ACDN: a Content Delivery Network for Applications

Pradnya Karbhari, Michael Rabinovich, Zhen Xiao, and Fred Douglis
AT&T Labs - Research

Content delivery networks (CDNs) currently deliver mostly static and streaming content. However, proxy caches can improve the delivery of these content types as well. A unique value of CDNs could be in improving access to dynamic content, which cannot be cached by proxies. We refer to such a CDN as an Applications CDN, or ACDN. An ACDN will allow a content provider not to worry about the amount of resources provisioned for its application. Instead, it can deploy the application on a single computer anywhere in the network, and then the ACDN will replicate and migrate the application as needed by the observed demand. This demo shows a functional prototype of an ACDN.

An ACDN has a fundamental difference with a traditional CDN oriented towards static content. A traditional CDN server is willing to satisfy any request for any content from the subscriber Web site, either from its cache or by obtaining the response from the origin server. In contrast, an ACDN server must have the requested application, including executables, underlying data, and the computing environment, to be able to process a request. Deploying an application at the time of the request is impractical; thus the ACDN can distribute requests only among the servers that currently have a replica of the application; at the same time, the applications must be placed on ACDN servers *asynchronously* with requests. Thus, ACDN must provide solutions for the following problems that traditional CDNs do not face:

Application distribution framework: An ACDN needs a mechanism to deploy an application replica dynamically, and to keep the replica consistent. The latter issue is complicated by the fact that an application typically contains multiple components whose versions must be mutually consistent for the application to function properly.

Content placement algorithm: An ACDN must decide which applications to deploy where and when. Content placement is solved trivially in traditional CDNs by cache replacement algorithms.

Request distribution algorithm: in addition to load and proximity factors that traditional CDNs must consider, the request distribution mechanism in an ACDN must be aware of where in the system applications are currently deployed.

An ACDN is different from Edge-Side Includes (ESI) from Akamai,[1] which divides a dynamic page into a static template and dynamic fragments and reconstructs the final page at cache servers. Unlike ACDNs, ESI cannot replicate computation required to generate dynamic fragments. Ejasent[2] has announced a system that appears to have similar goals to ours.

Our implementation of the framework is based on the concept of a *metafile*, inspired by ideas from the area of software distribution such as the technology by Marimba Corp.[3] Conceptually, the metafile consists of two parts: the list of all files comprising the application along with their last-modified dates; and the *initialization script* that the recipient server must run before accepting any requests. For example, the initialization script might create a directory for the application's work files and set necessary environment variables. The initialization script can be an arbitrary shell script.

The metafile is treated as any other static Web page and has its own URL. Using the metafile, replicating an application can be done entirely over standard HTTP. Each server is given a `cgi-bin` script that accepts as a parameter the URL of the metafile. This script obtains the metafile using its URL, obtains all application files, and then executes the initialization script if any. Therefore, one creates an application replica on a server by simply accessing the above cgi-bin script with the URL of the application metafile as the argument.

The metafile also provides an effective solution to the consistency maintenance problem. Whenever some object in the application changes at the primary server, the primary updates the metafile accordingly. Other ACDN servers detect that their cached copies of the metafile are not valid and download the new metafile and then copy all modified objects as prescribed in the metafile. Thus, the metafile reduces the problem of maintaining consistency of applications to maintaining consistency of an individual static page (the metafile), a well-studied problem.

Each ACDN server runs a *local replicator* that keeps track of the demand for local applications and the load of all ACDN servers, and periodically makes autonomous content placement decisions for local applications. There is also a central replicator that facilitates exchange of load reports between servers and periodically computes a request distribution policy. The policy is expressed as percentage of requests originated from a given geographical or Internet region that must be sent to a given ACDN server. This policy is downloaded to a load-balancing DNS server[4] that uses it to choose a server when resolving client DNS queries for the application. For update-heavy applications, one can specify the limit on the number of replicas. In particular, if only one replica is allowed, the application will migrate to be proximate to most of the requests.

[1] http://www.esi.org/

[2] http://www.ejasent.com

[3] A. Van Hoff et al. Method for the distribution of code and data updates. U.S. Patent Number 5,919,247, July 6 1999

[4] A. Biliris et al. CDN brokering. 6th Workshop on Web Caching and Content Distribution. 2001.

ACM SIGMOD '2002 June 4-6, Madison, Wisconsin, USA