Network Working Group
Internet-Draft
Updates: 8620 (if approved)
Intended status: Standards Track
Expires: 25 August 2026

B. Gondwana
Fastmail

21 February 2026

JMAP Blob Extensions
draft-gondwana-jmap-blobext-00

Abstract

The JMAP base protocol (RFC8620) provides the ability to upload and download arbitrary binary data. This binary data is called a "blob", and can be used in all other JMAP extensions.

The JMAP blob extension (RFC9404) added additional ways to create and access blobs by making inline method calls within a standard JMAP request.

This extension adds more methods to work with blobs, including handling large blobs by processing them in chunks (building on RFC9404's blob construction support), and providing ways to efficiently describe small changes.

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 25 August 2026.

Copyright Notice

Table of Contents

1. Introduction

The JMAP Blob extension ([JMAP-BLOB] — JSON Blob Management) offers additional ways to create blobs, and query where they are used.

This extension builds on that work, offering ways to find more information about the internal structure of the server's blob store in order to work efficiently with it, and ways to efficiently transfer small changes to an existing blob without copying all the data.

2. Addition to the Capabilities Object

The capabilities object is returned as part of the JMAP Session object; see [JMAP-CORE], Section 2.

This document defines an additional capability URI.

2.1. urn:ietf:params:jmap:blobext

The capability urn:ietf:params:jmap:blobext being present in the "accountCapabilities" property of an account represents support for these extended properties on that account. If this capability is present in one or more "accountCapabilities" properties then the server MUST also include the key in the "capabilities" property.

The value of this property in the JMAP session "capabilities" property MUST be an empty object.

The value of this property in an account's "accountCapabilities" property is an object that MUST contain the following information on server capabilities and permissions for that account:

* resumableUploadUrl: "String|null"

  If present, a [URI-TEMPLATE] which supports [HTTP-RESUMABLE-UPLOADS]. This MAY be the same as the uploadUrl, and has the same keys.

* chunkSize: "UnsignedInt|null"

  The size in octets which the server users to split large files into chunks. If a client uploads blobs with exactly this size except for the final chunk, and uses Blob/upload with data blobs containing these chunks, it is expected that the server can optimise these chunks. Servers MUST allow other sizes for the individual data blocks in Blob/upload though, and will then choose whether to store them as an array of blobs still, or to combine them.

2.1.1. Capability Example

TODO

2.2. Additions to Blob/get

Blob/get returns an additional parameter (returned by default when this capability is included in using)

* chunks: Array[DataSourceObject]

An array of one or more data source objects. The file is created by concatenating the DataSourceObjects in the listed order. While it is expected that DataSourceObjects will include the entire blob and offset, it is legal for the server to give offset and length values which do not cover the entire blob in the same way that Blob/upload can, and client MUST use the length and offset to figure out what to append to the resulting file.

* rdiffSignature: blobId

  Returns a blobId which can be fetched to download the rdiff signature of the blobId being fetched.

2.3. Additions to DataSourceObject

All DataSourceObjects (including data: objects) can have an offset and length to trim the final values, though it's just a waste of data when applied to data: objects. This allows client and server code to be more consistent for all types.

Every DataSourceObject (on both upload and get) is extended to include any supported digest which is requested, so a client could request 'digest:sha' and every DataSourceObject would include the 'digest:sha' as well as the top level, allowing each blob to be separately validated.

There is a new parameter to each DataSourceObject

* rdiffPatch: blobId

  If provided, apply the patch in the given blobId to the result of this DataSourceObject AFTER applying any length and offset. If a digest value is given, the digest is calculated on the resulting octets AFTER applying the patch. The sequence is:

  1) decode any base64 or fetch any blob 2) apply and offset or length to find the raw octets 3) apply any rdiffPatch to get new raw octets 4) check any supported digest: value that was provided to ensure the result matches

NOTE: do we want to add any ability to create zip/tar files out of multiple blobs here?

3. Security Considerations

Other than the security considerations present in [JMAP-CORE] and [JMAP-BLOB], this document adds the security considerations around ridff handling, including server load and denial of service risks, and any ability to smuggle content through security scanners by splitting it into multiple parts is exacerbated by the ability to split things into parts and use rdiff patches.

4. IANA Considerations

4.1. JMAP Capability Registration for urn:ietf:params:jmap:blobext

IANA is requested to register the "Blob Extended" Capability as follows

Capability: urn:ietf:params:jmap:filenode

Reference: this document

5. TODO

* Look into ZIP files, compression, etc.

6. Changes

EDITOR: please remove this section before publication.

The source of this document exists on github at: https://github.com/brong/draft-gondwana-jmap-blobext/

draft-gondwana-jmap-blobext-00

* initial proposal

7. Acknowledgements

TODO

{backmatter}

8. Normative References

[HTTP-RESUMABLE-UPLOADS]
Kleidl, M., Zhang, G., and L. Pardue, "Resumable Uploads for HTTP", Work in Progress, Internet-Draft, draft-ietf-httpbis-resumable-upload-10, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-resumable-upload-10>.

[JMAP-BLOB]
Gondwana, B., Ed., "JSON Meta Application Protocol (JMAP) Blob Management Extension", RFC 9404, DOI 10.17487/RFC9404, August 2023, <https://www.rfc-editor.org/rfc/rfc9404>.

[JMAP-CORE]
Jenkins, N. and C. Newman, "The JSON Meta Application Protocol (JMAP)", RFC 8620, DOI 10.17487/RFC8620, July 2019, <https://www.rfc-editor.org/rfc/rfc8620>.

[URI-TEMPLATE]
Gregorio, J., Fielding, R., Hadley, M., Nottingham, M., and D. Orchard, "URI Template", RFC 6570, DOI 10.17487/RFC6570, March 2012, <https://www.rfc-editor.org/rfc/rfc6570>.

Author’s Address

Bron Gondwana
Fastmail
Email: brong@fastmailteam.com