Caching Emulation for the Edgecast CDN

Marcel Flores
Verizon Digital Media Services
marcel@vdms.io

Harkeerat Bedi
Verizon Digital Media Services
bedi@vdms.io

ABSTRACT

One of the core tasks in operating a Content Delivery Network is caching requested content for content providers. In order to ensure that our caching systems are well optimized, we are constantly exploring alternative caching techniques that involve a rich variety of optimizations and adjustments to our caching policies and structures. In order to rapidly develop and test such improvements, we have developed a caching *emulator*, the Edgecast Caching Emulator (ECE), to allow us to freely experiment with various techniques. Rather than simulating user behavior, the emulator allows us to re-play access logs of real client behavior through alternative techniques and examine caching system performance with alternative caching policies. The ECE further keeps track of important metrics such as hit rate, disk writes, disk reads, original traffic, among many others.

1 INTRODUCTION

Content Delivery Networks (CDNs) have enabled a significant increase in the scale of Internet content delivery. In particular, they deliver large volumes of traffic to globally distributed groups of end users on behalf of content providers. In order to do so efficiently, these CDNs employ large and complex caching systems which allow content to be stored and delivered from geographically distributed Points of Presence (PoPs).

The Edgecast CDN is a global, multi-tenant, commercial CDN [1]. Edgecast features over 125 Points of Presence (PoPs) distributed globally, with over 3,000 interconnects, providing a total global egress capacity of over 49 Tbps. Edgecast provides CDN services for thousands of customers whose content includes live video streaming, static website contents, large software updates, dynamic site content, among many others. This diversity of customer profiles, and consequently client access behaviors, means that solutions which may work well for some customers may induce pathological behavior in others.

In order to address these complex interactions, while still making improvements to the performance fo the Edgecast caching systems, we implement a mechanism for exploring the impact of new caching algorithms and techniques. In particular, this mechanism consumes real data and allows us to evaluate the impacts of such changes on our core performance metrics, before we implement and deploy such systems in production.

2 THE CACHING EMULATOR

The Edgecast Caching Emulator (ECE) is designed to provide insight into the behavior of arbitrary caching policies that we may want to implement in the production CDN. The core idea of the system is that it provides important cache metrics for a given set of access behaviors – which will be given by real behavior observed from CDN access logs.

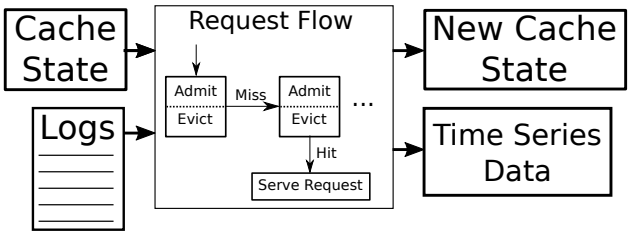

Figure 1: Data flow diagram of the emulator. Each client request passes through a chain of arbitrarily configured caches. Output includes periodically computed metrics.

Figure 1 provides an overview of the emulator. The system takes two inputs: (1) the set of files currently in the cache, called the *cache state*, (2) a set of access logs taken from the production CDN. Each request is then "played back" over the cache policy being simulated, where the object is either found in the cache (a cache hit), or must be retrieved from origin (a cache miss). The output of the system is then a new cache state (*i.e.* the set of files in the cache, which will depend on the cache policy being examined as well as the input logs), and a set of time series data. This time series data includes information about the hit rates (both in terms of number of requests and bytes), the amount of bytes that where read and written from the cache, the amount fetched from origin, and other, extensible properties.

The ECE allows significant extensibility in the cache policies it can evaluate. In particular, it implements the cache as a chain of sub-caches. Each sub-cache may be of arbitrary size, implements an arbitrary admission and eviction policy, and has differing costs associated with read and write operations. While current implementations access the sub-caches linearly (*e.g.* they try the first cache, if a miss, try the second, etc.), non-linear paths may be implemented.

By taking log data as its primary input, the system is able to assess the impacts on *real* customer traffic as seen directly by production servers. Doing so avoids many of the complexities and errors experienced by attempting to model customer traffic. Analysis of the time series output can further be subdivided by customer, allowing for explorations of the impacts on a per customer basis.

In this demonstration, we provide an overview of the ECE, including its system design and implementation. We further provide an overview of some of the most interesting recent findings, which have demonstrated the ability to serve nearly 50% of requests out of small up-front caches that sit in front of our traditional load balancers.

REFERENCES

[1] [n. d.]. Edgecast CDN. https://www.verizondigitalmedia.com/platform/edgecast-cdn/. ([n. d.]).