Resumable Uploads

draft-ietf-httpbis-resumable-upload
IETF 119: March 19, 2024
Marius Kleidl

Implementations

- Servers implementations in .NET and Go
- Client implementations in JavaScript (Node.js and browser) and Swift (iOS)
- Load testing tool for benchmarking servers
- Specification conformity checker for servers
- https://github.com/tus/rufh-implementations/

Changes in -03 draft

- Add upload progress notifications via informational responses.
- Explain the use of empty requests for creation uploads and appending.
- Allow 200 status codes for offset retrieval.
- Include request filtering and resource exhaustion attacks in security considerations.

PATCH request media type (#2610)

- PATCH needs a media type in Content-Type describing the patch document
- `application/octet-stream` is not applicable
- Proposal: New `application/partial-upload` media type (#2743)
  - Request content is partial data from file
  - Patch is applied by appending the content to the upload resource
  - Similar media type exists: `application/vnd.adobe.partial-upload`

Upload limits (#2741, #2747)

- Servers usually place limits on the uploads, e.g.:
  - Maximum size for the entire upload
  - Minimum or maximum size for a single POST/PATCH requests
  - Lifetime of the upload resource
  - Maximum number of concurrent upload requests (potentially for parallelized uploads in the future)
- How can the client discover those?

Upload limits (#2741, #2747)

- Servers announced limits in responses to POST and HEAD requests
- Option 1: Separate header fields

```
Upload-Size-Limit: 10000000
Upload-Expires: 1710833400 (maybe Sunset or another header field can be used?)
Upload-Min-Append: 5000
Upload-Max-Append: 10000000
```

- Option 2: One header fields with dictionary

```
Upload-Limit: size=10000000, expires=1710833400, min-append=5000, max-append=10000000
```

  - Requires a registry of upload limits

Interrupted PATCH requests (#2760)

RFC 5789: The server MUST apply the entire set of changes atomically and never provide [...] a partially modified representation. If the entire patch document cannot be successfully applied, then the server MUST NOT apply any of the changes.

- What happens if a PATCH request for appending data is interrupted?
- Can the server save the received data? If not, the client must retransmit
- Resource is never in an invalid state and upload can always resume

Content coding of upload resource (#2674, #2746)

- Example: Upload resource is created with gzip coding

```
POST /uploads HTTP/1.1
Upload-Complete: ?0
Content-Type: application/json
Content-Encoding: gzip

[partial data]
```

- What content coding for resuming the upload? Should it target the same representation?

```
PATCH /uploads/1 HTTP/1.1
Upload-Complete: ?1
Upload-Offset: 500
Content-Encoding: gzip # (?)

[remaining data]
```

Other open issues

- Error handling for Upload Creation Procedure (#2596)
- Interaction with Digest Fields (#2748)
- Indicating subsequent resources with Content-Location (#2744)
- Require that upload offset does not decrease (#2695)
- Prioritization of uploads (#2242)
- Fetch API proposal (WHATWG #1626)