

# Byte Range PATCH

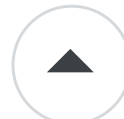
**A media type for writing at offsets (part 3)**

**Austin Wright, July 2025**

# Review: The Problem

- I only want to change the first four bytes of a file over HTTP
- Current solutions require prior coordination:
  - Endpoint-specific resource e.g.  
`POST http://example.com/logs.update`
  - Endpoint-specific URI format that identifies only selected bytes, e.g.  
`PUT http://example.com/logs?bytes=200-299`
  - RFC 9110 Content-Range PUT request

# Origin



28




I'm developing http client/server framework, and looking for the correct way to handle partial uploads (the same as for downloads using GET method with Range header).


But, HTTP PUT is not intended to be resumed. And PATCH method, as i know, doesn't accept Range header.

Is there any way to handle this in by HTTP standard (not using extension headers or etc)?

http upload

Share Improve this question Follow

edited Feb 2, 2023 at 20:23  
 E\_net4  
30.3k ● 13 ● 116 ● 154

asked Jan 7, 2014 at 10:32  
 Андрей Москвичёв  
1,612 ● 1 ● 18 ● 28



7



PATCH would be a logical method to choose for resumable uploads: it expects a media type that indicates how to change the target resource. Though not specifically defined as format to perform patching, `multipart/byteranges` specifies a byte range and the contents of that range, making it suitably well defined for PATCH payloads.

Example:

```
PATCH /document HTTP/1.1
Content-Type: multipart/byteranges; boundary=THIS_STRING_SEPARATES

--THIS_STRING_SEPARATES
Content-Type: text/plain
Content-Range: bytes 10-21/22

1234567890
--THIS_STRING_SEPARATES--
```

This example uploads twelve bytes at a ten-byte offset. `THIS_STRING_SEPARATES` is an arbitrary, user-picked delimiter, and should be randomly generated. Some headers omitted for brevity, each line is terminated with `CRLF`.

Share Improve this answer Follow

edited Nov 29, 2023 at 18:38

answered Jun 18, 2019 at 4:47

 awwright  
665 ● 5 ● 8

That's an interesting idea. How about writing this down in an Internet Draft? – Julian Reschke Jun 18, 2019 at 5:45

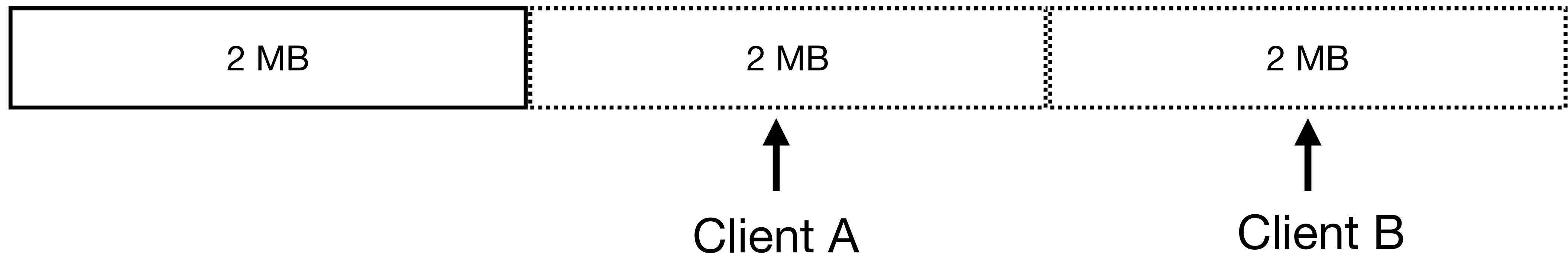
# Easy Solution

- Create a media type to use with PATCH
- Re-use the RFC 9110 Content-Range semantics in multipart/byteranges
- Also define message/byterange and application/byteranges, since decoding multipart messages is unpopular

# Issue 1: Non-contiguous/sparse Resources

## Revisiting RFC 8673

- HTTP resources that are only defined over part of their contents, enables parallel upload & “forgetting” regions of data (shift buffers, e.g. only keep the last 1GB of logs)
- Needs ability to query available regions, and/or which regions remain to be uploaded



# Issue 1: Non-contiguous/sparse Resources

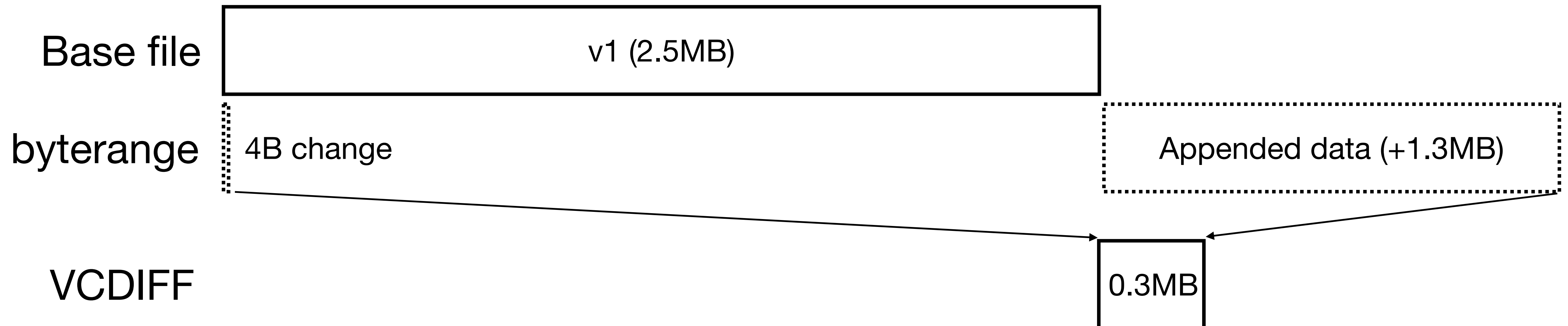
## Revisiting RFC 8673

- Proposed resolution: Out of scope for document, re-visit in HTTP WG after RFC
- Related work: [RFC 8673: HTTP Random Access and Live Content](#)

# Issue 2: Media Type Registrations

## Give VCDIFF (RFC 3284) a media type

- For very complicated changes to arbitrary files such as splicing, but requires access to their full contents
  - Uses a compression-like algorithm: In a survey of existing implementations, I was able to append 1.3MB of data to a 2.5MB .wav audio file as a 0.3MB patch



# Issue 3: An “Append” Media Type

- Add a media type whose entire semantics is “bytes to append to the resource”
- Advertises the need for a parser
- Useful for writing logs!
  - Advertises need for separate “GET /log” and “POST /append-to-log” endpoints.