Demand Routing in Network Layer for Load Balancing in Content Delivery Networks

A SHRAVANI,[1] M.Tech, Computer Science Engineering E mail: sravaniathome@gmail.com
SYED ABDUL MOEED[2]Asst.Professor, Department of CSE, E mail: abdulmoeedsyed@gmail.com
FASI AHMED PARVEZ[3], Associate professor and HOD, Department of CSE, parvez40509@gmail.com

[#] **BALAJI INSTITUTE OF ENGINEERING AND SCIENCES**, Warangal, Telangana, INDIA

Abstract*: Content Delivery Networks (CDNs) play key responsibility in content distribution over the Internet. As the internet becomes an increasingly popular alternative to traditional communications media, internet streaming will become a significant component of many content providers' communications strategy. Internet streaming, however, poses significant challenges for content providers since it has significant distribution problems. Scalability, quality, reliability, and cost are all issues that have to be addressed in a successful streaming media offering. In traditional approach the load balancing is maintained by using time continuous algorithm derived from fluid flow model. By this model the content may take long time to reach the desired location. In current Internet environment, application servers in a remote site may produce a long latency and bad response time such that users do not tolerate the long waiting before being served. The reason for that may be due to the heavy loading of application servers. In order to keep the prospective customers, the content must be made available to the users as soon as possible. Depending on the network layers and mechanisms involved in the process, generally request routing techniques done by DNS request routing. It enables the delivery of content more intelligently and efficiently by redirecting user requests and load-balancing the application servers in Content Delivery Network (CDN).*

Index Terms- Content Delivery Network (CDN), heavy loading of application servers, streaming media.

1. Introduction

CDNs classically deploy a large number of servers across the Internet. By doing this, CDNs offer their customers (i.e., content providers) large capacity on demand and better end-user experience. A Content Delivery Network (CDN) represents a popular and useful solution to effectively support emerging Web applications by adopting a distributed overlay of servers. By replicating content on several servers, a CDN is capable to partially solve congestion issues due to high client request rates, thus reducing latency while at the same time increasing content availability. Usually, a CDN consists of an original containing new data to be diffused, together with one or more distribution servers, called surrogate servers. Periodically, the surrogate servers are actively updated by the back-end server. Surrogate servers are typically used to store static data, while dynamic is just stored in a small number of back-end servers. In some typical scenarios, there is a server called redirector, which dynamically redirects client requests based on selected policies.

Depending on the network layers and mechanisms involved in the process, generally request routing techniques can be classified in DNS request routing, transport-layer request routing, and application-layer request routing . With a DNS-based approach, a specialized DNS server is able to provide a request-balancing mechanism based on well-defined policies and metrics. For every address resolution request received, the DNS server selects the most

appropriate surrogate server in a cluster of available servers and replies to the client with both the selected IP address and a time-to-live (TTL). The latter allows defining a period of validity for the mapping process. Typical implementations of this approach can provide either a single surrogate address or a record of multiple surrogate addresses, in the last case leaving to the client the choice of the server to contact. This becomes possible by distributing the task of data delivery on multiple centers in order to offload origin servers by delivering data on their behalf.

Recently, many CDN providers started migrating their networks into the cloud as the cloud provides numerous advantages for both CDN users and providers. The cloud helps reducing transmission latency as data is stored closest to the user. Operating costs are also reduced where resources could be rent from the cloud provider on demand. Cost reduction will also affect the users as they will no longer need to install physical storage devices to be part of the CDN, and will only pay for the content usage and content transfer.

However, there are many challenges that face cloud based CDNs, such as load balancing and network latency. Load balancing need to be done in parallel with locality awareness to force users' requests to be routed to the nearest data center. It is also required to balance the load on data centers while minimizing the operation cost and maximizing the overall performance. Latency should be reduced as possible by caching data at non-origin servers nearest to end users who request it the most and it is determined in network layer.

2.Related work

In an intelligent CDN, content is routed intelligently to the users passing through the intermediate nodes (e.g. ISG). This scenario improves the network scalability and reliability. The bandwidth consumption on the network is significantly reduced by pushing the content distribution to the edge of the network. However, how can an end-user enter the CDN and know which node is the best for quality content delivery service according to his location and situation?

2.1 DNS lookup and HTTP redirect:

The solution to the above questions is MediaDNS (MDNS) from SinoCDN. It intelligently assigns the best server node or cache engine to the end-users. MediaDNS performs the tasks in two different modes:

DNS lookup:

When a customer requests a streaming URL such as streaming.sinocdn.com, their media players queries their local Domain Name Server (DNS) to resolve the host name to an IP address. The local DNS performs queries iteratively, ultimately reaching the authoritative DNS for the given domain sinocdn.com. The authoritative DNS replies with one or more IP addresses for the given domain name. Once the local DNS has the IP address, it responds back to the customer's browser, which in turns opens a TCP connection with the given IP address and performs content delivery. The local DNS caches the authoritative DNS response and provides the same address for future requests to the domain until the time-to-live (TTL) parameter of the domain's IP address specified by the authoritative DNS expires.

MediaDNS based on this DNS lookup scenario with additional application layer intelligent to direct the end- users to the node (e.g. ISGs) that can provide the best response and quality delivery/streaming services to them, with load-balancing functionality.

2.1.0 HTTP redirect:

Other than DNS lookup, HTTP redirect is another method for redirecting end users to the closest and best server node. End users make http requests to the MediaDNS in this method. The MediaDNS can then obtain the IP addresses of the end users from the http requests. By

knowing this information, the MediaDNS knows the exact location (IP) of the end user. This helps the MediaDNS to locate the closest server node relative to the end user and thus enabling a more accurate decision. After the MediaDNS made the decision, it returns the URL of the best and closest node to the end user. The end user then joins in the CDN through this server node.

2.2 Operation of MediaDNS

DNS Lookup mode:

A client requests content by specifying the URL. The URL always includes a hostname. The URL is sent to the client local DNS server to resolve the hostname to an IP address. This is what happens next:

1. The client local DNS server sends a DNS request to the authoritative DNS server for the domain portion of the hostname streaming.yourcompany.com (in this case, yourcompany.com).
2. The DNS server at your site examines the hostname. If it is properly configured to work with MediaDNS, it will identify the computer where the MediaDNS is running as the authoritative DNS server for the sub-domain streaming.yourcompany.com.
3. The DNS server returns the IP address of where the MediaDNS is running to the client local DNS server.
4. The client DNS server sends an address resolution request to the MediaDNS. The MediaDNS checks whether the name streaming.yourcompany.com is listed in its configuration file as a virtual host.
5. . If the name is not listed, the MediaDNS returns an error message to the client DNS server.
6. If the hostname is listed, the MediaDNS checks the list of server nodes or ISGs that are qualified to respond to that name. It also eliminates from the list any node or ISG that is currently unavailable.
7. The MediaDNS selects a node or ISG using the criteria of loading, available bandwidth and network latency. It then returns the IP address of the best one to the Local DNS.

2.3 HTTP redirect mode:

HTTP redirect mode in MediaDNS provides the request redirection service and global load-balancing functions for streaming delivery on ISG-based CDN. In HTTP redirect mode, end users request for a meta file by http request. The MediaDNS, acts like a web server, receives the request and returns the streaming media address back to end- user. The steps for directing the end-user to the closest ISG in HTTP redirect mode is described as follows:

1. An end user makes an HTTP request by typing a HTTP UR.
2. The MediaDNS which acts as a web server of video.yourcompany.com receives the above request and thus it can obtain the IP address of the end user.
3. MediaDNS controls the ISGs to measure the round-trip-time (RTT) between them and the end user.
4. The MediaDNS selects an ISG using the criteria of ISG loading, available bandwidth and network latency. This IP address and the corresponding content path of the media is sent to the end user.
5. End user gets the full stream URL for streaming request. It then makes the request to the best ISG.

3.0Quick server node Selection

The selection of a server node or ISG for a client by the MediaDNS is based on the following metrics:

1. Available bandwidth: it ensures that the server node or ISG have enough

bandwidth to provide a good quality stream or service to the end user.

2. Loading: it ensures that the server node or ISG will not be overloaded or down after serving the client.
3. Network latency between server node/ISG and the client local DNS (DNS lookup mode) or end user (HTTP redirect mode): it ensures that the end-user can be served with fast response by a nearby node/ISG.
4. Administrator pre-defined network topology such that users come from a specific region are served by a specific (or a cluster of) server or ISG.

3.1 Enable Global/Local Server Load Balancing

MediaDNS enables the network to perform Global Server Load Balancing (GSLB) and Local Server Load Balancing (LSLB). GSLB and LSLB have emerged as the new way of reducing the risk of losing your customers to the competition. GSLB improves content response time by transparently steering the customer to the nearest server. GSLB improves content availability by automatically redirecting the customer away from failed or heavy loading servers or ISGs. LSLB reduces the chance of service failure by load-balancing local network services and redirect users away from failure nodes. Through these techniques, GSLB and LSLB protect an eBusiness from natural calamities, catastrophes, or power outages and ensures that the customer gets to the nearest functional server node.

3.2 Geography Based Selection

The MediaDNS resolves a domain name into the IP address of a server or ISG. When a client DNS makes an address resolution request for a stream, the MediaDNS returns the IP address of a server node/ISG based on the pre-set selection criteria. Other than comparing the latency and loading information, user requests can be redirected according to the geographical location. Users in a specific region can be configured to be served by a specific (or a cluster of) server node.

Scalability: We would like to have a system where capacity constraints can be resolved simply with additional hardware deployment. This implies that the design cannot have any inherent bottlenecks either on the number of streams that it can carry, or the popularity (i.e. number of end users) for any of the streams.

Quality: The network should deliver quality that is equal to or better than any streaming solution based on centralized client-server architectures. The basic idea behind our approach is to *reflect* a stream across multiple intermediate locations and reassemble it on the edge of the network before transmitting it to end users.

Reliability: Our network should have no single points of failure. Software, machine, and even wide area network failures should be dealt with transparently when possible and should result in graceful degradation of service. (Obviously delivering a stream to an end user inside a failed network is not possible.)

Cost: The network should minimize the cost of streaming delivery for customers. While this is a well understood priority in business environments it is often at odds with the preceding goals. It is only through detailed analysis of the system and careful engineering that all of the mentioned goals can be reconciled.

4.0Packet loss recovery

When an end user requests a live stream from an edge server, that server has to in turn get the stream from the encoder that is producing it. Chaining the subscription requests is described in section III-A, but after a request has been propagated to the stream's entry point and data is flowing to the edge server we need to make sure that the data flows with as little packet loss as possible. One possibility would be to use TCP

as the transport layer between the entry points, set reflectors, and edge reflectors but TCP's recovery mechanism from packet loss has undesirable properties like back off and bandwidth throttling than can harm an end user's experience. We therefore decided to explore recovery schemes on top of a UDP transport layer.

4.1 Adaptive multipath transmission:

Even with retransmits it is possible for a particular link in our transport layer to remain loss, or even worse to be down. In order to address this failure case we took advantage of the fact that our system can easily support multipath transmission. Packets originating from a particular entry point can be replicated and sent across multiple set reflectors to a single edge region. The packets are then recombined at the edge region, duplicates are removed, and the clean, lossless stream is handed to the streaming server that can then send it to end users. While multipath transmission can provide extremely high quality and virtually loss-free transmission of live streams, it has one great disadvantage. It is exceedingly costly, especially for unpopular streams. Imagine a case here a single user requests a live stream from an edge server. That stream has to leave the entry point, arrive at a number of set reflectors equal to the number of paths that we want to use, and then be forwarded to the edge region so it can be recombined before being streamed to the end user. If we assume three transmission paths to ensure lossless delivery, then 6 copies of the stream have to travel through the reflector network for the single copy served to the end user. Unfortunately the economics of such a scheme are unacceptable, especially since the additional paths are needed only rarely.

5.0Conclusion

This cram introduced an approach to load balancing in cloud based content delivery networks. The approach takes advantage of the delay table provided by applying the multiple parallel hysteresis model on each data center in network layer, which allows users' requests to be routed to the nearest data center while guaranteeing limited delays even if the data center is loaded up to its near capacity limit. Also shifting of any extra load from overloaded servers to under loaded ones could be done without affecting performance or users' service level agreements in terms of delay. It would also be interesting to have edge regions choose their parent set reflectors in a completely dynamic fashion and not have to rely on bucketing techniques for load balancing. The multipath transmission system can potentially benefit from modifications that would allow it to pick the best path amongst its choices, rather than the number of paths necessary to provide good quality.

6.0References

[1]. Akamai Technologies, http://www.akamai.com /html/technology/index.html.

[2]. Akamai Technologies, http://www.akamai.com /html/perspectives/index.html.

[3] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. In *Proc. of IEEE INFOCOM 1999*, March 1999.

[4] R. Caceres, F. Douglis, A. Feldmann, G. Glass, and M. Rabinovich. Web proxy caching: The devil is in the details. In *Workshop on Internet Server Performance*, June 1998.

[5] P. Cao, J. Zhang, and K. Beach. Active cache: Caching dynamic contents on the web. In *Proc. of IFIP Int. Conf. on Distributed Systems Platforms and Open Distributed Processing*, Sep. 1998.

[6] S. Chakrabarti, B.E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the Web's link structure. *Computer*, 32(8), 1999.

[7] D. D. Sorte,M. Femminella, A. Parisi, andG. Reali, "Network delivery of live events in a

digital cinema scenario," in *Proc. ONDM*,Mar. 2008, pp.
[8] Akamai, "Akamai," 2011 [Online]. Available: http://www.akamai. com/index.html
[9] Limelight Networks, "Limelight Networks," 2011 [Online]. Available: http://.uk.llnw.com
[10] CDNetworks, "CDNetworks," 2011 [Online]. Available: http:// www.us.cdnetworks.com/index.php
[11] Coral, "The Coral Content Distribution Network," 2004 [Online]. Available: http://www.coralcdn.org
[12] Network Systems Group, "Projects," Princeton University, Princeton, NJ, 2008 [Online]. Available: http://nsg.cs.princeton.edu/projects
[13] A. Barbir, B. Cain, and R. Nair, "Known content network (CN) request- routing mechanisms," IETF, RFC 3568 Internet Draft, Jul. 2003 [Online]. Available: http://tools.ietf.org/html/rfc3568

A SHRAVANI studying M.Tech (COMPUTER SCEINCE AND ENGINEERING) in BALAJI INSTITUTE OF ENGINEERING AND SCIENCES, NARSAMPET. Area of interest is DATAMINING, NETWORKING.

SYED ABDUL MOEED completed M.Tech Computer Science and Engineering from JNTU,Hyderabad. Currently working as Asst. Prof, at Balaji Institute of Engineering & Sciences, Narsampet, Warangal., His research areas include Databases, Programming Languages and Information Security, Cryptography, Network Security.

Fasi Ahmed Parvez working as Associate professor and HOD BALAJI INSTITUTE OF ENGINEERING SCIENCES-NARSAMPET, with 12+ years of Experience. Completed M.Tech from JNTU Hyderabad in2010. SUBJECT INTERESTED are programming languages, database management system and data ware house & data mining.