Resumable Uploads

draft-ietf-httpbis-resumable-upload
IETF 120: July 22, 2024
Marius Kleidl

What’s new in -04?

- `application/partial-upload` media type for PATCH requests
- Upload limits (max/min size, max/min request size, upload expiration)
- Guidelines for handling content and transfer codings
- Guidelines for integrity checks using digest fields
- Problem types for error responses

1) Upload limits

- When an upload is created, server can indicate limits in response: max/min size, max/min request size etc.

```
POST /upload HTTP/1.1
Host: example.com
Upload-Complete: ?0

HTTP/1.1 201 Created
Location: https://example.com/upload/b530ce8ff
Upload-Offset: 0
Upload-Limit: max-size=1000000000, min-append-size=5000000
```

- But client does not know limits upfront!

1) Discovering upload limits upfront (#2833)

- Proposal: Client can send OPTIONS request upfront

```
OPTIONS /upload HTTP/1.1
Host: example.com

HTTP/1.1 204 No Content
Upload-Limit: max-size=1000000000, min-append-size=5000000
```

- Browser may already send OPTIONS request for CORS preflight
- Thoughts?

2) Declaring upload length upfront (#2832)

- An upload may spread across multiple requests
- Server only knows the length once the upload is complete (Upload-Complete: ?1)
- Server cannot reject uploads early if they are too large
- Server cannot optimize upload storage depending on size

2) Declaring upload length upfront (#2832)

- Client could include advisory header in creation request (similar to Bikeshed-Length draft)
- Upload is still completed through `Upload-Complete: ?1`

```
POST /upload HTTP/1.1
Upload-Complete: ?0
Upload-Length: 500
Content-Length: 100
[100 bytes]

HTTP/1.1 201 Created
Location: https://example.com/upload/b530ce8ff
Upload-Offset: 100

PATCH /upload/b530ce8ff HTTP/1.1
Upload-Complete: ?0
Upload-Offset: 100
Content-Length: 100
[100 bytes]
```

- Server may reject upload if client uploads more or less data

3) Requesting digests from server (#2834)

- Client may include integrity fields (RFC 9530) when creating upload:

```
POST /upload HTTP/1.1
Host: example.com
Upload-Complete: ?0
Repr-Digest: sha-256=:RK/0qy18MlBSVnWgjwz6lZEWjP/lF5HF9bvEF8FabDg==:
```

- Server can check integrity when upload is completed
- But client must know digest upfront or use trailers (not always possible)

3) Requesting digests from server (#2834)

- Idea: Client requests digest from server when creating upload:

```
POST /upload HTTP/1.1
Host: example.com
Upload-Complete: ?0
Want-Repr-Digest: sha-256=1

[upload continues over multiple PATCH requests]

HTTP/1.1 204 No Content
Upload-Offset: 100
Upload-Complete: ?1
Repr-Digest: sha-256=:RK/0qy18MlBSVnWgjwz6lZEWjP/lF5HF9bvEF8FabDg==:
```

- Client can check integrity when upload is completed
- Is this a sensible use of integrity preference fields?

Next up

- Hopefully finish these topics soon
- Editorial improvements to draft
- Working group last call?