



Improving Your CDN's Cache Hit Ratio

(3 Master Tactics)

Table of Contents

Embracing Cache to Lower PLT	2
Keeping Score of Your Cache Hit Ratio	3
Optimizing Content Delivery Configuration	3
1. Improvement #1: Modify Cache Control Headers	3
2. Improvement #2: Have CDN Strip Cookies	4
3. Improvement #3: Have CDN Ignore Query Strings	6
+ Extra Improvement: Use Shared Hosted Libraries	7
Auditing Your Ratio for Long-Term Success	7
Questions and Follow-Up	7

Embracing Cache to Lower PLT

From second 0, users start abandoning websites, sessions, and shopping carts. And after 7 seconds - a time recognized by Web Performance Today as the performance poverty line - page load time (PLT) ceases to matter because conversions start to plateau.

To keep PLT low and conversions high, enterprises serve digital content from caches that put information closer to the end user. One of the most popular types of caches is the CDN, and the more times object requests hit the CDN cache, the faster pages load.

In this paper we'll show you how to diagnose the three most common cache hit problems and take corrective steps. As you'll see below, these improvements can increase cache hits by up to 50% or more.



Terms and resources that are helpful when talking about CDNs and cache hit ratio.

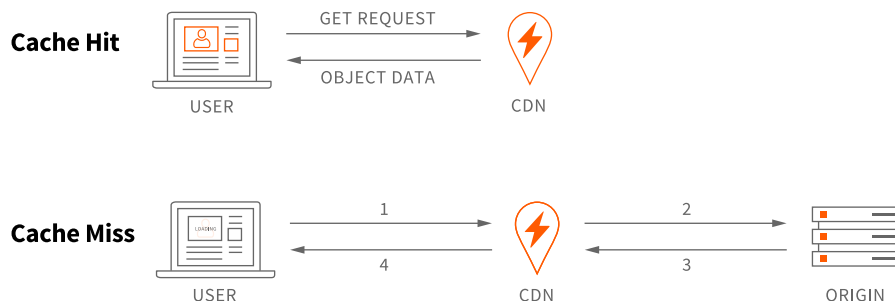
Header Servers and client devices exchange information about requests and objects using headers.

Cold Cache This is the slowest possible object request, usually because of a cache miss.

Warm Cache A warm or hot cache is a cache hit, and results in an object being served from the CDN.

Keeping Score of Your Cache Hit Ratio

When you're keeping score of cache performance, a cache hit is a win and a cache miss is a loss. A cache hit retrieves information from the CDN instead of traveling all the way to the origin server. Every cache miss means slower load times for end users.



With a cache hit, objects are served directly from the CDN. A cache miss requires a much longer round trip to the origin server.

The **cache hit ratio** is the relationship, represented as a percentage value, between cache hits and misses. The higher the ratio, the faster a website's performance.

High-traffic web applications should aim for a cache hit ratio of over 90% for static objects. Achieving such a high ratio requires evaluating the effective lifetime of web objects and configuring your content delivery architecture accordingly.

Optimizing Content Delivery Configuration

1

Improvement 1: Modify Cache Control Headers

The origin server sets a number of HTTP headers on each object it serves, some of which tell browsers and other web clients how long an object is considered fresh. Properly configured cache control headers on the origin inform the CDN of the age and freshness of each object.

If the origin is unable to set cache control headers on all objects, the CDN can set them. It can also override existing ones set by the origin. The CDN's ability to do this can give you the leverage you need to serve content faster from the edge.

Reading HTTP Headers

A number of HTTP headers control caching. These headers can set the maximum age of an object, an expiration time, or even specify that an object is not to be cached at all.

HTTP response/requests headers can be viewed with a number of different open source tools and libraries. Browser-based extensions include [FireFox Firebug](#) and [Chrome Dev Tools](#), while [cURL](#) is the definitive library for command line interfaces across OSX, Unix, Linux and Windows.

i

Common Cache Control Headers

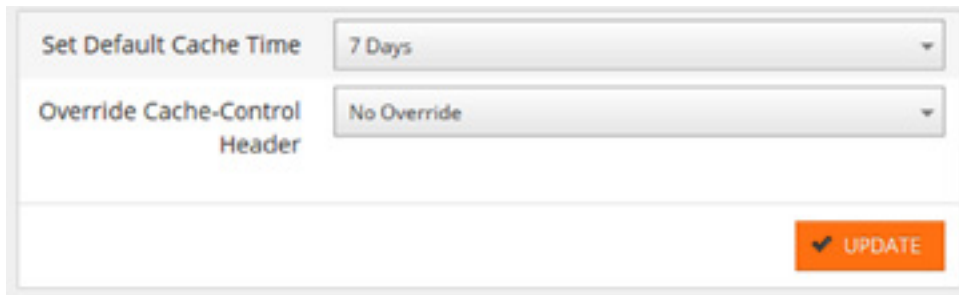
Last-Modified Reports the last time the object at that URL was modified on the origin.

ETag A unique tag for the object used to find it in a cache.

Cache-Control Various rules for caching the object, such as the maximum time between origin requests.

Expires A fixed expiration time for the object in a cache.

A deeper explanation of HTTP headers that affect the cache can be found on the [Mobify Developer Blog](#). You can also read the Internet Engineering Task Force's (IETF) RFC paper regarding [HTTP 1.1 Caching](#).

A screenshot of the MaxCDN control panel showing cache settings. It features two dropdown menus: 'Set Default Cache Time' with '7 Days' selected, and 'Override Cache-Control Header' with 'No Override' selected. An orange 'UPDATE' button with a checkmark icon is located at the bottom right of the settings area.

Cache settings inside MaxCDN control panel that can be changed in one click.

Important: Setting Default Cache Time to one year (max-age: 31556900) is not always the best option

Takeaway: Set a time that best fits your content. For instance, if an asset changes approximately every 2 weeks, a cache time of 7 days may be appropriate. However, if the asset is accessed frequently, you may want to use a Cache Time of one day or implement a version control query string on your back-end. The latter would require you to manage something like a Cache Query String setting on your CDN. (We elaborate on this below.)

2

Improvement 2: Have CDN Strip Cookies

If a static object such as a PDF (like this one) or an image resource is setting cookies on the client side, it's not acting like a static object. Since the CDN can't tell what's going on behind the curtain at the origin server, it needs to assume that objects with cookies are dynamic. Therefore they will be a cache miss every time.

Look for the "Set-Cookie" header in the HTTP Response header to check if your origin is setting cookies. You can do this with one of the open source tools mentioned above.

Notice how the Set-Cookie value changes in the following response headers for the same object. This can force the CDN into revalidating the asset on every request resulting in Cache MISS.



HTTP/1.1 200 OK

Set-Cookie: __cfduid=dbb8af148848b88f6fd38603755f216031425593837

Cache-Control: public, max-age=315360000

Accept-Ranges: bytes

X-Cache: MISS

HTTP/1.1 200 OK

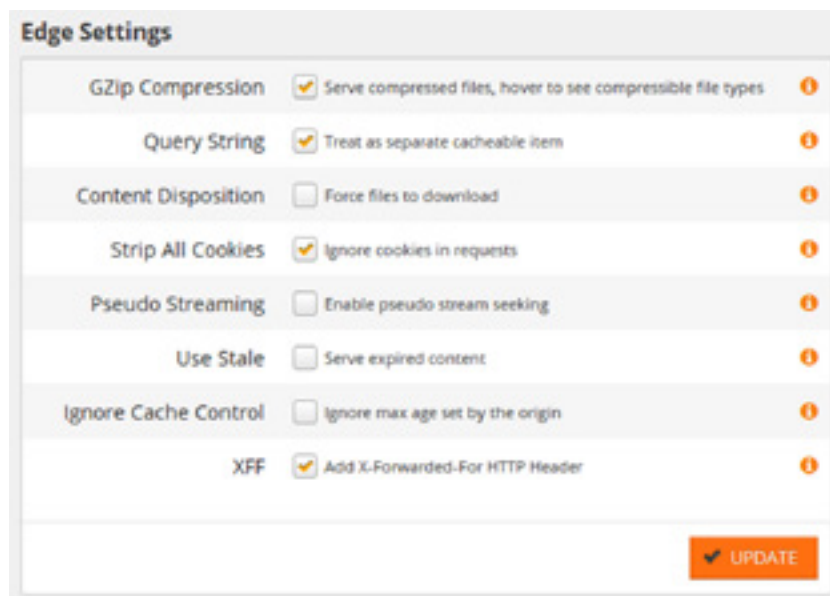
Set-Cookie: __cfduid=ec02ac3ebf9f6ea54ba4e09b1dd197ea51425593838

Cache-Control: public, max-age=315360000

Accept-Ranges: bytes

X-Cache: MISS

If it isn't possible to change how cookies are set on the origin server, some CDNs can ignore cookie data from the origin server's response. This improves the CDN's ability to cache an object.



In addition to managing cookies and query strings, MaxCDN provides a number of options for managing a website's cache.

Takeaway: According to [this MaxCDN tutorial](#) on CDN and cookies, there are scenarios when cookies should be used - for authentication, session id's, etc. By nature though, cookies are un-cacheable, which means files containing them are also un-cacheable. This is why it's important to have cookies ignored in requests for assets you want delivered quickly by your CDN.

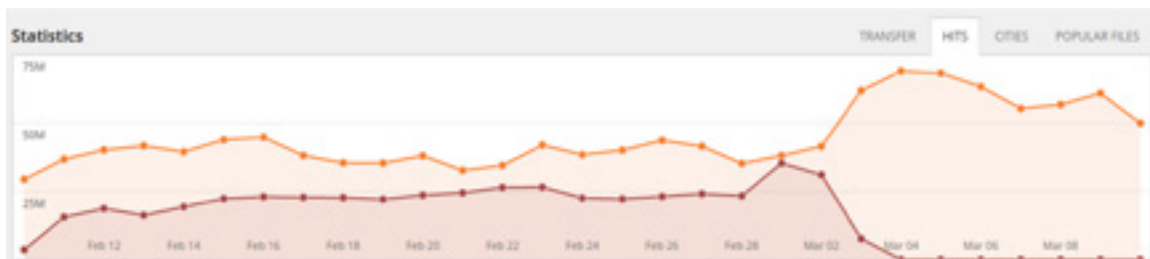
3

Improvement 3: Have CDN Ignore Query Strings

Query strings are used to interact with web applications and APIs, aggregate user metrics, and provide versioning information for objects. When query strings are included in static object URLs, they can be mistaken as unique objects and will be requested from the origin server on each request. This results in an unnecessary decrease in cache hit ratio.

If the origin server is unable to remove query strings from static object URLs, most CDNs should have the capability to ignore the query string, getting a cache hit for all objects with query strings.

Note: Sometimes query strings are warranted and represent unique content. This is why content delivery networks like MaxCDN honor the query string created at the origin as default unless otherwise stated.



Cache Hits (orange) / Misses (red) Vs. Date: MaxCDN's support team found that this high traffic website had query strings on static URLs. After configuring MaxCDN to ignore query strings in the specified zone, the cache hit ratio improved significantly. The fix was as easy as unticking the query string box in the CP (see previous image).



You can check if two static objects change with query strings using the [cURL](#) and [md5sum](#) tools from a Unix or Linux command prompt. If the MD5 checksums are the same, so are the static objects.

```
$ curl -s www.maxcdn.com/images/png/brand/maxcdn-logo.png?ver1212 | md5sum
888310813fffd5aa46f1e51206aea2b5 -
```

```
$ curl -s www.maxcdn.com/images/png/brand/maxcdn-logo.png?ver9191 | md5sum
888310813fffd5aa46f1e51206aea2b5 -
```

In this case, the md5sum hash values (highlighted) are identical. Therefore it's a good indicator that query strings should be ignored. If the hash values were not the same, query strings would represent dynamic content that shouldn't be cached.

Takeaway: When query strings are used to change content on static URLs for each new user, this means they're specific to the user and most likely enhance the user experience. Therefore they shouldn't be cached by the CDN.

On the other hand, you can improve your cache hit ratio by ignoring query strings on static URLs with content that doesn't change based on user. This can be verified with tools like md5sum and file meta-data.



Extra Improvement: Use Shared Hosted Libraries

If your website uses popular frontend libraries like jQuery or Bootstrap, you can use shared hosted libraries to improve your cache hit ratio and save money on CDN bandwidth.

MaxCDN hosts [jsDeliver](#), a service dedicated to shared hosting for JavaScript libraries, fonts, CSS frameworks and more. [Bootstrap CDN](#) is also available, as well as other [free open source CDNs](#).

Auditing Your Ratio for Long-Term Success

If your website's pages border on the performance poverty line, it's imperative to investigate your CDN's cache hit ratio, using this paper as an initial guide.

Even if your average PLT is less than the performance poverty of seven seconds, it's good practice to audit the hits and misses on your cache. Shaving seconds off PLT could be as easy as ticking (or unticking) a box, as seen in the query string case study above.

Audit your hits and misses weekly - if not daily - and always after editing code that affects pages sitewide or in select CDN zones.



Questions and Follow-Up

If you have questions about anything in this paper, contact Robert Gibb at rgibb@maxcdn.com. Based on your questions and our follow-up, we will update this paper accordingly.

Don't already have a MaxCDN account? Contact our sales team today at sales@maxcdn.com. They will set you up with a free test account and get you started on the right foot.

About MaxCDN

MaxCDN is a next-generation content delivery network based in Los Angeles.

Our entire team is obsessed with speed, automation, real time reports and implementation. At MaxCDN, the mission is simple: Provide the best possible CDN experience for today's DevOps teams and the users they serve.



MaxResponse

Ticket response times under two minutes and 24/7 access to CDN support engineers.



MaxControl

View CDN stats in real time, then provision changes instantly through the API or control panel.



MaxArchitecture

Anycast technology routes traffic to the fastest server for your users.



No Request Charges



No Extra Charge for HTTPS Traffic



Rollover Bandwidth



No Contracts Necessary



Great Prices Worldwide

Have any Questions?

[Chat Now](#)

Or speak to someone by calling 1-877-629-2361