ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 4, pp: (39-50), Month: October - December 2016, Available at: www.researchpublish.com

End-User Request Redirection Performance in Content Development Network using Software Defined Networking-Based Network

Alex L. Munyole[1], Erick M. Ayienga [2]

[1, 2] School of Computing and Informatics, University of Nairobi, Nairobi, Kenya
[1, 2] University of Nairobi, Nairobi, Kenya, www.uonbi.ac.ke

Abstract: **Content Delivery Networks has created a sharp rise in internet traffic in recent years. New technologies have simplified users access to rich content, however, as the internet continues to grow and more users access content stored in CDNs, issues such as reduced scalability, reliability and availability becomes a challenge to end users. This study investigates end user request redirection performance in content development network using software-defined networking - based network. The results should assist researchers in network management, Internet Service Providers (ISPs) in managing request redirection as a service. An experiment to determine effects of client-side DNS caching on the contemporary DNS-based redirection technique found out that both Firefox and Chrome are prone to client-side DNS caching. When subjected to a passive testing, Firefox and Chrome "timeout." Moreover, users had to refresh their browsers for over a minute to get a response from a CDN server. The findings further indicate that Software Defined Networking improves users' quality of service in request redirection. It took ten seconds for multiple users accessing CDN servers to download content and get redirected. Further, the Domain Name System augmented with Network Address Translation technique, based on SDN approach can reliably redirect user request to an optimal CDN server. This is when if it discovers the DNS contains out of date information. The Ryu controller gathered network statistics and pushed SNAT and DNAT rules. The rules directed TCP packets, hence reliably redirecting users to a better performing CDN server.**

Keywords: **Request redirection, domain name system, software defined networking, content development network, the end user.**

I. INTRODUCTION

The role played by Content Delivery Networks (CDNs) as the primary providers of online content is undeniably clear. Depending on the user's location, CDNs renders content promptly to the requesting user. Online services such as video streaming, web video to the TV, file downloads, social networks, and global web portals among others are defining modern CDNs[1]. This novel and emerging online services requires CDNs to offer scalable, yet prompt response time to users' requests. Though CDNs may use multiple approaches to redirect end user's to an optimal CDN replica server, the most common and widely used method is the Domain Naming System (DNS)[2].

Studies in end user request redirection have investigated end user request redirection performance in CDNs using different approaches. Traditionally, users were redirected to specific content using load balancers; however, a major rise and demand for ultra high-quality media and real-time videos have created a challenge on load balancers in redirecting users request optimally[3]. Load balancers focus in distributing requests from users across the entire CDN. The inability of the load balancers being unaware of bandwidth required by the users also hampers end user's requests being redirected at a slow pace, increasing response time [4]. Other approaches have focused on gathering information on the state of the network and making an appropriate decision based on this information. Major studies in this area have concentrated on

collecting information to assist in making routing decisions than on ways of performing redirections[5] [6]. Currently, DNS is employed in CDN networks because of its directory service ubiquity, transparency and simplicity. However, several studies have demonstrated the weaknesses of DNS in supporting the performance of user requests[7]. Studies have classified DNS shortcomings like lack of dependability, performing resolution at the domain level, dependency on Internet Service Providers (ISPs) and reduction of TTL[8][9][10].

Several researchers have extensively explored DNS as a technique CDN uses to redirect end users request to an optimal CDN server. Ranjan & Knightly[11] proposed an algorithm to optimize the response time of dynamic content on the web while reducing resource use. Despite being a success, the algorithm overlooked user's redirection request on static content which also contributed to a considerable percentage of traffic in CDNs. Further, hostnames for websites hosted by CDNS currently are resolved using DNS[10]. The DNS service offers end users with an IP address that map a particular user's request to an optimal CDN node at a given time. Though this is important in practice, cached DNS information may contribute to users accessing a non-optimal CDN server. Baggio[12] observed that DNS is widely used to route users' access requests to an optimal CDN replica server. However, the approach overlooks DNS performed by end users. This leads to effects such as underperformance and decreased browsing experience by end users. Work done by Elkotob & Andersson [13] showed DNS limitations such as having multiple phases of redirection and not embracing scalability. Such shortcomings cause various stages of DNS redirections to occur because DNS caches miss the Round Trip Time (RTT) to be centralized on DNS servers. Furthermore, the Short-to-Live (TTL) used to respond automatically to changes in the network environment increases DNS server load, affecting redirection performance.

On examining published studies on the end user request redirection performance in content development network, it can be seen that there are several methods CDNs uses to redirect end user's request. However, Domain Naming System is the most preferred technique[14]. While pervasiveness of DNS as a directory service is cited as component making the approach efficient, DNS caching provoked by end users has not yet been considered adequately to determine DNS performance. With Cisco's prediction that by 2019, sixty-two percent (62%) of all global Internet traffic will cross CDNs, these developments have a direct effect on performance as well as users access to services rendered by CDNs[15]. This study is vital to researchers in network management, Internet Service Providers (ISPs) as well as CDNs, especially in managing request redirection as a service. Further, the study contributes to other studies by designing an approach based on SDN technology. The proposed approach will redirect end users requests optimally.

This paper explored end user request redirection performance in content development network (CDN) using software defined networking (SDN)-based network. The specific objectives driving this study were to investigate effects of client-side DNS caching on the contemporary DNS-based redirection technique, to investigate SDN-based technique in enhancing quality of service in user request redirection ,to design a user request redirection method using SDN that can redirect users reliably when it recognizes DNS contains out of date information and lastly, to evaluate the effectiveness of the proposed user request redirection technique in improving end user request redirection performance.

This paper is organized in various sections. The first section is related literature on end user request redirection is reviewed; this is followed by research methodology and experimental setup. Next, the paper presents the results and discussion and finally, the paper concludes with directions for further research.

II. RELATED STUDIES

In the past twenty years, Content Delivery Networks (CDNs) has attracted extensive discussion in practice and research. Traditionally, users were redirected to specific content using load balancers; however, a major rise and demand for ultra high-quality media and real-time videos by end users have created a challenge on load balancers in redirecting users request optimally[16][17].

Baggio[18] designed a distributed redirection scheme for end users to access the closest replica by exploiting locality. Distributed redirection achieved request redirection at the client machine locally. However, the distributed redirection scheme was not evaluated and compared with other approaches to ascertain improved user request performance. An efficient request redirection technique should espouse simplicity and benchmarked with other approaches to allow flexibility in managing user request in simple ways.

Xu et al. [20] designed a routing model where the routing decision was pushed near the client using a customized DNS local server. The model permitted the CDNS request-redirection DNS server to return a listing of all probable servers with locally available DNS servers, and with current load information. The locally available DNS server then acquired hope count, route bandwidth, and network latency about a probable server using local Internet gateway router. Though this routing scheme redirected users to an optimal CDN server, it is evident that overheads resulting from aspects such modification DNS servers at the end user's side constrained its performance. To guarantee optimum performance, a solution designed need to scale as the number of user requests and replica grows. Scaling allows optimum usage of network resources as well as decongesting the network.

Other studies used a modified DNS scheme[21]. The DNS modified scheme pushed to end users to request redirection closer to clients using locally available DNS servers. By this method, the DNS redirection to the DNS server returns a possible list of servers together with their valid load information. The local server then acquired extra information such as hop count towards other servers using the local Internet gateway router, network latency, and route bandwidth. While this approach utilizes locally available DNS servers, overheads linked to extracting information constrains its performance. An efficient approach such as using Software Defined Networking (SDN) relieves servers from this duty. A controller collects network statistics and appropriately directs it to the switch. This allows the switch to push relevant information to the network component requiring such information. This approach eases network traffic and improves end user request performance.

An openCache was modeled [17] to bypass caching in CDNs. OpenCache was an SDN caching platform, still under research. However, the approach is more focused on providing a caching solution for ultra-high quality streaming video.

Akamai designed a DNS-based request router that would direct end user to an optimal available CDN node [23]. The router combined network topology and operator media server information with latency, load, and information provided by hyper cache nodes to direct user requests. The router was successful for improving deployment of operator multi-screen services which was confirmed regarding performance, scalability and improved QoE. The router worked by using enhanced DNS-based mechanism. The router translated host names of end user requests and mapped them to IP addresses of edge caches. The router also handled local and global balancing at all CDN levels. However, the router main strength depended on CDN cache to maximize traffic redirection performance.

Work was done by Wang et al. [24] proposed an algorithm to work around DNS. The algorithm was based on OpenFlow standard where network switches divide traffic over server replicas using packet handling rules was designed using OpenFlow. A separate controller installed these rules. He also introduced wildcard rules. For the controller to perform efficiently, it exploited the switch to aid wildcard rules to improve scalability meant for directing traffic of clients to replica servers. The algorithm introduced also computed algorithms that supported traffic distribution automatically. The algorithm simplified load balancing by maintaining existing connections. Although being a scalable solution, this technique has inherent shortcomings. The technique overloads the control plane of a switch. For example, a client requesting to reach its replica server gives priority to the initial call originating from the client to install forwarding rules before proceeding. Also, the solution overloads the switch with many rules.

Harahap & Wijekoon[25] compared DNS and router based redirection approaches using Service-oriented Router (SoR) architecture. SoR was a router capable of redirecting requests from the clients to a server with minimal costs. While SoR router recorded improved service rate performance at 50% and a higher redirection possibility when sandwiched between two servers, its strength depended on CDN cache to maximize traffic redirection performance.

Architectural Design

The goal was to come up with a prototype of a solution for end user request redirection. The block diagram below represents the architectural design of the solution. The solution is an innovation using existing Software Defined Networking.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 4, pp: (39-50), Month: October - December 2016, Available at: www.researchpublish.com

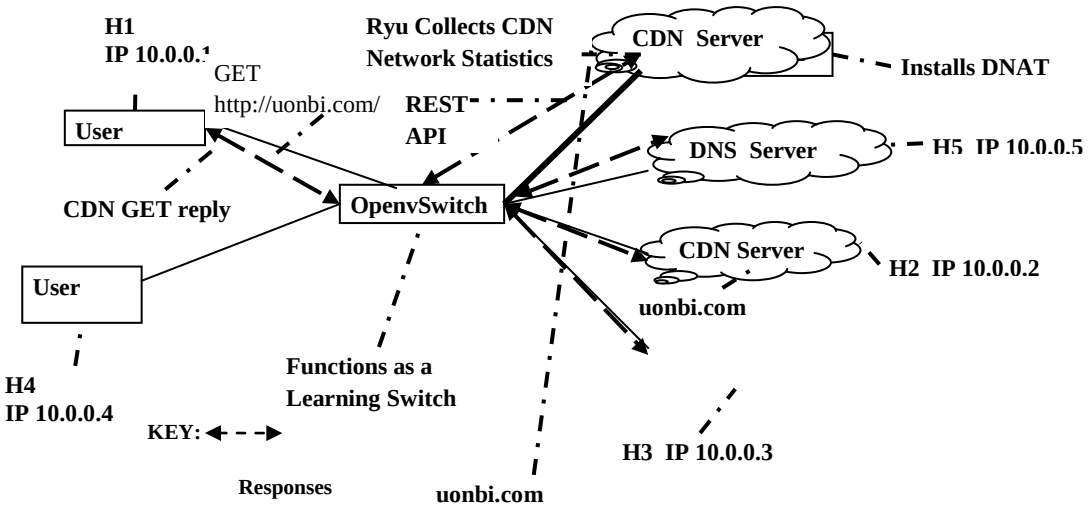

Fig. 2. Architectural Design

III. METHODOLOGY

An end user wishing to access content hosted on a particular CDN, type a URL in a browser. The end user's browser then performs a DNS request, mapping an IP address to the URL typed in the browser. This process results in a request being serviced by the browser's DNS cache, with response returning instantly. The process prohibits external DNS servers from resolving the URL. The end user's browsers then connects to a newly established TCP session; communicating TCP packets with a CDN replica server. The newly established session is identified by source and destination TCP port, and the source and destination IP. When located within a local CDN network, for example, an ISP CDN network, the first packet reaches the edge router. However, if the destination IP address for this session is identical to the originating optimal server, mapped to a particular end user, then no further redirection will be required. Therefore, a route, establishing new flow for this TCP packet to a CDN server is established and the packet is redirected normally (see Appendix VI Wire shark Statistics). On the other hand, if the IP address for the TCP session differs from the IP of an optimal CDN server, end user redirection occurs. A flow is established, performing a DNAT that changes the destination IP packets belonging to this TCP session to a new CDN address. The TCP session then redirects the packets to an optimal CDN server, as a router normally does. A reverse route is the again established, performing SNAT on packets returning from a CDN server.

Source	Destination	Protocol	Info
10.0.0.1	10.0.0.5	DNS	Standardquery 0xc74c A uonbi.com
10.0.0.5	10.0.0.1	DNS	Standard query response 0xc17c A 10.0.0.3
10.0.0.1	10.0.0.3	TCP	53790 > http [SYN] Seq=0
10.0.0.3	10.0.0.1	TCP	http>53790 [SYN, ACK] Seq=0 Ack=1

Fig. 3. A NAT process on a TCP packet flow

In Fig. 3. H1 with an IP address 10.0.0.1 make a request to 10.0.0.5, a DNS server for uonbi.com. A DNS provide response with IP 10.0.0.3. H1 (10.0.0.1) establishes a TCP-SYN to CDN server (IP 10.0.0.3). H1, 10.0.0.1, is assigned a TCP-SYN packet directed to 10.0.0.3 (CDN server) after DNAT has been attached. The packet trend continues until SNAT is attached to all the flow. The switch installs these flows, helping redirect end users based on the TCP session created. The TCP session established allows end users seamlessly connected, without being redirected midway.

Experimental Setup

We used quasi-experimental approach. This section describes the experimental environment used to design, implement and test DNS end user request redirection augmented with NAT.

A. Topology Devices

Topology devices were emulated in a Mininet test bed. Ubuntu 14.04 acted as our operating system. The devices consisted of hosts. These hosts were labeled as H1, H2, H3, H4, H5, S1 and C0. H1 was designated to test our 'improved' CDN end user redirection mechanism. The host was within a CDN network. H1 was connected with 20Mbps to switch 1 to simulate an ADSL2+ high-speed subscriber line. H2 and H3 were our Web servers. They were designated as CDN replica servers. They served similar content. When not overloaded, they had 100Mbps connection to switch 1.

H4 was appointed as a host. It was used to contrast a standard CDN end user request redirection mechanism with H1 ('improved' mechanism). H4 was available outside a CDN network. This scenario was adopted to ensure H4 does not experience the effects of our proposed request redirection mechanism. H4 was connected with a 20Mbps connection to switch 1 to simulate an ADSL2+ high-speed subscriber line. H5 was used as our DNS server. S1 was our OpenvSwitch. We used C0 as our controller.

B. C0 - Ryu Controller

Ryu was selected as our network controller framework for in this study. It had an inbuilt REST API. The controller was used to communicate with the network switch. We implemented a REST API using a shell scripting language to push network changes to the controller. The Network Controller (C0) was connected to the CDN network to collect information. The information the controller collects includes CPU load, link load, and network throughput.

C. S1 - OpenvSwitch

Our OpenvSwitch is an instance of OpenvSwitch running in a Mininet topology. OpenvSwitch is represented by S1 with an IP 10.0.0.6. We inspect flow tables any time using this switch by executing "ovs-ofctl –O OpenFlow13 dump-flows s1" command at the command line (CLI). The switch is operated in OpenFlow1.3. Before the controller is started, the switch's flow table is empty, but when the controller is started, a flow table is inserted in the switch. OpenvSwitch is a learning switch with extra flow rules that allow NAT and DNAT insertion on TCP flows passing through it.

D. H1 and H4 - Users

Our end users are represented using H1 and H4 in our network. They have 10.0.0.1 and 10.0.0.4 IP's respectively. We use H1 as a control to test our 'improved' CDN end user redirection mechanism while H4 is used to contrast the performance of our technique with H1. H1 has an IP address 10.0.0.1 while H4 has 10.0.0.4. H1 and H4 are both installed with Chrome and Mozilla Firefox. The browsers chosen are a representative of the proportion of the contemporary browser share market.

E. H5 - DNS server

The study designed a simple DNS server. The DNS was implemented on H5 using the BIND9. The DNS server was designated as an authoritative nameserver for uonbi.com content in our testing framework. When simulating network change (adding or changing optimal CDN replica server to the (Hostname), A record provided by H5 is updated using a shell script. In our experiment, this server has an IP address of 10.0.0.5 and sets the TTL on its response to be 30 seconds.

F. H2 and H3 – CDN Servers

H2 and H3 represent our CDN server network. H2 and H3 are implemented using apache2 server files and renders access through port 80. They have IP addresses as 10.0.02 and 10.0.0.3 respectively. They both point to uonbi.com to render identical web pages.

E. Links

The links in our network topology had a configuration adding 10ms delay to TCP packets passing through them. The 10ms delay aimed at stimulating the location of all network components closer to the network edge router.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 4, pp: (39-50), Month: October - December 2016, Available at: www.researchpublish.com

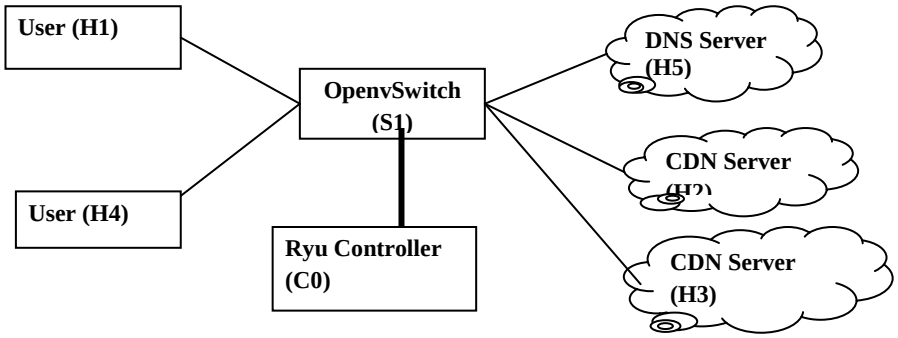

Fig.4. Diagrammatic representation of network topology

Sampling

The target population considered by this study was packets (GET, TCP, DNS, and HTTP) from end users (H1 and H4). The unit adopted for analysis was individual packets. Performance tool was used to generate twenty-five test samples randomly. Since "sniffing" of end user requests begun at H1 and H4, the objectives of collecting test sample at this point were to make it efficient and secure. This means that only valid tests on end user requests send out towards CDN replica servers for starting/accessing uonbi.com contents was collected. Tests on the data gathered were done at intervals. The test samples generated were used as the sampling frame. The packet sampled provided a reasonably representation of users in a CDN network. Random sampling was appropriate owing to the uniqueness of this study.

Data Collection

A systematic observation of GET, TCP, DNS and HTTP packet behaviors during the experiment was recorded at intervals. Since "sniffing" of end user requests begins at H1 and H4, the objectives of collecting packets at this point was to make it efficient and secure. This meant that only valid tests on end user requests send out towards CDN replica servers for starting/accessing uonbi.com contents were collected. Observation on tests regarding data gathered was done at intervals. HTTP GET requests were only collected using a web performance tool available in Mozilla Firefox and Chrome. Observation method was highly favored in this study. It required no demand characteristics or subject reactivity.

Test Procedures

We classified our tests into two parts; passive user and active user; with the ultimate goal of attaining end user performance in a CDN network. The tests were desirable so as to model 'expected' user behavior in times of duress. As such, the study modeled a scenario where users access to content by typing the URL into their browser and another scenario where if congestion was experienced, or load time was greater than 20 seconds, the user refreshes the page.

Request redirection after link failure

In this method, we observed the effectiveness of our proposed request redirection mechanism after a CDN replica server encountered a link failure. The primary concern in this test was content access delay.

Request redirection when the link is congested

In this test, the study observed the success of our proposed request redirection approach when a CDN replica server was under congestion.

Request redirection after introducing a new CDN replica server

This test aimed at providing information on the speed at which a new CDN replica is established depending on server request and how it affects visitors on a CDN network.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 4, pp: (39-50), Month: October - December 2016, Available at: www.researchpublish.com

IV. RESULTS

Request redirection after link failure

The result for request redirection after link failure was determined using Firefox and Chrome. Reference time was used to determine the load of the website located in the CDN replica server without invoking request redirection. When Chrome and Firefox browsers were used passively as a testing approach, they were unable to fully load a page after one to two minutes, the browsers "timeout". However, when active invocation was done, Firefox and Google Chrome showed a significant outcome on the load time.

Firefox recorded a mean of 80.00 seconds difference in the load time compared to Chrome. A similar experiment with Firefox also achieved 120.10 seconds. Chrome average mean load was 30 seconds compared to Firefox. The active behavior was similar. The refresh rate was 30, 35 and 150.05 based on the simulation observation. In this case, the access delay for the page was a minute after a refresh was done after the total of about a minute's delay since the browser visited uonbi.com, the first test page appeared to begin loading. However, despite the starting to load, it did not load making nine images out of ten to load on the academic.html.

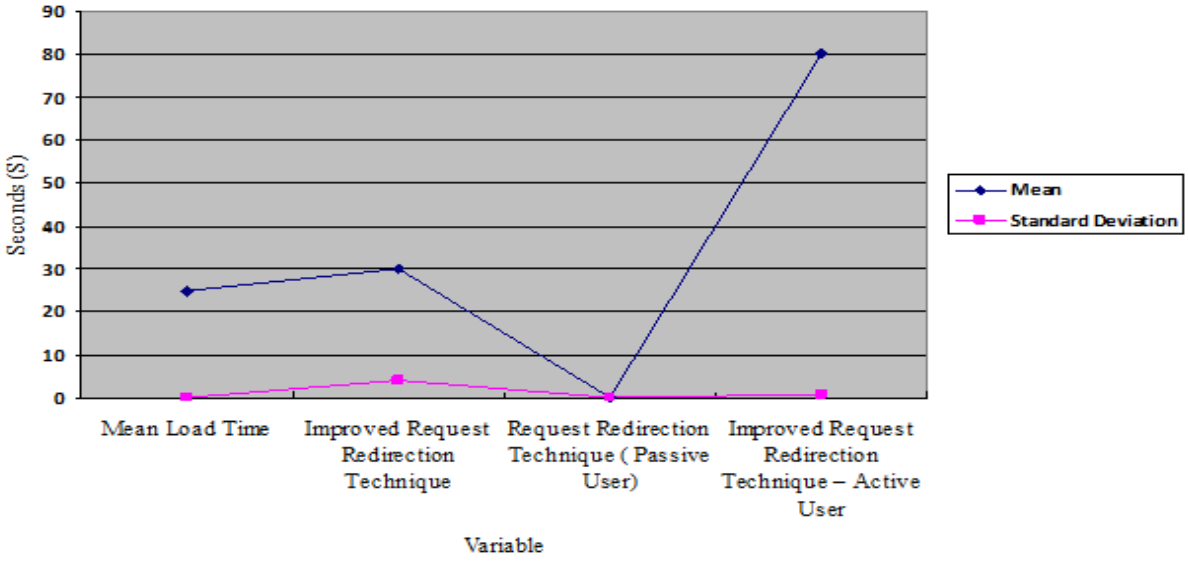

Fig. 5. Time in seconds (s) Firefox browser took to traverse and load uonbi.com website

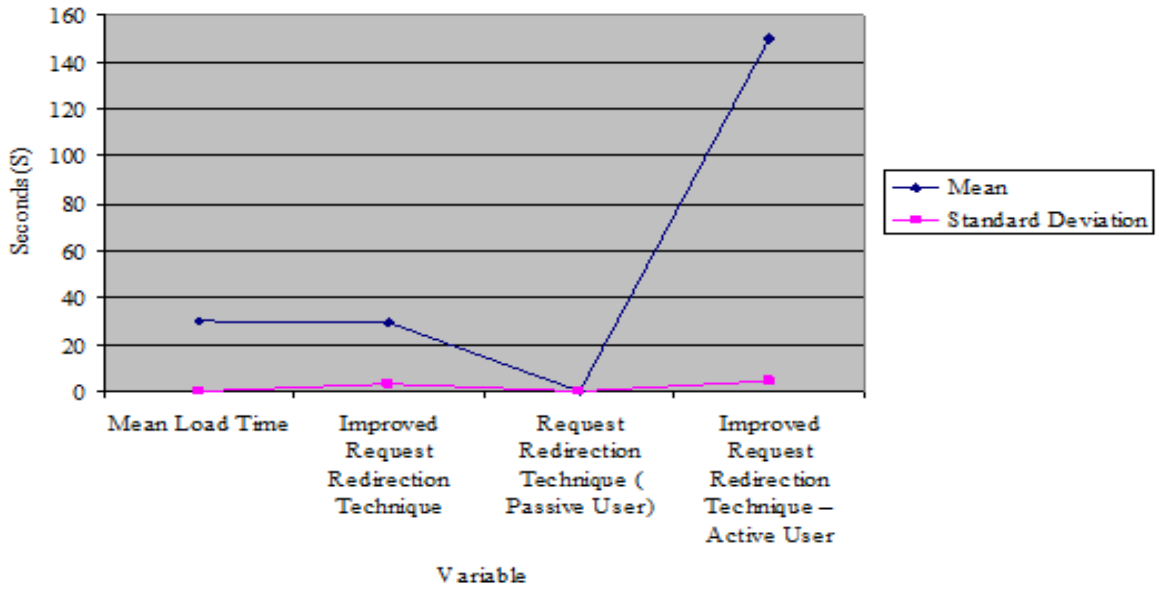

Fig. 6. Time in seconds (s) Chrome took to traverse and load uonbi.com website

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 4, pp: (39-50), Month: October - December 2016, Available at: www.researchpublish.com

Request redirection during link congestion

Request redirection as a result of link congestion was simulated using two methods; when the link was uncongested (*request redirection after link failure*) and when the link was congested. Mean load time when a link was congested was the duration uonbi.com loaded all pages on the congested link with 0 end user request redirection. In a test to assess load time without improved request redirection mechanism, Chrome recorded a mean of 86.51 seconds, and Firefox recorded 84.23 seconds. However, both browsers recorded a delay when connecting to a new CDN server without using improved request redirection. Firefox recorded the highest mean of 71.20 while Chrome had 59.04. However, the time taken to connect to a new CDN server using an improved request redirection technique was low with Firefox recording a mean of 15.71 seconds while Chrome showed. Further, load time using an improved request redirection differed among the browsers. Firefox recorded a mean of 29.05 while Chrome recorded 30.40 seconds.

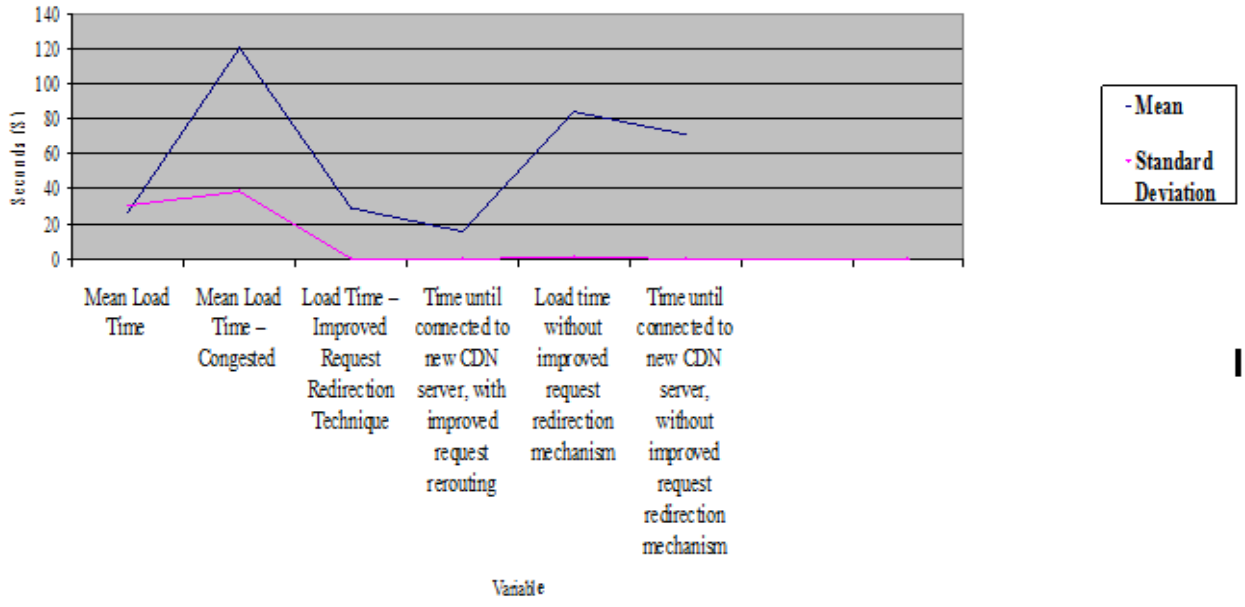

Fig. 7. Time in seconds a user connects to an optimal CDN replica server using Firefox and the time taken to traverse uonbi.com website

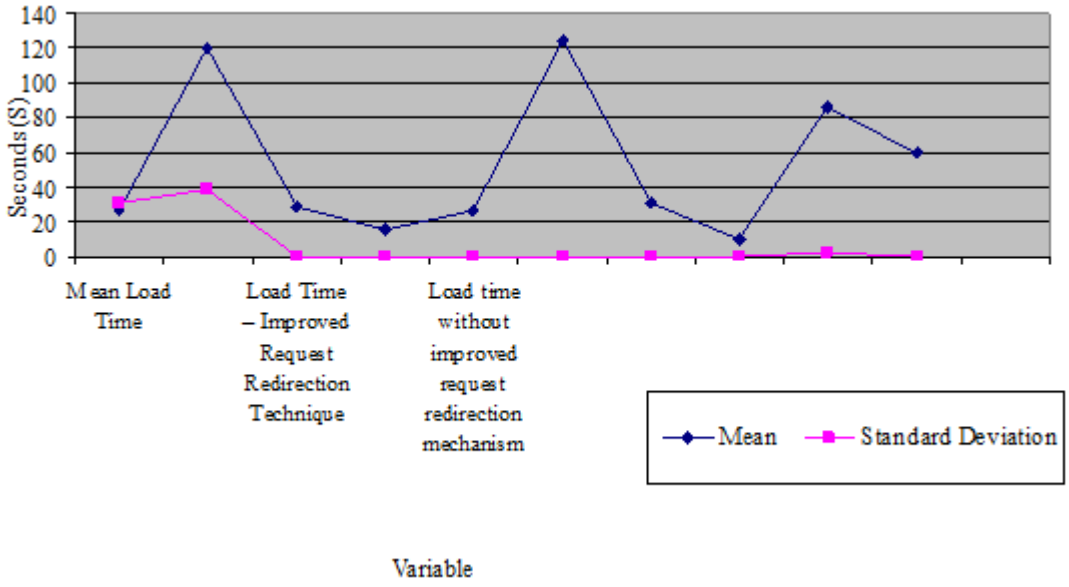

Fig. 8. Time in seconds a user connects to an optimal CDN replica server using Chrome and the time taken to traverse uonbi.com website.

End-User Request Redirection with the introduction of a new CDN server

This test determined the duration H1 and H4 hosts took to complete loading uonbi.com website. This was when the hosts were accessing a congested H2 CDN replica server simultaneously. When the measurement was taken, H1 and H4 appeared to load the uonbi.com randomly. This test measured H2 total time when subjected to the load. Further, even if a single host had completed uonbi.com test pages, it was noted that the time was lower than a half achieved compared to H4. The higher standard deviation values observed in other measurements are as a result of H1. When H1 was flooded with GET request responses emanating from H2 at the 40-second mark, it could take 15 to 20 seconds for H1 to complete the uonbi.com.

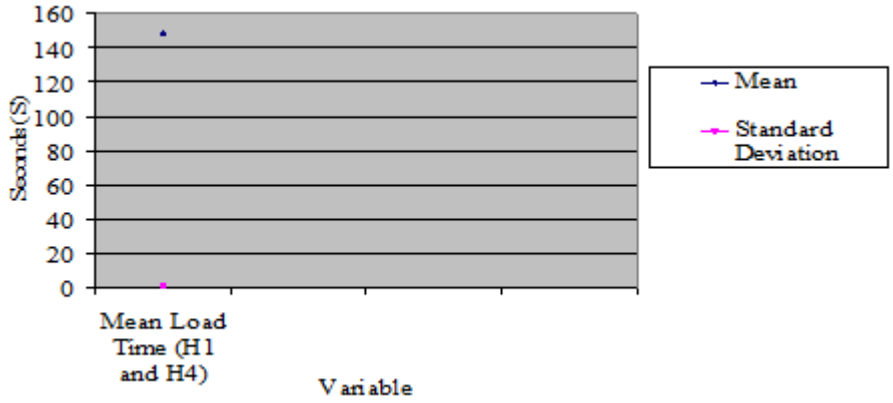

Fig. 9. Duration in seconds H1 and H4 take to load uonbi.com website

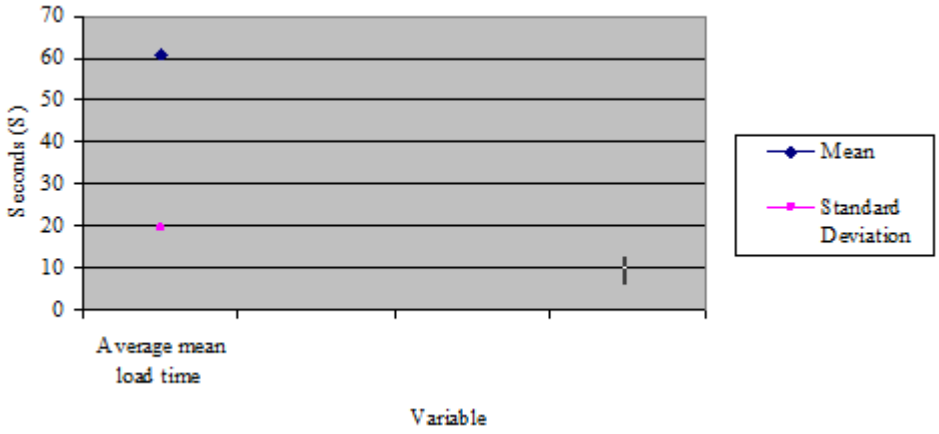

Fig. 10. Duration in seconds H1 takes to load uonbi.com website

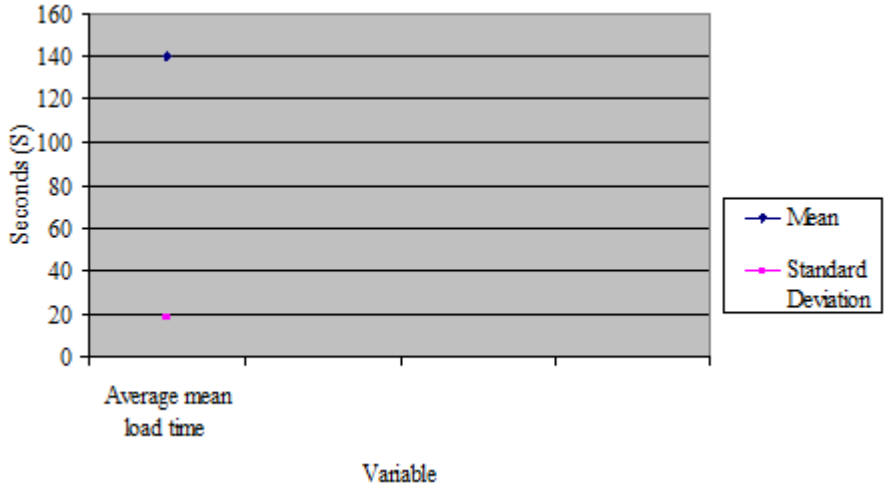

Fig. 11. Duration in seconds H4 takes to load uonbi.com website

V. DISCUSSION

This study investigated end user request redirection performance in content development network using software-defined networking -based network. To our knowledge, it is a novel study that demonstrates the effects associated with client-side DNS caching on the contemporary DNS-based redirection technique.

Summary Of Findings

In a test sample investigating request redirection after link a failure, the results show that Firefox and Chrome fail to load a page when in a passive mode. End users were prompted to refresh the page several times. It is only after continuous refreshes do the page load. This explanation is in tandem with previous studies that have postulated effects connected with client-side on DNS caching on modern DNS-based end user redirection[10]. The results for an "improved request redirection technique" by an active user shows that despite Chrome and Firefox taking more seconds on a uonbi.com website, the page began loading. Such a delay resulted in access delay. Hence an extra time is shown in Fig. 3. As shown Table 2, this extra time indicates a performance cost of 5.05 seconds higher compared to our proposed end user redirection technique compared to the reference.

Software Defined Networking can improve user's request performance in a CDN. This assertion is demonstrated by a close mean load time exhibited by both Firefox and Chrome browsers. When congested, the approach redirects users to uncongested CDN using network information supplied by the controller. This is shown by a higher discrepancy of the reference time (0.2 seconds)**.** The processing of TCP packets by the controller might be the reason causing the 0.2 seconds lag.

The proposed end user request redirection can successfully be tailored to use the introduction of rapidly established CDN replica servers to augment Quality of Experience for users and CDN network by indirectly reducing link overloading. This is indicated by the results. The results for request redirection with the introduction of a new CDN server demonstrate higher standard deviation values, unlike other previous measurements. The higher values could be linked to H1. When H1 is flooded with GET request responses emanating from H2, it takes more time for H1 to complete the uonbi.com. Similarly, the median time H1 takes to complete uonbi.com test pages is less compared to that of H4. This scenario indicates that SDN reduces content load time compared to traditional DNS approach, subsequently improving users request and quality of service.

Results achieved by investigating side effects of client-side DNS caching are perplexing. Concerns such as the timing effects of a refreshed page and how quickly that page refreshes is still a paradox to the researcher. While the author expected errors such as page timeouts as a result of DNS caching, timing errors were not. Our proposed redirection approach has not yet attracted rigorous examination to assess its scalability. Perhaps, this is because the SDN technology is still growing compared to traditional legacy networks. Further, scalability and performance research is not publicly available as in the case of a purely reactive technique as the one this research has proposed.

VI. CONCLUSION

In this paper, we investigated end user redirection performance in Content Development Network (CDN) using Software Defined Networking (SDN)-based network. The study has successfully demonstrated SDN-based approach can be used to redirect users away from accessing content on a non-optimal CDN server to a better performing CDN replica server. This might be the case when the network changes or if the user side contains stale DNS information as a result of client-side DNS caching. The study focused majorly on improving end-user access to content stored in CDNs, a logical continuation of this work will be designing and implementing end user redirection algorithm that can make use of end user redirection approach proposed by this study. Further, live implementation of the proposed technique to ascertain its scalability provides another area of improvements. This is because this study used a virtualized environment to carry out performance tests.

REFERENCES

[1] M. Ghasemi, P. Kanuparthy, A. Mansy, T. Benson, and J. Rexford, "Performance Characterization of a Commercial Video Streaming Service," IEEE Internet Comput., vol. 16, no. 5, pp. 1–15, 2016.

[2] M. Jansen, "EDNS0 Client-Subnet for DNS based CDNs," in The Akamai Intelligent Platform, 2014, pp. 1–24.

[3] R. Saini, "A Hybrid Algorithm for Load Balancing," Int. J. Adv. Res. Comput. Sci. Softw. Eng. Res., vol. 5, no. 7, pp. 3–8, 2015.

[4] D. Calin and H. Schulzrinne, "Intelligent content delivery over wireless via SDN," 2015 IEEE Wirel. Commun. Netw. Conf., pp. 2185–2190, 2015.

[5] A. Whitaker, M. Shaw, and S. D. Gribble, "5th Symposium on Operating Systems Design and Implementation," in The Effectiveness of Request Redirection on CDN Robustness LiminWang, 2002, p. 33.

[6] D. Juneja and A. Garg, "Collective Intelligence based Framework for Load Balancing of Web Servers," vol. 3, no. 1, pp. 64–70, 2012.

[7] J. Lay, "Request rerouting for ISP operated CDNS," New South Wales University, 2014.

[8] A. Garg, "Analysis of various techniques for improving Web performance," Ijarcsms, vol. 3, no. 3, pp. 271–279, 2015.

[9] B. Frank et al., "Pushing CDN-ISP collaboration to the limit," ACM SIGCOMM Comput. Commun. Rev., vol. 43, no. 3, p. 34, 2013.

[10] a. Shaikh, R. Tewari, and M. Agrawal, "On the effectiveness of DNS-based server selection," Proc. IEEE INFOCOM 2001. Conf. Comput. Commun. Twent. Annu. Jt. Conf. IEEE Comput. Commun. Soc. (Cat. No.01CH37213), vol. 3, pp. 1801–1810, 2001.

[11] S. Ranjan and E. Knightly, "High-performance resource allocation and request redirection algorithms for web clusters," IEEE Trans. Parallel Distrib. Syst., vol. 19, no. 9, pp. 1186–1200, 2008.

[12] A. Baggio and M. Van Steen, "Transparent distributed redirection of HTTP requests," Proc. - 2nd IEEE Int. Symp. Netw. Comput. Appl. NCA 2003, pp. 17–24, 2003.

[13] M. Elkotob and K. Andersson, "Challenges and opportunities in content distribution networks: A case study," in GC'12 Workshop: The 4th IEEE International Workshop on Mobility Management in the Networks of the Future World Challenges, 2012, pp. 1021–1026.

[14] N. Weaver, C. Kreibich, B. Nechaev, and V. Paxson, "Implications of Netalyzr ' s DNS Measurements," Icsi.Berkeley.Edu, no. II, pp. 1–8, 2011.

[15] M. S. Clara et al., "NSDI ' 16 : 13th USENIX Symposium on Networked Systems Design and Implementation," in 13th USENIX Symposium on Networked Systems Design and Implementation, 2016.

[16] G. Silvestre, S. Monnet, D. Buffoni, and P. Sens, "Predicting popularity and adapting replication of internet videos for high-quality delivery," Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS, pp. 412–419, 2013.

[17] P. Georgopoulos, M. Broadbent, B. Plattner, and N. Race, "Cache as a service: Leveraging SDN to efficiently and transparently support video-on-demand on the last mile," in Proceedings - International Conference on Computer Communications and Networks, ICCCN, 2014, pp. 1–9.

[18] A. Baggio, "Yet another redirection mechanism for the World-Wide Web ?," 1999.

[19] C. Xu, B. Chen, and H. Qian, "Quality of Service Guaranteed Resource Management Dynamically in Software Defined Network," J. Commun., vol. 10, no. 11, pp. 843–850, 2015.

[20] I. Khalil, E. Neuhold, A. M. Tjoa, L. Da Xu, and I. You, "Information and communication technology: Third IFIP TC 5/8 International Conference, ICT-EurAsia 2015 and 9th IFIP WG 8.9 working conference, CONFENIS 2015 held as part of WCC 2015 Daejeon, Korea, October 4-7, 2015 proceedings," in IFIP International Federation for Information Processing 2015, 2015, vol. 9357, no. 10, pp. 13–19.

[21] A. A. R. Curtis, J. C. J. J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: scaling flow management for high-performance networks," Proc. {ACM} {SIGCOMM} 2011 Conf. Appl. Technol. Archit. Protoc. Comput. Commun. Toronto, {ON}, Canada, August 15-19, 2011, pp. 254–265, 2011.

[22] I. Khalil, E. Neuhold, A. M. Tjoa, L. Da Xu, and I. You, "Information and communication technology: Third IFIP TC 5/8 International Conference, ICT-EurAsia 2015 and 9th IFIP WG 8.9 working conference, CONFENIS 2015 held as part of WCC 2015 Daejeon, Korea, October 4-7, 2015 proceedings," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9357, pp. 13–19, 2015.

[23] Akamai, "Request Router," New Delhi, 2016.

[24] R. Wang, D. Butnariu, and J. Rexford, "OpenFlow-Based Server Load Balancing Gone Wild Into the Wild : Core Ideas," Hot-ICE'11 Proc. 11th USENIX Conf. Hot Top. Manag. internet, cloud, Enterp. networks Serv. Serv., p. 12, 2011.

[25] E. Harahap and J. Wijekoon, "Modeling of Router-based Request Redirection for Content Distribution Network," Int. J. Comput. Appl., vol. 76, no. 13, pp. 37–46, 2013.