SWM49 Distributed Systems Principles

Post-course assignment 2010, Part I:

‘Solving Latency and Bandwidth Scalability Issues with Content Delivery Networks’

Mike Pountney, Student ID: 09843038

Table of Contents

Introduction

User expectations for web content delivery have increased to the point where a conventional single-site web architecture cannot reasonably meet the bandwidth and latency demands of the modern - global - web user. The latencies introduced in hosting a content site in a single location, yet serving a global marketplace, result in delays to page and asset download - and are unavoidable due to the hard physical constraint of the speed of light. Also, single site architectures have limited bandwidth 'vertical' scalability - adding more bandwidth at a single site can increase costs massively.

Content Distribution/Delivery Networks (CDNs) are a common method used to address these issues in web content delivery: by ensuring that key content is stored as closely to the user as possible (in both in network hops and physical distance), network latency is minimised. Similarly, by locating the content closer to the user, bandwidth usage is distributed rather than centralised to the originating web server - and hence is more scalable.

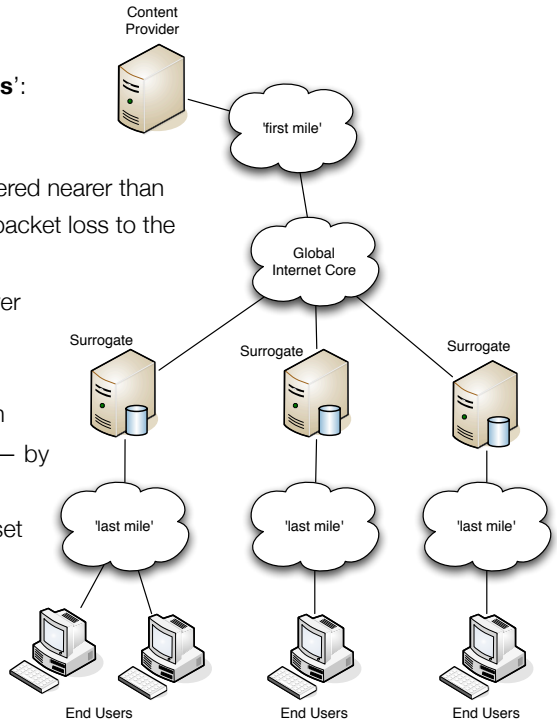
This essay presents the architectures, providers, and implementation details of modern commercial distributed CDNs, by assessing currently available academic literature and other supporting information. In the interest of limiting focus, only web-based non-streaming assets will be discussed.

Architectures of CDNs

The core architectural principle of a CDN is to ensure that the content is as close to the 'last mile' [Dilley et al, 2002] as possible - meaning as close to the end-user as it can be. This ensures both low latency and – as the 'first mile' and internet back-bone is not used, or is used sparingly – highly scalable bandwidth.

To support this however, care needs to be taken. To quote Akamai's approach [Dilley et al, 2002] to determining suitable '**last mile servers**':

> **Nearest** is a function of network topology and dynamic link characteristics: A server with a lower round-trip time is considered nearer than one with a higher round-trip time. Likewise, a server with low packet loss to the client is nearer than one with high packet loss.
> **Available** is a function of load and network bandwidth: A server carrying too much load or a data center serving near its bandwidth capacity is unavailable to serve more clients.
> **Likely** is a function of which servers carry the content for each customer in a data center: If all servers served all the content — by round- robin DNS, for example — then the servers' disk and memory resources would be consumed by the most popular set of objects.



The interesting term in these definitions is 'function' - Dilley highlights that they are dynamically calculated, and hence shows that the selection of 'edge server' [Dilley et al, 2002] (or 'surrogate server' [Mulerikkal & Khalil, 2007]) should not be rigid. '**Nearest'** ensures that network performance to the last mile is optimal. '**Available'** ensures that the servers that serve the user are operational and performant. '**Likely'** ensures that the server is the best fit for serving the content to the user – as it is impossible for all servers to host all content, there is a distinct possibility that the **nearest** and **most available** server may not have the content to serve.

In order to reliably base decisions on these functions, the following areas must be considered: **physicality** (providing 'Nearest'), **monitoring** (ensuring Availability, Likely, and Nearest), **replication** (addressing 'Likely'), and of course **selection** - picking the correct edge server for the task. Finally, **redirection** is used to direct the end-user to the target edge server.

Physicality

In physical terms, to ensure that the 'nearest' function can be optimised, a CDN provider must create a presence as close to the end user as possible. This is typically done by locating servers at ISP '**edge networks**' [Dilley et al, 2002]. To maintain availability, the provider must ensure that there are more than enough servers to cope with demand at each location, and adequate network bandwidth to serve all requests. To increase the likelihood of content being available, the service provider must ensure adequate storage space at each location.

Considering the number of ISPs and other potential candidate edge locations required to serve the world, this creates the need for a considerable distributed infrastructure to be created. In 2002, Akamai quoted their network to be 12,000 servers on 1000 networks [Dilley et al, 2002], in 2010 this had grown to 61,000 servers in 1000 networks [Nygren et al, 2010].

Monitoring

In order to be aware of the changing landscape within the global internet, the CDN provider must ensure that they adequately monitor their network, and adjust their selection algorithms based on the results. With a network size of 61,000 servers, hardware and software failure is a regular occurrence, and must be expected [Nygren et al, 2010].

They must ensure that content caches are populated with the correct content for customer requirements, ensure that servers are not overloaded, and that their network has not been saturated. Relatedly, they must also ensure that a given client is not adversely affecting another client.

For billing purposes, they must also be able to determine how much bandwidth has been consumed for each client, if the CDN is a commercial operation [Mulerikkal & Khalil, 2007].

In addition to monitoring the heath of their own network, the provider must also be aware of alterations to the routing and traffic patterns of the wider internet: BGP routing issues; DDoS attacks; bandwidth saturation; packet loss; and peering disputes. All these factors can change the 'nearest' decision, requiring a different selection of edge server.

Replication

To satisfy the 'Likely' function, the provider need to ensure that the content that their customer (the content provider) requires distributed is available in the locations that the content is required.

Different approaches can be used here [Mulerikkal & Khalil, 2007]:

- **Co-operative push based**: Content is pushed to the edge servers (or a CDN distribution mechanism) from the origin content server (or another means).
- **Non-cooperative pull based**: The edge servers pull the content from the origin servers as required ('on cache miss') - essentially providing a controlled web-cache operation.
- **Co-operative pull based**: Edge servers co-operate with each other to find copies of the content in the event of a cache-miss - this may be quicker than retrieving from the origin server.

In the pull-based case, whereby content is delivered in a method similar to most transparent web caches used by ISPs, the distinguishing characteristic is that the CDN gives the content provider control (they can remove content from the edge servers if required) and visibility (they can see the hit-rates for their content at the edges) [Dilley et al, 2002].

Selection & Redirection

Based on the monitoring of the CDN network, a 'mapping' algorithm [Dilley et al, 2002] is then used to select the best-fit edge network to provide the content to the end-user. The root of this calculation is typically the source IP of the end-user.

To then redirect the end-user to the target edge server, DNS redirection is typically used [Mulerikkal & Khalil, 2007]. The user will have been provided a static URL, typically from the content provider, that is under the authoritative control of the CDN provider, for example:

http://s3q.gh1.akamai.net/my_cdn_content.js

DNS has no awareness of the URL itself, and simply provides the means to resolve the 's3q.gh1.akamai.net' component of our example to a target IP address of the selected edge server.

The client DNS resolver (local or via their ISP) will contact the DNS root nameservers (and then '.net' nameservers) to establish the nameservers for 'akamai.net' - or discover them in its local cache.

Once these have been established, it will contact them and ask for the nameservers for 'gh1.akamai.net', and finally ask them for the IP address for 's3q.gh1.akamai.net'. It is within this process that the CDN provider (Akamai in this case, but the principle is general) has the power to adjust the DNS 'A' (IP address) records provided to the end-user DNS resolver - based on the source IP address seen from the DNS resolver, applying the mapping algorithm to calculate the most 'likely' fit for the traffic.

Once the IP address has been established, the end-user web-browser will connect using the full URL to the CDN edge, where the URL can be inspected fully - there is no requirement at this point for the IP address to map to a single server - and the request directed to the best fit edge server within that edge network.

Drawbacks in using CDNs

The primary reason against using a commercial CDN is cost. Many services are priced outside of the realms of smaller content providers [Mulerikkal & Khalil, 2007]. This is changing, with the introduction of more competition in to the marketplace from companies such as Amazon with their CloudFront offering.

Vendor lock-in can also be a problem [Broberg et al, 2009], due to the integration costs required to use a CDN for content, it can be hard to then switch providers.

Like any third party service, care need to be taken to ensure that the service is performing well - there is the possibility that the CDN provider favours one customer over another in the event of bandwidth limitations, DDoS attacks or otherwise.

Finally, there is also the added complexity to be considered. Whilst it is in the interests of providers to make their systems easy to use and serve their customers (the content providers), there is naturally more complexity than in a single site solution. Care needs to be taken to understand the changed deployment landscape, ensuring correct caching headers are set, and fully understanding how wide-area cached content will affect the site operation.

Implementation Scenarios for CDNs

Supporting page asset offload

A large component of reducing page download times is ensuring that supporting assets for the site – CSS, Javascript, Flash, & image files – are downloaded swiftly.

Firstly, most browsers have a limit on the number of concurrent connections made to the same server (typically four). CDNs help alleviate this problem by providing more 'servers' to download from, allowing for more assets to be downloaded concurrently. In addition to this, the reduction in latency that a CDN provides ensures content is downloaded quickly.

For example, if a content provider is 250ms away (in round trip time) – which is not uncommon for a site that is on the other side of the world since latency is ultimately limited by the speed of light – this adds at least 0.5s to the page download. If there are many assets to the page and the '4 connections limit' applies, then this could easily add up to 2 seconds or more, no matter how swiftly the content server or edge bandwidth can provide the content.

Large content offload

For large & popular file downloads - such as video files, application download files (for software delivery), PDF documents - CDNs can be implemented to improve scalability (as 1000 edge networks can provide 1000x the bandwidth of one content

provider network) and reduce first-mile bandwidth costs. Care does however need to be taken with calculation of costs and popularity - CDN services may be more expensive than source bandwidth, and if content is infrequently downloaded it may result in the similar levels of bandwidth usage at the source (as the CDN re-caches it).

Care also needs to be taken to ensure that protected content is still protected, as the domain of control will have passed to the CDN rather than the content site itself, but most CDN providers have features to keep content protected - via cookie inspection or one-time URL schemes.

Full front-end site offload

Akamai [Nygren et al, 2010] and other CDN providers can act as the front-end web platform for a given site, and select content for display based on a variety of features, such as cookies, authentication tokens, etc. Some also provide 'application server' capabilities, allowing for dynamic content to be provided at the CDN edge.

References

Reference	
[Mulerikkal & Khalil, 2007]	An Architecture for Distributed Content Delivery Network Mulerikkal, J.P.; Khalil, I.; RMIT Univ, Melbourne, Nov 2007 DOI: 10.1109/ICON.2007.4444113
[Dilley et al, 2002]	Globally Distributed Content Delivery J Dilley, B Maggs, J Parikh, H Prokop, R Sitaraman, B Weihl; Akamai Technologies IEEE Internet Computing, Sept 2002 DOI: 10.1109/MIC.2002.1036038
[Nygren et al, 2010]	The Akamai network: a platform for high-performance internet applications Erik Nygren, Ramesh K. Sitaraman, Jennifer Sun; Akamai Technologies & University of Massachusetts August 2010 SIGOPS Operating Systems Review , Volume 44 Issue 3 DOI: 10.1145/1842733.1842736
[Broberg et al, 2009]	Creating a 'Cloud Storage' Mashup for High Performance, Low Cost Content Delivery James Broberg, Rajkumar Buyya and Zahir Tari Lecture Notes in Computer Science, 2009, Volume 5472/2009 DOI: 10.1007/978-3-642-01247-1_17