Effective HTTP Caching Part III: Public, Private and No-Store

msingh Follow
Dec 1, 2018 · 4 min read

As we saw in earlier blogs, `Cache-Control` header field is used to specify directives for caches along the request/response chain. A cache MUST obey the requirements of the Cache-Control directives. In this section, we will explore three such directives- `public, private and no-store` .

These can be used as both request and response directives but we will limit our discussion in context of response.

Public vs. Private

The “public” response directive indicates that **any cache MAY store** the response whereas the “private” response directive indicates that the response message is intended for a single user and **MUST NOT be stored by a shared cache.**

So private responses can’t be stored by CDN’s like AWS CloudFront but can be stored by the browser which is considered a private cache.

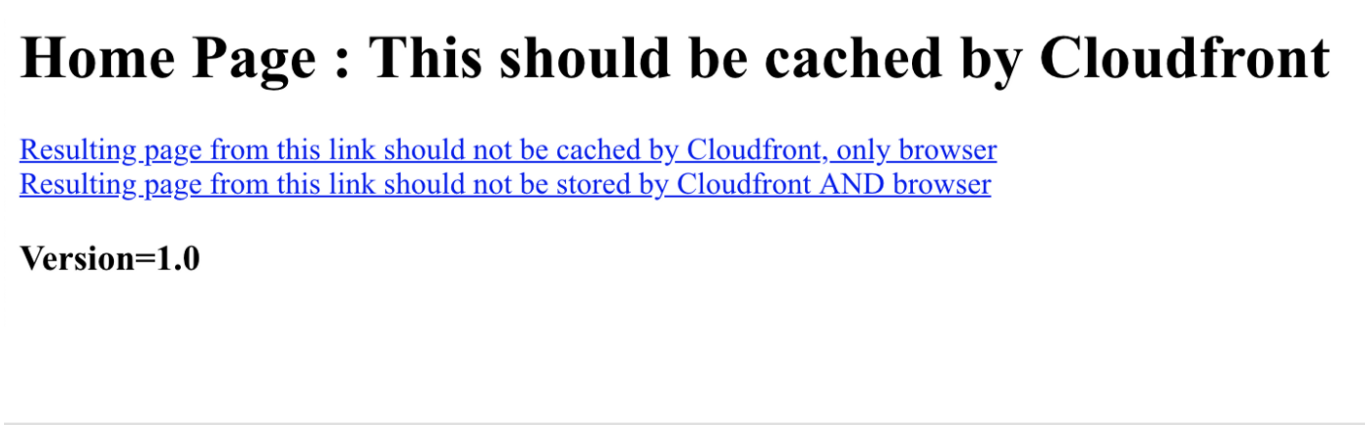
See it in action

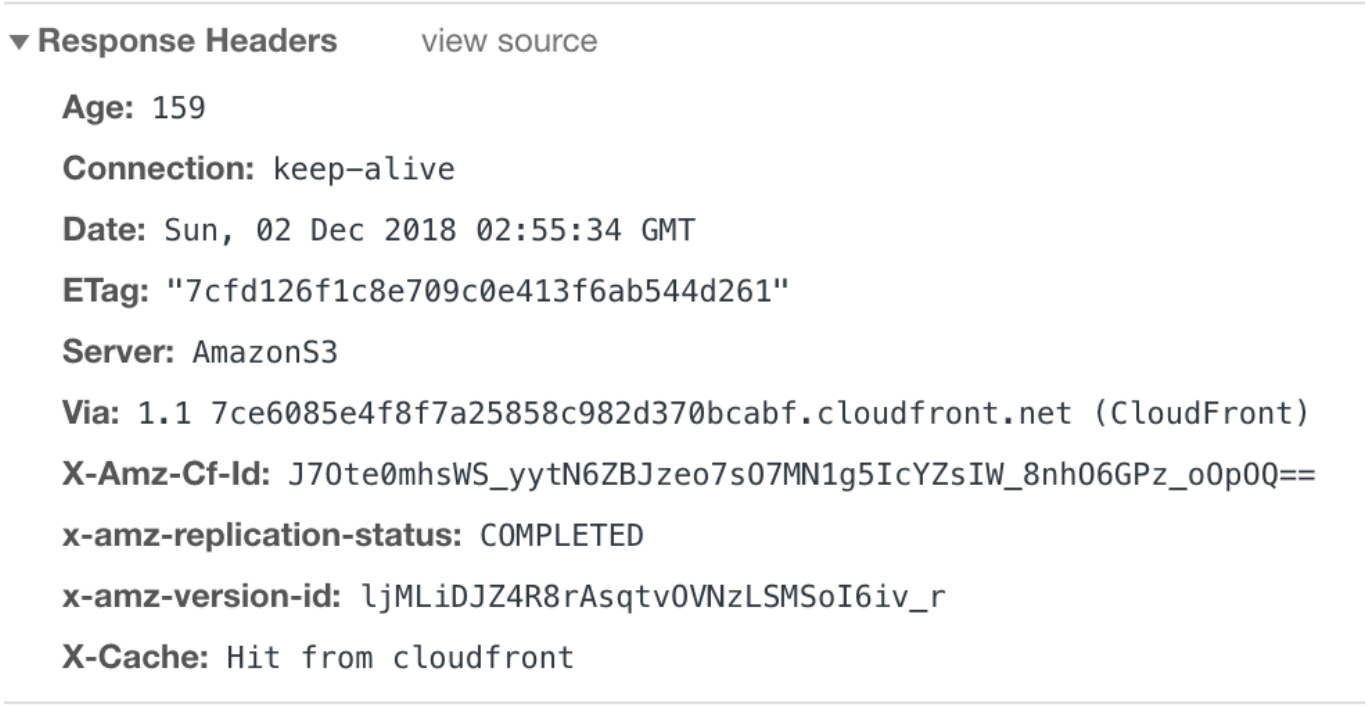
In order to visualize this, you would need an AWS account and setup CF as a front to serve your pages. Easiest way to get this up and running is using AWS S3 bucket as backing store for the pages to be served.

There are tonnes of tutorials on the internet about setting up CF distribution with S3 e.g. this one from AWS blogs

For this article, I hosted a simple “Home.html” in a S3 bucket with public read access to the page and created a CF distribution to serve this bucket, with cache default TTL as 24

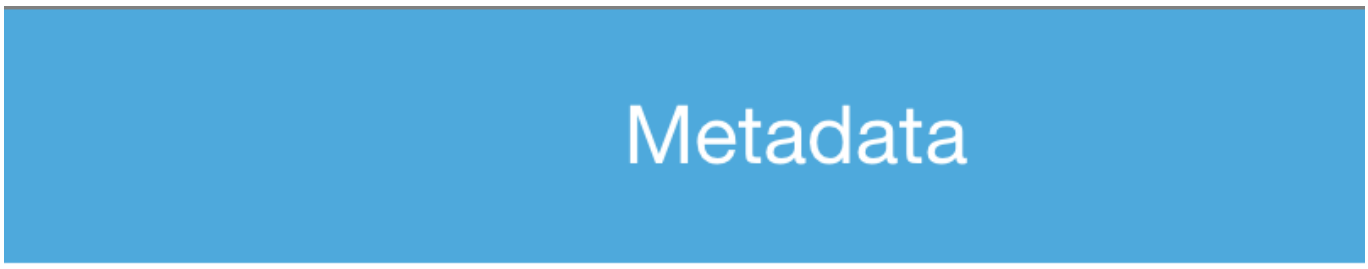
hours. Accessing the page (`CFdistributionBaseURL/home.html`) shows following .

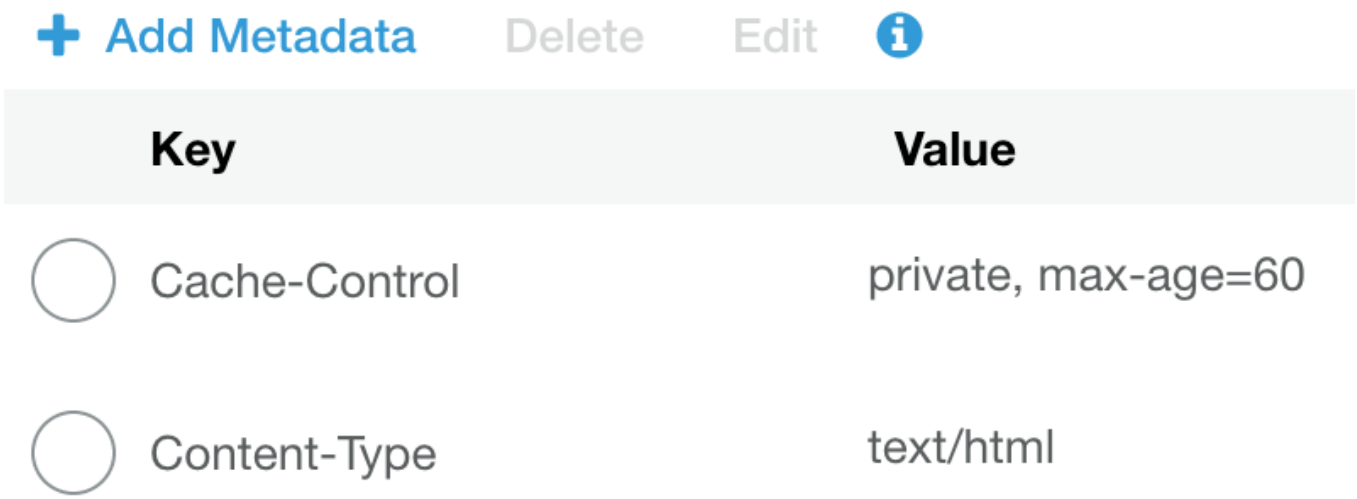




Public Cache : From the response header captured in Chrome, notice that the browser serves page from Cache after first access. (see additional header `X-Cache "Hit from cloudfront" .` Multiple refresh of this page would return `304 Not Modified`

Private Cache: The first link on the home page takes user to another page which is also uploaded to same S3 bucket but additionally has metadata configured (see image)

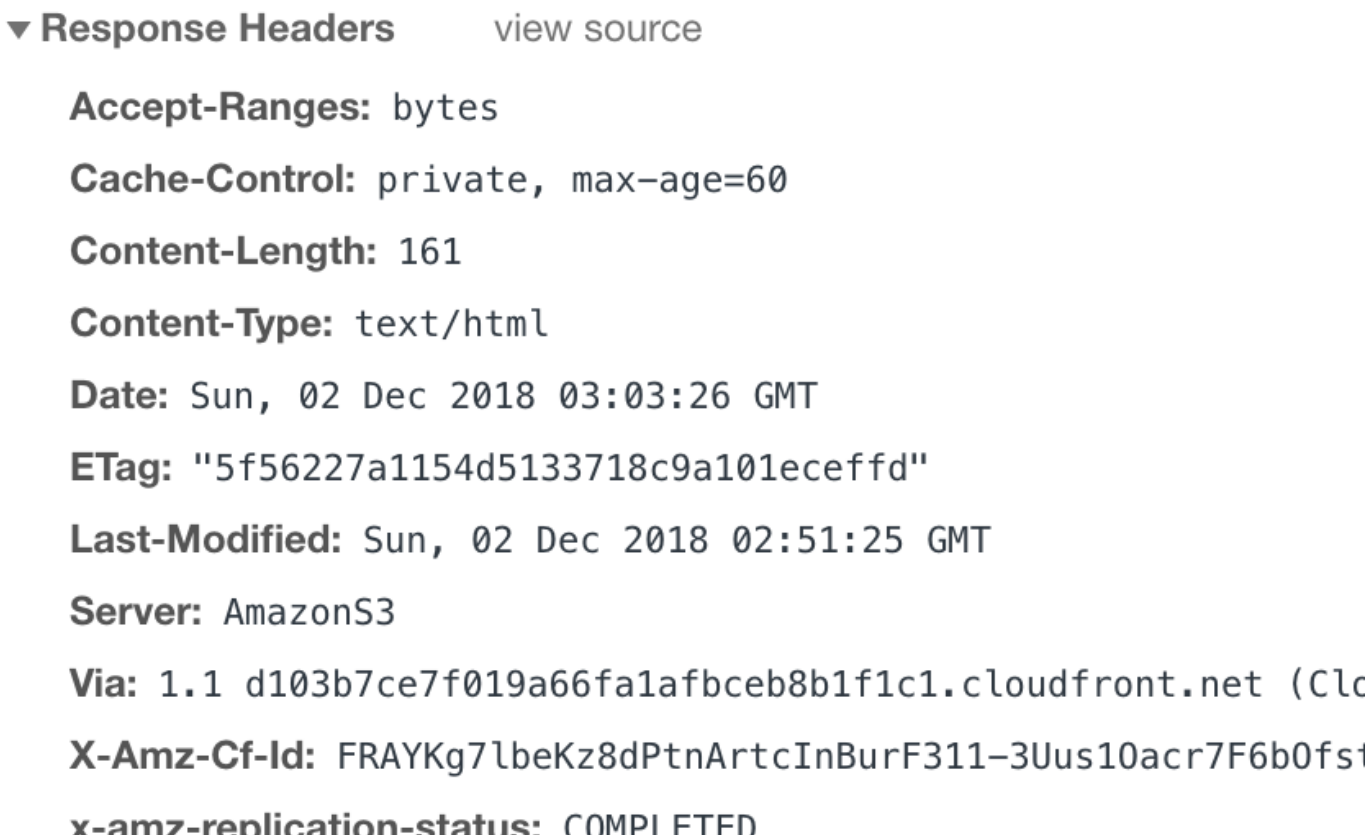




S3 Metadata Properties for Cache-Control

This means that the resource can't be cached anymore by CF and can be cached by Browser for 1 minute.

Navigating the link gets this response. Notice the Cache-Control header values as specified in the S3 metadata. Repeated request to this page will result in `"Miss from CloudFront"`



```
x-amz-version-id: vNIz.zYsuWsAVfl8g3Qs5FMzEH04Tk2i
X-Cache: Miss from cloudfront
```

However, when the page is navigated again within the 60 seconds window, it is served from the browser disk cache (as allowed by private directive)

```
▼ General
   Request URL: http://d1nf5el53kycnx.cloudfront.net/cache_private.html
   Request Method: GET
   Status Code: 200 OK (from disk cache)
   Remote Address: 52.84.225.228:80
   Referrer Policy: no-referrer-when-downgrade
```

Subsequent refresh of the page would have the Browser check with CF, which returns `304 Not Modified` based on the fact that the new version of the page was not added to S3.

```
▼ General
   Request URL: http://d1nf5el53kycnx.cloudfront.net/cache_private.html
   Request Method: GET
   Status Code: 304 Not Modified
   Remote Address: 52.84.225.228:80
   Referrer Policy: no-referrer-when-downgrade

▼ Response Headers      view source
   Cache-Control: private, max-age=60
   Connection: keep-alive
   Date: Sun, 02 Dec 2018 03:07:46 GMT
   ETag: "5f56227a1154d5133718c9a101eceffd"
   Last-Modified: Sun, 02 Dec 2018 02:51:25 GMT
   Server: AmazonS3
   Via: 1.1 3a6d09c229b46334ae8150e9562036de.cloudfront.net (CloudFront)
   X-Amz-Cf-Id: n30howCt5JWMb87sr787efPSbSJ-tvldYXeMz3K3G3qtJ6Lj-3rEiA==
   x-amz-replication-status: COMPLETED
   x-amz-version-id: vNIz.zYsuWsAVfl8g3Qs5FMzEH04Tk2i
```

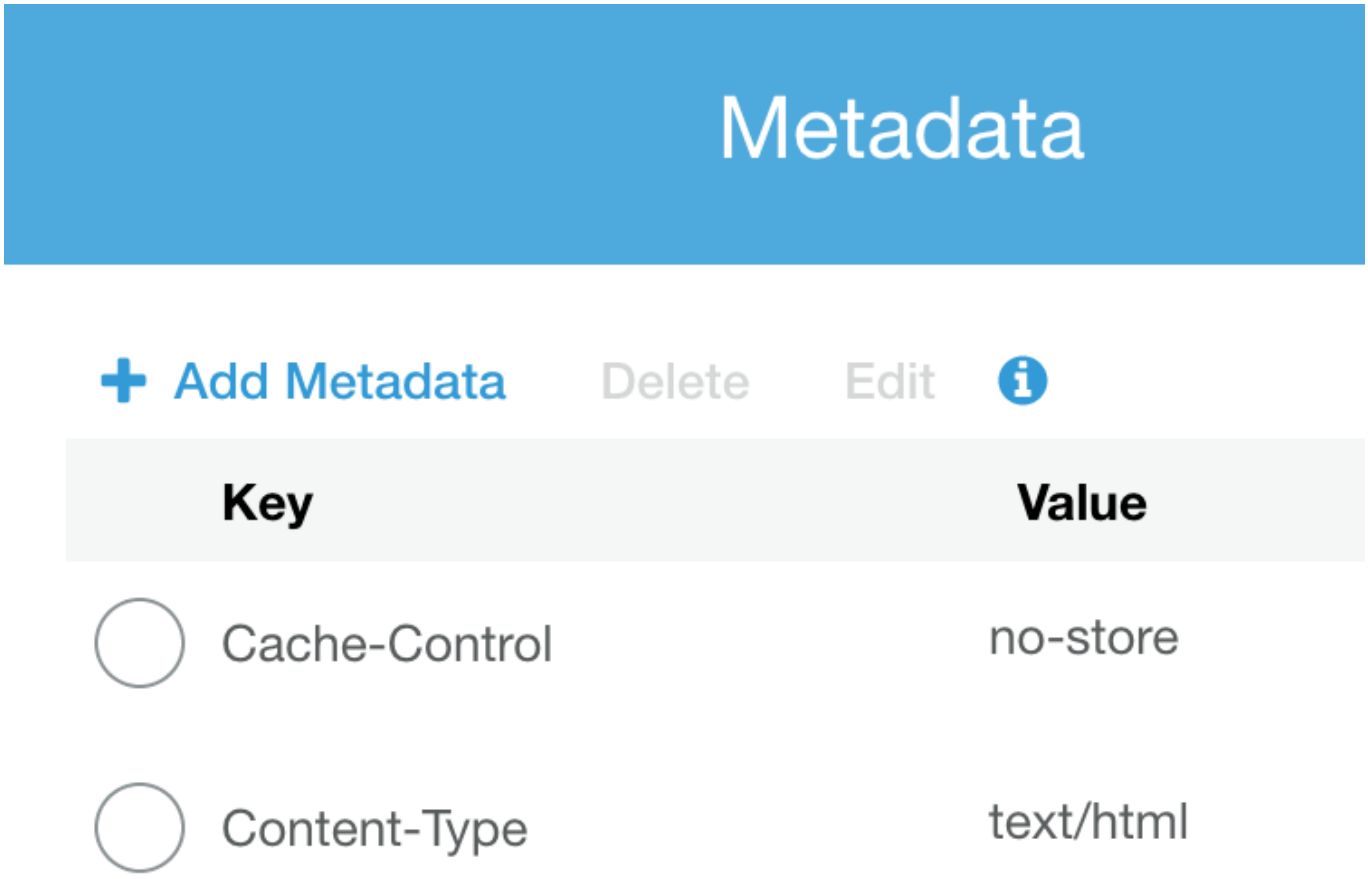
```
X-Cache: Miss from cloudfront
```

If you were to add a new version of the Page, **CF would return newer version** since it never cached the Page.

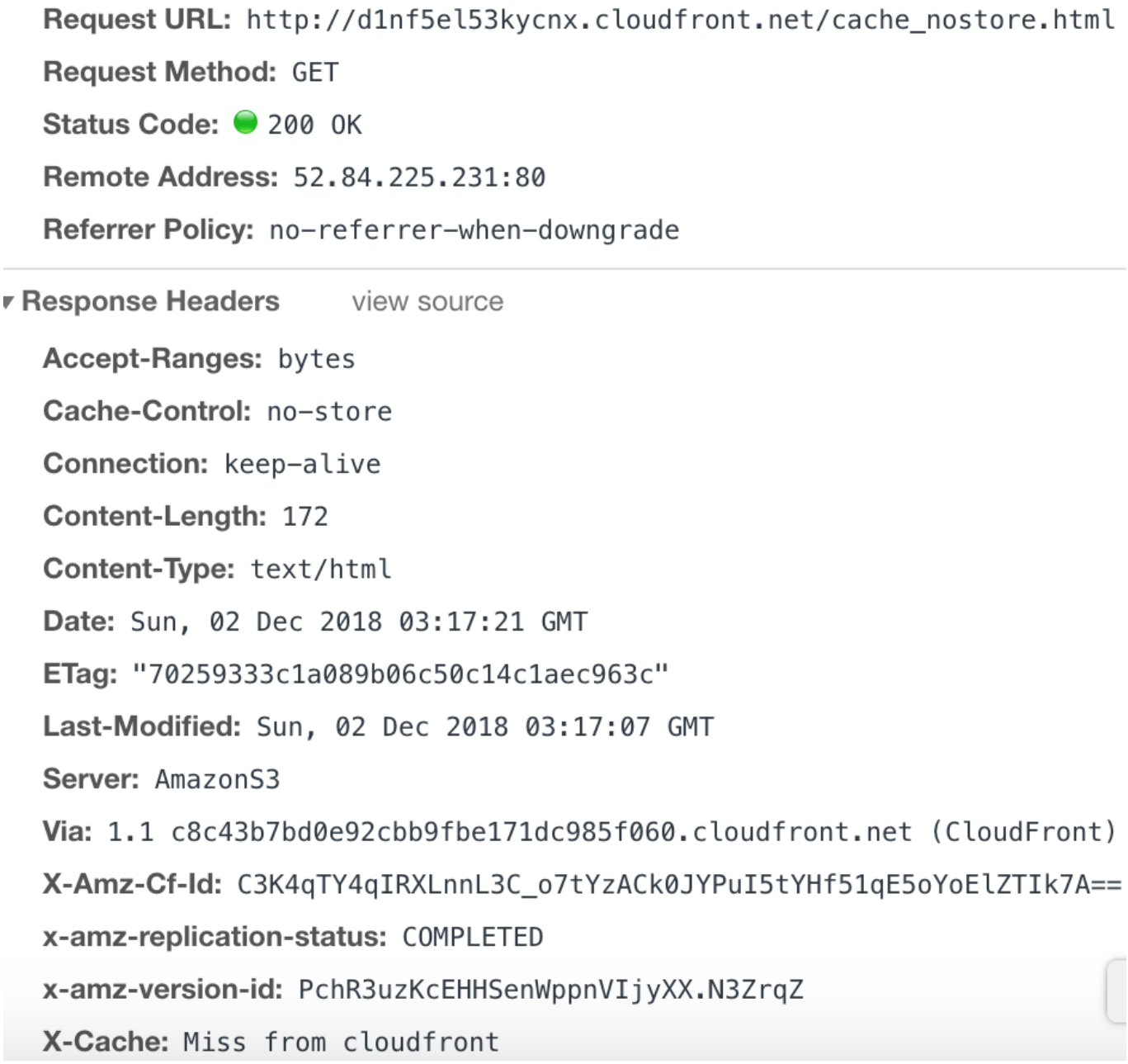
No-Store

The “no-store” response directive indicates that a cache MUST NOT store any part of either the immediate request or response. This directive applies to both private and shared caches.

No-Store: For this exercise, we upload another page to same S3 bucket and configure following metadata. Now the Browser shouldn’t be storing the response either.



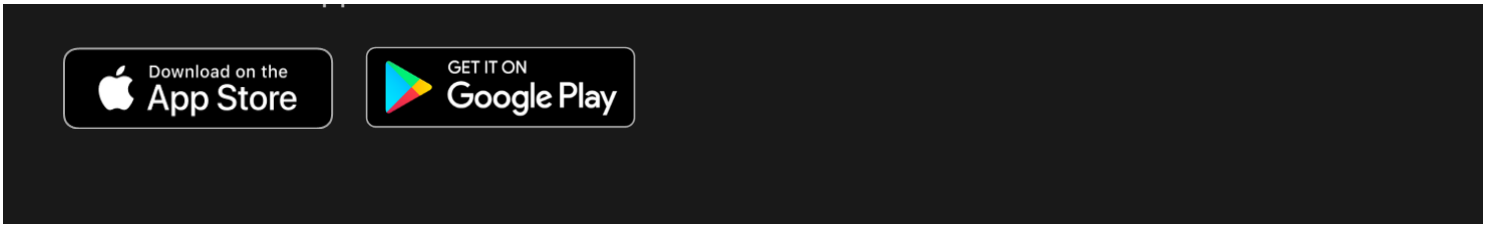
Any number of requests or navigation to the page would always result in 200 OK request/response since no cache is allowed to store it



Hope this article has given some insights about how you can tune your API/site to control caching behavior at the CDN and Browser level.

Thanks for reading !

AWS Cloudfront Browser Cache Http Request No Store


Download on the
App Store
GET IT ON
Google Play