username": "foo",  
  "accounts": { .... },  
  "capabilities": { .... },  
  "apiUrl": "/jmap/",  
  "downloadUrl": "/jmap/download/{accountId}/{blobId}/{name}",  
  "uploadUrl": "/jmap/upload/{accountId}/",  
  "wsUrl": "wss://localhost/jmap/ws/"  
}
```

# draft-murchison-imap-websocket

- Example HTTP/1.1 handshake:

Client >>

GET /jmap/ws/ HTTP/1.1

Host: server.example.com

Upgrade: websocket

Connection: Upgrade

**Authorization: Basic Zm9vOmJhcg==**

Sec-WebSocket-Key: dGhllHNhbXBsZSBub25jZQ==

**Sec-WebSocket-Protocol: jmap**

Sec-WebSocket-Version: 13

**Sec-WebSocket-Extensions: permessage-deflate**

# draft-murchison-imap-websocket

<< Server

HTTP/1.1 101 Switching Protocols

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

**Sec-WebSocket-Protocol: imap**

**Sec-WebSocket-Extensions: permmessage-deflate**

[WebSocket connection established]

# draft-murchison-imap-websocket

- Example HTTP/2 Handshake

<< Server

SETTINGS

SETTINGS\_ENABLE\_CONNECT\_PROTOCOL = 1

# draft-murchison-imap-websocket

Client >>

HEADERS + END\_HEADERS

:method = CONNECT

:protocol = websocket

:scheme = https

:path = /jmap/ws/

:authority = server.example.com

**authorization = Basic Zm9vOmJhcG==**

**sec-websocket-protocol = jmap**

sec-websocket-version = 13

origin = http://www.example.com

# draft-murchison-imap-websocket

<< Server

HEADERS + END\_HEADERS

:status = 200

**sec-websocket-protocol = jmap**

[WebSocket connection established]

# draft-murchison-imap-websocket

- Top-level errors (e.g. incorrectly formatted Request Object):
  - Can't return HTTP response
  - MUST return a Problem Details JSON Object (RFC 7807)
  - Can the client auto-detect Problem Details vs Response Object? If not, how does the server indicate?

# draft-murchison-imap-websocket

- Should we allow out of order processing of requests?
  - This would require an 'id' property on requests and responses
- Should we allow push notifications?
  - Can the client auto-detect Response vs StateChange objects or do we need to add a 'type' property to each top-level object?

# draft-murchison-imap-websocket

- Other issues?

# Vacation Response

- As previously in the Mail spec
- Neil will write

# Edit Sieve

- Any interest?
- Any volunteers to write?
- Advantage over “just vacation”:
  - Can refer to existing docs
  - Much more powerful
- Disadvantage over “just vacation”:
  - Complexity
  - Server may not have sieve already

# Generating MDN

- Any interest?
- Any volunteers to write?

# Single HTML Body

- Any interest?
- Any volunteers to write?
- General concept:
  - Make it really simple to write dumb clients.
  - Take all the inline images and make them into data URIs, replacing cid references.
  - Mash all the text together as divs into a single HTML doc.
  - Send to the client with “render this in a div/iframe, and you’re done”.

# Search inside attachments

- Any interest?
- Volunteers to write?

Any Other Business?