HTTP Random Access and Live Resources

IETF 100

Barbara Stark
bs7652@att.com

Darshak Thakore
d.thakore@cablelabs.com

Craig Pratt
pratt@acm.org / craig@ecaspia.com
Use existing bytes Range Unit with “very large” numbers (draft-ietf-httpbis-rand-access-live)

- Clients use existing Range semantics to determine accessible bytes:
  
  ```
  HEAD /my_resource HTTP/1.1
  Range: bytes=0-
  
  HTTP/1.1 206 Partial Content
  Content-Range: bytes 0-99408383/*
  ```

  or GET...

  ```
  HTTP/1.1 206 Partial Content
  Content-Range: bytes 99400000-9999999999999/*
  ```

  “*” Indicates the representation length is unknown

- and use Very Large numbers to indicate an indeterminate endpoint:

  ```
  GET /my_resource HTTP/1.1
  Range: bytes=99400000-9999999999999
  
  HTTP/1.1 206 Partial Content
  Content-Range: bytes 99400000-9999999999999/*
  ```

  Request starts at point before “current length” (e.g. for “fast start”)

  ```
  Transfer-Encoding: chunked
  ```

  Server returns the same “Large Number” the Client provided to indicate it’s including “live” content

  Server returns the current length of the representation, per RFC7233

  Client provides “Large Number” (>>current representation len) to indicate it supports draft semantics
Interoperability with Servers/Proxies that do not support draft semantics

- Client uses existing semantics to determine accessible byte range (as before):
  
  HEAD /my_resource HTTP/1.1
  Range: bytes=0-
  
  HTTP/1.1 206 Partial Content
  Content-Range: bytes 0-99408383/*

- Server instead returns the current length:
  
  GET /my_resource HTTP/1.1
  Range: bytes=99400000-9999999999999
  
  HTTP/1.1 206 Partial Content
  Content-Range: bytes 99400000-99410000/*
  Transfer-Encoding: chunked

  "Large Number" (>>current representation len) to indicate it supports draft semantics

  Server returns the current representation length, indicating to the client that it doesn’t support live range requests

  Request starts at point before “current length” (e.g. for “fast start”)

  "*" Indicates the representation length is unknown

  Server returns the current length of the representation, per RFC7233

  or GET...
Prototype Server

- Implemented a simple HTTP 1.1 Server compliant with the draft RFC hosting a continuously-buffering representation of a live NASA-TV stream
  - Live resource directly accessible via
    - http://ietf100.ecaspia.com:8000/live/nasatv.ts
  - Varnish reverse proxy (v4.1.1)
    - http://ietf100.ecaspia.com:6081/live/nasatv.ts
  - Squid reverse proxy (3.5.12)
    - http://ietf100.ecaspia.com:3128/live/nasatv.ts
  - Source jar:
    - http://ietf100.ecaspia.com:8000/static/live-range-server.src.jar
  - Runtime jar:
    - http://ietf100.ecaspia.com:8000/static/live-range-server.jar
Direct Access Example Session

$ curl -I -H "Range: bytes=0-" http://ietf100.ecaspia.com:8000/live/nasatv.ts
HTTP/1.1 206 Partial Content
Content-range: 0-15622047/*
Date: Thu, 16 Nov 2017 06:00:21 GMT
Content-type: video/mp2t
Accept-ranges: bytes
Cache-control: max-age=3600

Server returning the same large value (>>>15622047) indicates the server supports live-range requests on this resource

$ curl -v -H "Range: bytes=15600000-99999999999"
http://ietf100.ecaspia.com:8000/live/nasatv.ts > out.ts
< HTTP/1.1 206 Partial Content
< Content-range: 15600000-99999999999/*
< Date: Thu, 16 Nov 2017 06:00:50 GMT
< Transfer-encoding: chunked
< Content-type: video/mp2t
< Accept-ranges: bytes
< Cache-control: max-age=3600

"*" Indicates the representation length is unknown. Can check to see if the server supports live-range RFC
Direct Access Example Session (cont)

- Server feeds data as it's made available:

  Response: 206

  ###################################################################
  #.............................#.............................#...........
  #.............................#.............................#...........
  #.............................#.............................#...........
  ###.............................#.............................#...........
  #.............................#.............................#...........
  #.............................#.............................#...........
  #.............................#.............................#...........

- Client (curl in this case) sees bitrate go up and down as data is aggregated:

  100 2744k  0  2744k  0  0  222k  0  --:--:--  0:00:12  --:--:--  136k
  100 3468k  0  3468k  0  0  141k  0  --:--:--  0:00:24  --:--:--  153k
  100 4156k  0  4156k  0  0  113k  0  --:--:--  0:00:36  --:--:--  0
  100 4848k  0  4848k  0  0  9928k 0  --:--:--  0:00:50  --:--:--  0
  100 6232k  0  6232k  0  0  97368 0  --:--:--  0:01:05  --:--:--  150k
  100 6940k  0  6940k  0  0  91083 0  --:--:--  0:01:18  --:--:--  0
  100 8328k  0  8328k  0  0  99803 0  --:--:--  0:01:25  --:--:--  314k

- Data will be continuously downloaded until the client cancels the transfer.
# Testing Matrix

<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varnish</td>
<td>Works (kinda)</td>
<td>Does caching but as static content (limited caching)</td>
</tr>
<tr>
<td>Varnish (range request support)</td>
<td>Works</td>
<td>Works as a straight thru proxy, no caching</td>
</tr>
<tr>
<td>Squid (reverse proxy)</td>
<td>Works</td>
<td>Does not cache, always get a cache miss, goes to origin server every time</td>
</tr>
<tr>
<td>CloudFlare</td>
<td>-</td>
<td>Still testing</td>
</tr>
</tbody>
</table>
Varnish Reverse Proxy Results - Standard Config

- Default Varnish configuration will make a live server look like a static content server. But is byte-wise coherent. e.g.

```bash
HTTP/1.1 206 Partial Content
Date: Thu, 16 Nov 2017 07:05:24 GMT
Content-type: video/mp2t
X-Varnish: 32772 3
Age: 126
Via: 1.1 varnish-v4
Accept-Ranges: bytes
Content-Range: bytes 0-24109496/24109497
Content-Length: 24109497
Connection: keep-alive

{ [1188 bytes data]
100 22.9M 100 22.9M 0 0 1089k 0 0:00:21 0:00:21 --:--:-- 1153k
```
Varnish Reverse Proxy Results + Range Support

- Server configured with Range support per https://info.varnish-software.com/blog/caching-partial-objects-varnish is fully functional with live ranges

```
< HTTP/1.1 206 Partial Content
< Date: Thu, 16 Nov 2017 07:33:20 GMT
< Content-type: video/mp2t
< Cache-control: max-age=3600
< X-Varnish: 32770 3
< Age: 13
< Via: 1.1 varnish-v4
< Content-Range: 0-99999999999/*
< Transfer-Encoding: chunked
< Connection: keep-alive
```
Varnish Reverse Proxy Results + Range Support
(cont)

- As is the case with direct [non-proxied] access, random access content is downloaded at higher bitrate, and reduces once the live point is hit

<table>
<thead>
<tr>
<th>File Size</th>
<th>Downloaded</th>
<th>Range</th>
<th>Size</th>
<th>Start Time</th>
<th>Elapsed Time</th>
<th>Range Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.1M</td>
<td>0</td>
<td>0</td>
<td>1464k</td>
<td>0:00:15</td>
<td></td>
<td>1532k</td>
</tr>
<tr>
<td>41.3M</td>
<td>0</td>
<td>0</td>
<td>1391k</td>
<td>0:00:30</td>
<td></td>
<td>1702k</td>
</tr>
<tr>
<td>61.8M</td>
<td>0</td>
<td>0</td>
<td>1392k</td>
<td>0:00:45</td>
<td></td>
<td>1303k</td>
</tr>
<tr>
<td>80.3M</td>
<td>0</td>
<td>0</td>
<td>1360k</td>
<td>0:01:00</td>
<td></td>
<td>1214k</td>
</tr>
<tr>
<td>96.6M</td>
<td>0</td>
<td>0</td>
<td>1311k</td>
<td>0:01:15</td>
<td></td>
<td>1040k</td>
</tr>
<tr>
<td>116M</td>
<td>0</td>
<td>0</td>
<td>1315k</td>
<td>0:01:30</td>
<td></td>
<td>1807k</td>
</tr>
<tr>
<td>137M</td>
<td>0</td>
<td>0</td>
<td>1336k</td>
<td>0:01:45</td>
<td></td>
<td>1427k</td>
</tr>
<tr>
<td>145M</td>
<td>0</td>
<td>0</td>
<td>1230k</td>
<td>0:02:00</td>
<td></td>
<td>147k</td>
</tr>
<tr>
<td>145M</td>
<td>0</td>
<td>0</td>
<td>1106k</td>
<td>0:02:15</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>147M</td>
<td>0</td>
<td>0</td>
<td>1004k</td>
<td>0:02:30</td>
<td></td>
<td>132k</td>
</tr>
<tr>
<td>147M</td>
<td>0</td>
<td>0</td>
<td>914k</td>
<td>0:02:45</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>149M</td>
<td>0</td>
<td>0</td>
<td>846k</td>
<td>0:03:00</td>
<td></td>
<td>111k</td>
</tr>
<tr>
<td>149M</td>
<td>0</td>
<td>0</td>
<td>836k</td>
<td>0:03:02</td>
<td></td>
<td>137k</td>
</tr>
</tbody>
</table>

- Data returned is byte-wise coherent with the origin server

- Varnish may cache too much data without additional configuration
  - It appears to continue trying to buffer the entire requested range - probably needs to be bounded...
Squid Reverse Proxy Results
Standard Disk Caching Config

- Seems to support the live range semantics

```bash
< HTTP/1.1 206 Partial Content
< Date: Thu, 16 Nov 2017 07:33:20 GMT
< Content-type: video/mp2t
< Cache-control: max-age=3600
< X-Varnish: 32770 3
< Age: 13
< Via: 1.1 varnish-v4
< Content-Range: 0-99999999999/
< Transfer-Encoding: chunked
< Connection: keep-alive
```
Squid Reverse Proxy Results
Standard Disk Caching Config (cont)

- And supports live range transfer:

<table>
<thead>
<tr>
<th>Time</th>
<th>Size</th>
<th>Caching</th>
<th>Size</th>
<th>Coalescing</th>
<th>Time</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:00:15</td>
<td>1532k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:00:15</td>
<td>1807k</td>
</tr>
<tr>
<td>0:00:30</td>
<td>1702k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:00:30</td>
<td>1427k</td>
</tr>
<tr>
<td>0:01:00</td>
<td>1303k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:01:00</td>
<td>147k</td>
</tr>
<tr>
<td>0:01:15</td>
<td>1214k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:01:15</td>
<td>132k</td>
</tr>
<tr>
<td>0:01:45</td>
<td>1427k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:01:45</td>
<td>0</td>
</tr>
<tr>
<td>0:02:00</td>
<td>147k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:02:00</td>
<td>132k</td>
</tr>
<tr>
<td>0:02:15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:02:15</td>
<td>0</td>
</tr>
<tr>
<td>0:02:30</td>
<td>111k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:03:00</td>
<td>111k</td>
</tr>
<tr>
<td>0:03:02</td>
<td>137k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0:03:02</td>
<td>137k</td>
</tr>
</tbody>
</table>

- Caching and coalescing of byte ranges doesn't appear functional. But the data returned is byte-wise coherent with the origin server.