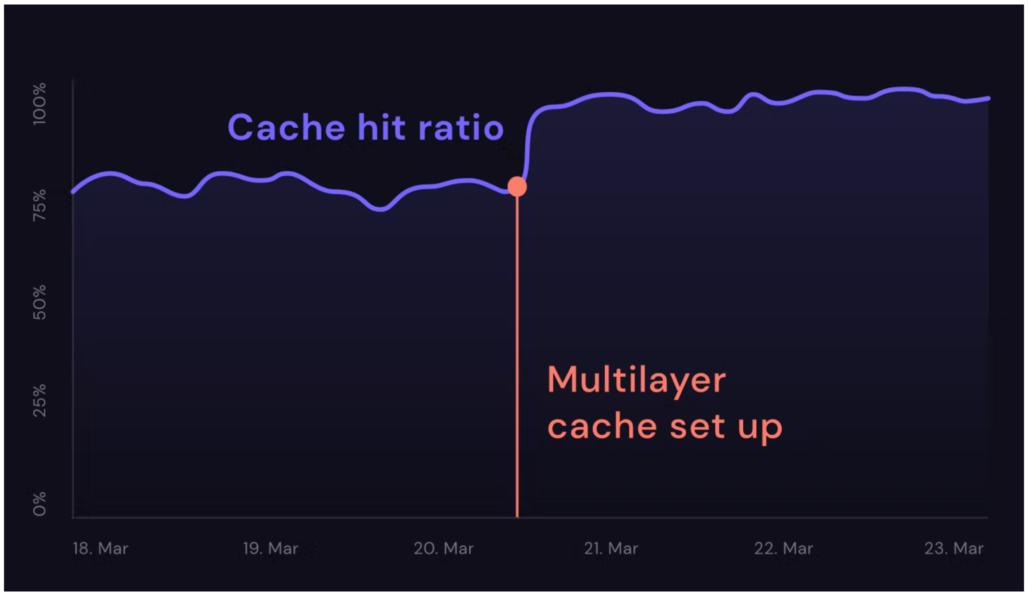
CDN77

← Blog overview

24 March 2021 · 5 min read

CDN for long-tail content? Fight the cache miss with multilayer caching!

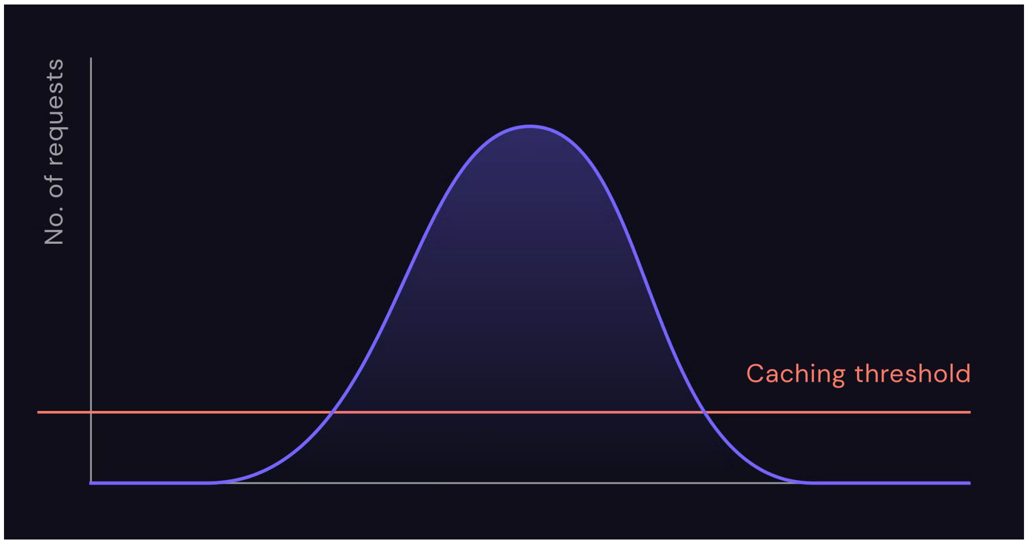


Having the majority of content served from the cache comes with clear benefits. It will improve the performance, offload origin servers, and cut the related egress costs. In CDN77, we've successfully aimed to keep the average cache hit ratio above 97% for our clients. Yet, in some cases, reaching these figures can be quite challenging.

When every piece of your content is only requested a few times, but the sum of those requests amounts to considerable traffic, the traditional CDN approach may not be sufficient. Such content is frequently referred to as "long-tail". But is it still possible to benefit from CDN? Well, yes, it is, and the answer lies in a multilayer cache.

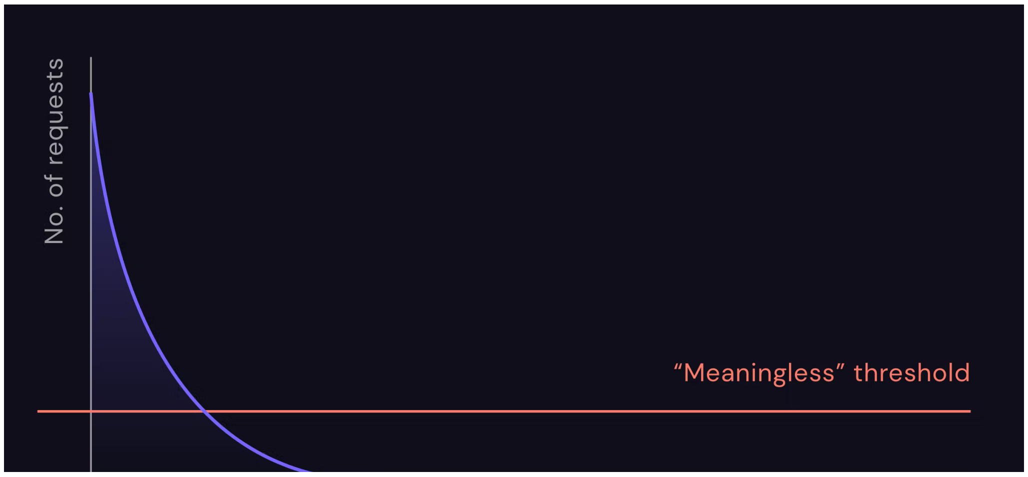
Ideal content structure

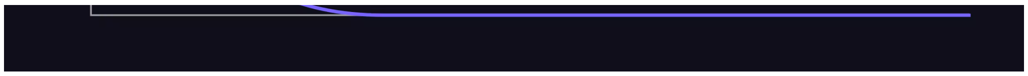
In an ideal case, demand for your data follows the normal distribution, and you can easily cache the most wanted content. As users lose interest, it falls down on the curve to the point where caching it would no longer make sense. Simply setting a reasonable threshold works wonders for the cache hit ratio.



Long-tail content

For long-tail cases, setting a single threshold becomes meaningless. Most of your content would never make it into the cache, and if it did, it would almost never be served from cache again. It would be soon replaced by content requested after it, pushing the hit ratio very low.





An example could be an educational platform with a wide range of subjects. These sites have enormous traffic, but there is no top content for the majority of users. Instead, people choose whatever they currently need from an extensive archive.

The absence of the most wanted content is frequent but not the only scenario that creates long-tail. Some VOD platforms must also deal with a long-tail, even if they can identify the most played content. The problem emerges from the nature of adaptive bitrate streaming. URLs for various qualities differ, so the content would have to be requested enough times to pass a caching threshold for each quality separately. Take HLS and DASH, each with 4 adaptive profiles, and you end up with 8 different copies of each piece of content. People can, and often do, also request only parts of the video. This fragmentation can negatively affect the performance of your CDN.

The solution to this challenging problem is, in fact, relatively simple. We can set up multiple layers of cache that, eventually, converges to a majority of the content stored on the origin.

Multilayer cache

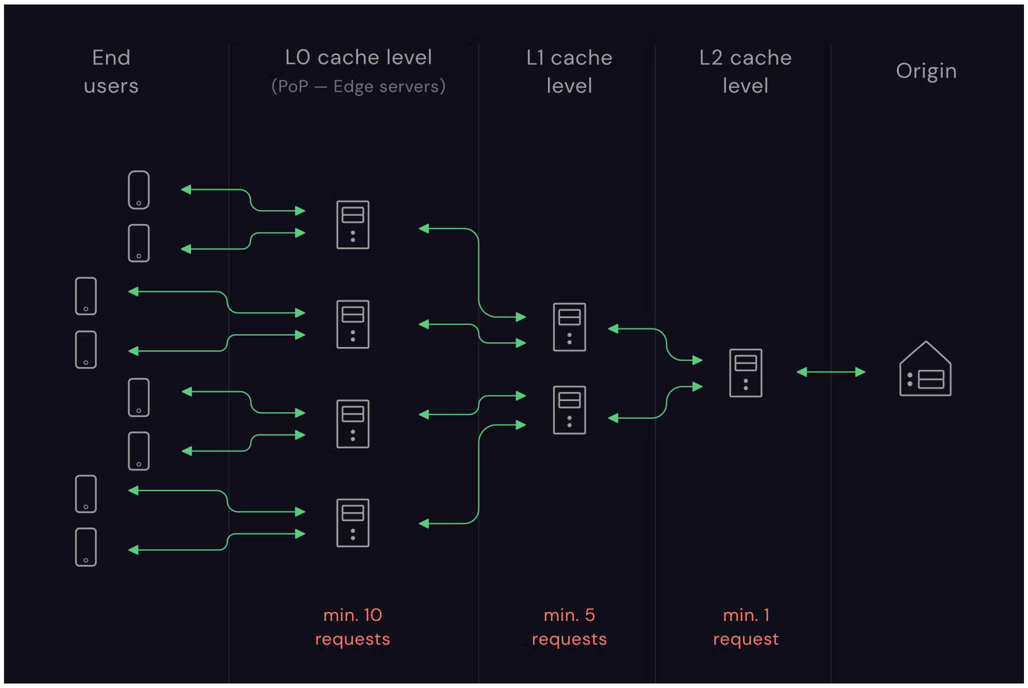
Each layer of the cache works with a different threshold.

The first layer (L0) consists of a multitude of edge servers located in our PoPs. It’s closest to the end-user and stores the most wanted content. Let’s say that you cache your file there when it’s requested at least 10 times.

The second layer (L1) is distributed along some of our bigger data centers, close to the respective PoP and to the user. It also consists of a number of servers and will get requests for content from L0 fairly often. We can imagine a 5 request caching threshold.

And finally, the third layer (L2). Servers on this layer store the actual long-tail content. That means you'll want to cache anything with at least one request. It's located closest to the origin and will serve most of the content instead of it. If the file is not in the cache, only then will this layer send the request to the origin, pushing egress costs to a minimum.

There is no "one size fits all" approach. A solution that works in one region doesn't necessarily have to perform so well in another. For example, you may need L2 in 6 different locations on one continent and only 2 on another. You may even end up not needing one of the layers somewhere. We optimize the topology of all layers to maximize performance in all geolocations.



The hit ratio for such a setup is cumulative. You can expect a pretty low ratio on edge servers – that is the reason we go through all this trouble, after all. Each layer will, however, perform progressively better than the previous one. L0+L1 together will take care of the most wanted content in the area, while L2 holds mostly the long-tail.

You may assume that to keep the hit ratio high, L2 should hold almost all of the data from origin. That's not actually necessary. When a URL is requested enough times to warrant caching the file closer to users, it will result in a fewer number of requests (zero in fact) at subsequent layers. After some time, it will be replaced by more recent, less requested content. This allows you to distribute files evenly across the layers and use available space much more efficiently without duplicates in a cache.

In the event of a failure at any layer, our optimized failover mechanism will forward traffic to a backup, which is chosen with regards to latency and cache hit probability.

> If you’re having trouble with the hit ratio for your long-tail content, a multilayer cache is your go-to solution. It’s fully customizable and you can try it out yourself. Our awesome **Client Solutions team** will be happy to provide more information and offer you a tailored trial. We provide a multilayer cache as a part of our custom solutions along with additional benefits such as dedicated Infrastructure, scalability with the size of origin, and **more**.

Martin Knakal
Developer

LATEST ARTICLES

31 December 2021

Last year in numbers