An Architecture for Distributed Content Delivery Network

Jaison Paul Mulerikkal, Ibrahim Khalil
Distributed Systems and Networking
School of Computer Science, RMIT University,
Melbourne 3000, Australia
Email: jmulerik, ibrahimk@cs.rmit.edu.au

***Abstract*— Commercial Content Delivery Networks create their own network of servers around the globe to effectively deliver Web content to the end-users. The peering of Content Delivery Networks (CDN) would increase the efficiency of commercial CDNs. But still the high rental rates resulting from huge infrastructure cost make it inaccessible to medium and low profile clients . Academic models of peer-to-peer CDNs aim to reduce the financial cost of content distribution by forming volunteer group of servers around the globe. But their efficiency is at the mercy of the volunteer peers whose commitment is not ensured in their design. We propose a new architecture that will make use of the existing resources of common Internet users in terms of storage space, bandwidth and Internet connectivity to create a Distributed Content Delivery Network (DCDN). The profit pool generated by the infrastructure savings will be shared among the participating nodes (DCDN surrogates) which will function as an incentive for them to support DCDN. Since the system uses the limited computing resources of common internet users, we also propose a suitable load balancing (LB) algorithm so that DCDN surrogates are not burdened with heavy load and requests are fairly assigned to them. Simulations have been carried out and the results show that the proposed architecture (with LB) can offer same or even better performance as that of commercial CDN.**

I. Introduction

As the Web content and the Internet traffic increases, individual Web servers find it difficult to cater to the needs of end-users [10], [15], [12]. A proposed solution is to replicate the same Web content around the globe for easy access of end-users [3]. However, it is not financially viable for individual content providers to set up their own server networks. An answer to this challenge is the concept of Content Delivery Network (CDN) [6] that allows a number of content providers to upload their Web content into the same network of Web servers and thereby reduce the cost of content replication and distribution [15].

In a typical CDN environment (Figure 1), the replicated Web server clusters are located at the edge of the network to which the end-users are connected. There are a number of CDNs available around the globe [6], [15], [1] and are collectively called as *Conventional CDN architectures* in this paper. Conventional CDN architectures - Commercial CDN and Academic CDN - have got their own advantages and disadvantages, but the marjor pitfalls of them are:

- High rental rates resulting from huge infrastructural cost.
- Efficiency of academic CDNs is at the mercy of the volunteer peers whose commitment is not ensured in their design.

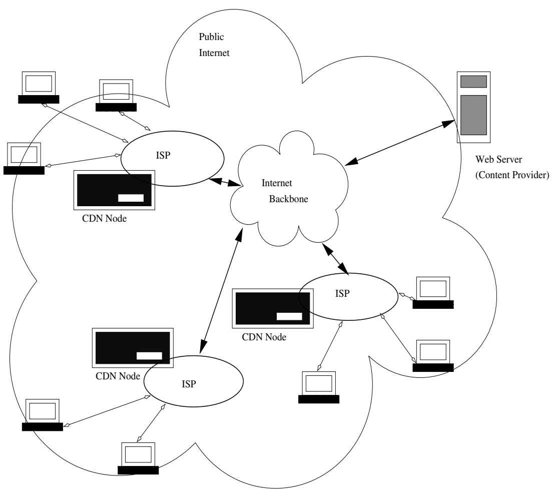


Fig. 1. CDN and Web Content Distribution

A lot of work has been done to address these issues. Globule - which is envisaged as Collaborative Content Delivery Network (CCDN) aims to provide performance and availability through Web servers that cooperate across a wide area network [11]. J. Coppens and P. Demeester propose the idea of a self-organizing Adaptive Content Distribution Network (ACDN), which will use a less centralized replica placement algorithm - (COCOA - Cooperative Cost Optimization Algorithm) to push content more to the clients [4].

This paper proposes a new architecure for CDN to overcome the major limitations of Commercial and Academic models. According to the proposed Distributed Content Delivery Network (DCDN) common Internet users become content space providers (they are called DCDN surrogates). It will empower Internet users to fetch revenue from their existing resources (bandwidth, processing power and Internet connectivity) which they share in support of DCDN. While DCDN offers promising features, one basic question needs to be answered: can DCDN be expected to produce the same or better quality of service as that of commercial CDNs by pushing content to the end-users? To answer this we have proposed a load balancing (LB) algorithm for DCDN servers based on efficiency analysis of DCDN surrogates. The LB mechanism ensures that common internet users with limited computing resources are not burdened with heavy load and requests are fairly assigned to them. Performance evaluation of DCDN architecture using simulation techniques shows that proposed architecture (with LB) can indeed offer same or even better performance as that of commercial CDN in some cases.

In the rest of the sections, first we discuss the background of and the rationale behind DCDN. Then we introduce the proposed DCDN architecture. On the basis of queuing analysis of DCDN surrogates, we propose a load balancing algorithm for DCDN servers and finally present the simulation results and conclusions.

II. Background

A. Conventional CDN architectures

1) Commercial Architecture: It is basically a client-server architecture. The classical example is of Akamai. We briefly discuss the working of Akamai to understand commercial CDN.

In Akamai's centralized approach, the customers (the content providers) hire their share of space in Akamai servers to support the distribution and easy download of their Internet content. It is done by installing a worldwide network of more than twenty thousand Akamai surrogate servers [5].Akamai provides the required service in the following fashion:

1) The client's browser requests the default Web page at the Content Provider's site. The site returns the Web page index.html.
2) The html code contains link to some content (eg: images) hosted on the Akamai owned server.
3) As the Web browser parses the html code it pull the content from Akamai Server [16].

If a particular CDN provider is unable to provide quality service to the end-user requests, it may result in Service Level Agreement (SLA) violation and adverse business impact. In such scenarios, one CDN provider partner with other CDN provider(s), which has caching servers located near to the end-user and serve the users request, meeting the Quality of Service (QoS) requirements. This is called peering of CDNs [9], [13].

2) Academic Architecture: They are peer-to-peer architectures. Distributed computer architectures labelled *peer-to-peer* are designed for the sharing of computer resources (content, storage, CPU cycles) by direct exchange, rather than requiring the intermediation or support of a centralized server or authority. The same technique has been adopted for creating reliable CDN for the propagation of Internet content [2], [14].

In its most basic form, a peer-to-peer content distribution system creates a distributed storage medium that allows for the publishing, searching, and retrieval of files by members of its network. So, instead of delegating content delivery to an external company (like Akamai), content providers can organize together to trade their (relatively cheap) local resources against (valuable) remote resources [8].

B. Limitations of Existing CDN Architectures

Commercial CDN providers compete each other and set up costly infrastructure around the globe. The huge financial cost involved in setting up and running a commercial CDN compels the commercial CDN providers to charge high remuneration from their clients (the content providers). Usually this cost is so high that only large firms can afford it. As a result, Internet content providers of medium and small sizes are not in a position to rent the services of commercial CDN providers.

Moreover, the revenue from content distribution is monopolized. That means, only large CDN providers with huge infrastructure around the world are destined to amass revenue from this business. At the same time, the resources in terms of processing power, storage capacity and the network availability of large number of common Internet users are ignored who would support a content delivery network for proportionate remunerations.

The academic CDNs are non-profitable initiatives and serve only the content providers who own *their own* network of servers around the globe. They do not provide a built-in network of independent servers. Eventually, the risk and responsibility of running CDN goes back to the content providers themselves. Or else, they have to become a part of a voluntary net of servers whose efficiency is at the mercy of the volunteer peers. The commitment of volunteer peers is not ensured in academic CDN architecture.

C. DCDN - An Effective Alternative

In the light of above discussion, we find that there is a need for much reliable, responsible and scalable CDN architecture, which can provide reliable content distribution and delivery without additional infrastructure investment.

DCDN aims at involving general Internet users with comparatively high bandwidth of Internet connection (broadband or higher) to form a highly distributed CDN. It will not demand installation of fresh new infrastructure. This approach is supposed to reduce the economic cost. The acquired new value (or profit pool) is to be shared among the DCDN surrogates through proper accounting and billing mechanism. It will serve as an incentive for the DCDN surrogates to share their resources in support of DCDN, unlike in the case of academic models. Moreover, the efficiency of the content retrieval in terms of response time is expected to improve, since the content is pushed very much to the local levels. It will also reduce network traffic, since many clients may find the desired content from locally placed surrogates.

III. Architecture: Distributed Content Delivery Network

In order to provide a highly distributed network of DCDN surrogates a basic structure of commercial client/server CDN is adopted with novel peer to peer concepts. Therefore the DCDN architecture will be a hybrid architecture which integrates some of the major features of conventional client/server CDN and an academic peer-to-peer CDN.

A. DCDN Framework

A collection of Local DCDN Servers and innumerable DCDN Surrogates are networked together to deliver requested Internet content to the clients. The main elements of DCDN architecture Content providers, DCDN servers and DCDN surrogates are arranged in a hierarchical order as in Figure 2.

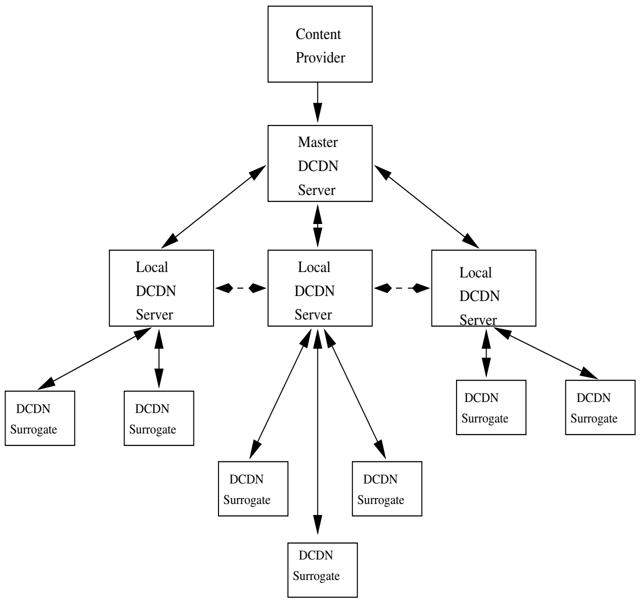


Fig. 2. DCDN hierarchy

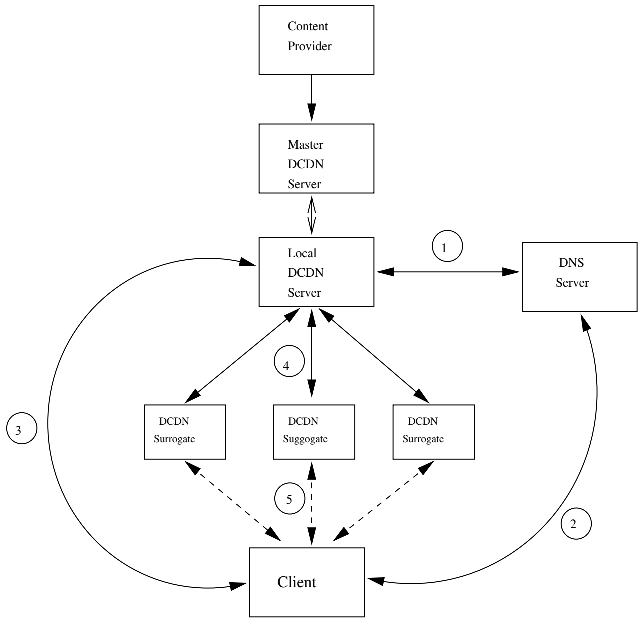


Fig. 3. DCDN Content Delivery

Content Provider: It is that entity that request to distribute its Web content through DCDN.

DCDN Administrators: Rather than a technical entity, it is a managerial/business entity. The entire DCDN network is managed, supported and run by a team of administrators.

DCDN Servers: DCDN servers are basically redirectors with load balancing properties. They do not store any content as such. They are of two types: Master and Local.

- *DCDN Master Servers*: Master DCDN servers are the first point of contact of a content provider. A global network of Master DCDN servers are set up in such a way that every network region will have at least one Master DCDN server. (Here, network region can be geographical regions or network regions identified on the basis of other criteria like, network traffic, network volume, etc.).
- *DCDN Local Servers*: They are placed much more local than that of Master servers and a number of Local servers can come under the service of a single Master server. They have got two major functions. Firstly, they decide where to place the content (among the surrogates) and keep log of it. Secondly, they find out and return the IP address of the best available surrogate a client on request for a particular content under the care of DCDN. In doing so, they also function as a load balancer that will protect the surrogates in the network from being overloaded.

DCDN Surrogates: As explained before, DCDN surrogates are the large number of Internet users who offers resources in terms of storage capacity, bandwidth and processing power to store and make available DCDN Web content.

DCDN Client: The client refers to an end user, who makes a request for a particular Internet content using a Web browser.

B. Distribution of Content - The Process

The aim is the place the replica of the content as close as possible to the clients. DCDN shall achieve this goal in the following way.

The content providers approach DCDN administrators and once the service agreement is reached, they can upload their content to DCDN net. This can be done either through the Master DCDN servers or through the Local DCDN servers assigned by the Master DCDN servers. The Master servers or the Local servers push replicas to the surrogates in their own region and keep a track of these records. The Master servers will have more universal knowledge about it (Like, what are the network areas in which a particular content is distributed) and the Local servers will have more local knowledge of the location of the content (That is, which are the surrogates that actually holding a particular content).

On request from a Local Server, a surrogate may share the replicas with other surrogates in a peer-to-peer fashion. This will offload the Local severs from additional workload. The process will make sure that the local Server still has the knowledge about the replicated content in the new surrogate/s.

However, the content providers need not choose to distribute their content in a true global manner. If they want DCDN to support only for some region/s, they can request for regional support too. In that case, the administrators (with the help of Master servers) choose only those Local DCDN servers, which are set by the parameters given in the QoS (Quality of Service) agreement between the content provider and DCDN administration. (For example, if the content is to be distributed in the Asia and Asia Pacific region, it is sent to the Local DCDN servers at those regions only).

C. Content Delivery to a User

The local DCDN server, which is envisaged as a redirector, will follow the DNS protocol. It will take care of the queries related to the Websites under the care of DCDN. This information is shared with other DNS servers too. So, when there is a request for a Website under the care of DCDN, the DNS redirectors will redirect it to the nearest available Local DCDN server. The DCDN Local server searches the log of active surrogates holding those files using a suitable technique (Eg. Distributed Hash Table (DHT) algorithm). It will then make a decision based on load balancing and other relevant parameters (availability of full or partial replica content, bandwidth, physical/online nearness, etc) and will return the IP address of the best suitable surrogate to the client. The client fetches the content from the respective surrogate. Diagrammatical representation of the above process is given in Figure 3. If the requested content is not available in a Local DCDN net, contingency plans are evoked to fetch it from surrogates under other Local DCDN net. They are not discussed in this paper.

IV. Experimental Methodology

Performance of DCDN surrogates constitute the most important part of DCDN architecture, for they have limited resources, especially in terms of Internet bandwidth. So, we analyze the performance of DCDN surrogates in terms of total system delay and rejection rate for different queuing theory models. The results of queuing theory analysis give us an idea about the possible design of surrogates. A suitable load balancing algorithm for DCDN servers is proposed on the basis of surrogate design. Replicating these parameters, the DCDN architecture is simulated against a commercial CDN architecture and the results are compared.

A. Queuing Metrics

We assume a Poisson process of randomly spaced requests in time and an exponential distribution of service-time which result in M/M/c/k model of queuing analysis, where c is number of servers and k is the number of requests in the system including the queue. We assume to replicated the effect of multiple servers in a single surrogate by running more than one DCDN surrogate daemons (as multiple processes or threads) within the same machine. Total system delay in a DCDN surrogate for different M/M/c/k models are found out using the following formulae as explained by D. Gross and C. M. Harris [7].

- *Service Time (S):* In our case, it is the transmission time, which is equal to; $S = \left(\frac{Filesize}{UpstreamCapacity}\right)$
  Therefore, Service rate of the DCDN surrogate $\mu = (\frac{1}{S})$
- *Server Utilization (ρ):*
  $\rho = (\frac{\lambda}{c\mu})$ for M/M/c/k queuing model
  where λ is the arrival rate of requests to DCDN surrogate.
- *Effective arrival rate λ_e:* $\lambda_e = \lambda(1 - P_k)$ where$P_k$is the probability of k requests in the system.
- *Probablity of zero requests in the system P_0:*

$$P_0 = \left[\sum_{i=0}^{c-1} \frac{(\lambda_n/\mu)^i}{i!} + \frac{(\lambda/\mu)^c}{c!} \frac{1-\rho^{k-c+1}}{1-\rho}\right]^{-1}$$

  – Probablity of n customers in system for $0 \leq n \leq c$
  $P_n = \left(\frac{\frac{\lambda}{\mu})^n}{n!}\right) P_0$
  – Probablity of n customers in system for $c \leq n \leq k$
  $P_n = \left(\frac{\frac{\lambda}{\mu})^n}{c!c^{n-c}}\right) P_0$
- *Average number of requests in the queue L_q:*
  $L_q = \sum_{n=c}^{k}(n-c)P_n$
- *Average number of requests in the system L:*
  $L = L_q + \frac{\lambda}{\mu}(1 - P_k)$
- *Total System Delay W:* for M/M/c/k, $W = \frac{L}{\lambda_e}$
- *Rejection Rate:* The number of requests that will be lost due to congestion per unit time is given by: λP_k

Using a Web Page Analyzer[1] it is found that the average size of Web pages of medium size content providers (example: www.rajagiritech.ac.in) is about 30 KB. It is assumed that the surrogates are supposed to have a minimum of DLS/Cable Web access. The minimum capacity of DLS/Cable line is rated at 768 Kbps downstream and 128 Kbps upstream. The performance for different queuing models are evaluated by assuming that the mean arrival rate of requests (λ) to a surrogate is not beyond its service rate. *Total System Delays* for those models are compiled in Figure 4(a) and their *Rejection rates* are compiled in Figure 4(b). A server originally intending to serve a single request at a time may actually end up serving 2 or 3 with reduced rates, for M/M/1/1 the service rate is μ; for M/M/2/2 it is $\mu/2$, and for M/M/3/3 the rate is $\mu/3$. Based on these results, the basic ideas for load balancing algorithm of a DCDN server are laid out.

B. Load Balancing Algorithm for DCDN Servers

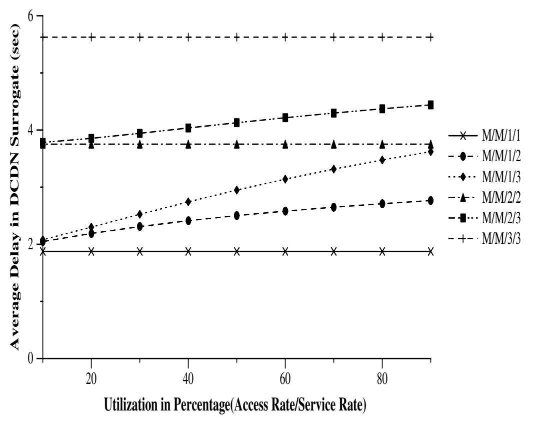
By carefully analyzing the Utilization v/s Queuing Delay graph (Figure 4(a)) and Utilization v/s Rejection rate graph (Figure 4(b)), the following inferences are made.

1) The best performance is expected from DCDN, when surrogates follow M/M/c/c queuing models.
2) Loss of requests can be reduced by increasing the number of requests in the whole system by increasing the value of k in M/M/c/k queuing model.

Based on the above conclusions an algorithm based on M/M/c/c model is found to be more scalable and efficient than other models both in terms of total system delay time and rejection rate. We make the assumption that DCDN surrogates will be designed to support M/M/c/c queuing model of request streams where the value of c will be proportional to the processing capacity of surrogates. The following LB algorithm for equal sever load is proposed.

```
Load Balancing Algorithm for DCDN Server
1: let, DCDN server has the knowledge of the service rate ($\mu$) of
   its surrogates;
2: let, DCDN server is aware of the requests send to ($\lambda$) its
   surrogates;
3: let, DCDN surrogates support only M/M/c/c queuing models;
4: $c$ is the maximum number of requests allowed in a particular
   surrogate;
5: Web requests arrive at the Local DCDN Server;
6: **if** requested content is available in the Local DCDN surrogate
   network **then**
7:     **if** there are surrogates with $P_0$ (Probability of NO requests)
       equal to 100 (i.e., idle surrogates) **then**
```

[1]Available at: http://www.websiteoptimization.com/services/analyze/



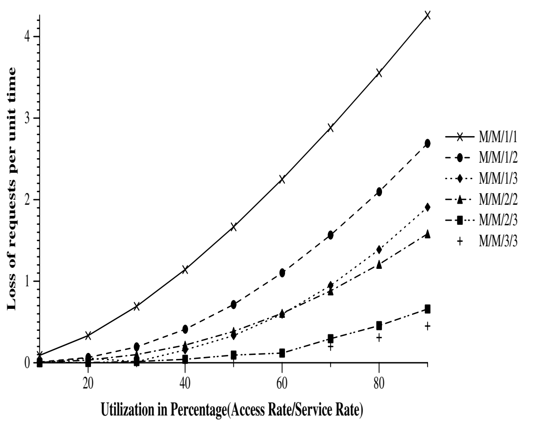


Fig. 4. (a)Surrogate Utilization v/s Total System Delay.(b) Surrogate Utilization v/s Request Rejection Rate

```
8:      send request to the surrogate with highest $c$ value ( i.e., to
        surrogate with highest service capacity);
9:    else
10:     while search do not exceed the Max_Trial_ Number do
11:       find the surrogate with lowest Server Utilization
          ($\rho = (\frac{\lambda}{c\mu})$);
12:     end while
13:     send request to that surrogate;
14:   end if
15: else
16:   redirect request to other Local DCDN server who has the
      requested content;
17: end if
```

V. Simulation and Results

DCDN architecture with LB algorithm proposed in last section is simulated against commercial CDN architecture using Opnet IT Guru - Academic Version in a standard PC. The performance of DCDN architecture is compared against the performance of commercial client-server CDN architecture in terms of page response time, utilization of DCDN server (load balancer, in case of conventional CDN) and utilization of DCDN surrogate (server, in case of conventional CDN).

Four scenarios were created for simulations by altering the number of clients, number of surrogates(or Servers) and link capacity in the simulation setup for commercial CDN and DCDN. They are:

Commercial CDN: It is the standard CDN setup with 150 HTTP clients. They were served by a server farm of 3 CDN servers. The link capacity of the CDN servers was set to 100 Mbps to reflect the fact that commercial CDN can afford more resources. Round robin algorithm was used for load balancing.

DCDN - Scenario 1: The scenario is changed to DCDN setup serving same number of (150) HTTP clients. The three CDN servers in the previous setup was replaced with six DCDN surrogates(work stations). DCDN surrogate link capacities were reduced to 10 Mbps to simulate the fact that they will have lower link capacity than of commercial CDN servers. Optimum server load algorithm is used for load balancing.

DCDN - Scenario 2: In this DCDN setup, the number of clients was reduced to 75 by keeping all other parameters the same as the previous setup. Optimum server load algorithm is used for load balancing.

DCDN - Scenario 3: The number of clients was further reduced to 30 by keeping all other parameters intact in this DCDN setup.Optimum server load algorithm is used for load balancing.

A. Simulation Results

The results of the simulations - page response time, surrogate (or server) utilization and DCDN server (load balancer) utilization - are compiled in Figure 5(a), Figure 5(b) and Figure 6 respectively. Page response time is the interval between the instance at which an end-user at a terminal enters a request for Website and the instance at which the Webpage is received at the terminal. CPU utilization is the measure of how much of the available processing power in an enterprise is used. In our case, it will give us an idea about the amount of resources we need to put in terms of processing power by a single DCDN surrogate and a DCDN Server.

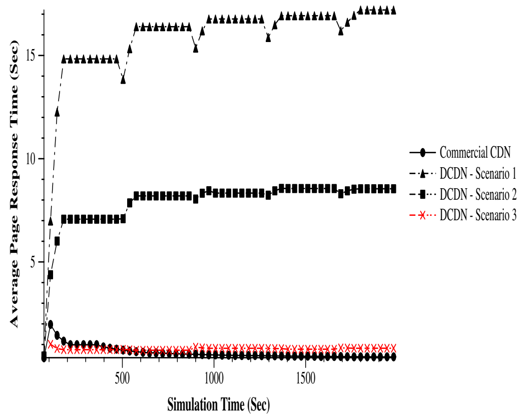
VI. Evaluation

The whole exercise was to find out the feasibility of DCDN architecture in comparison with commercial CDN architecture. The average page response time was the key parameter for our evaluation. The DCDN scenarios 1 and 2 could not reproduce the performance of commercial CDN but left enough traits to suggest that DCDN setup could produce it if the request load is further reduced. In DCDN scenario 3, the request load was further reduced by reducing the number of clients to 30.

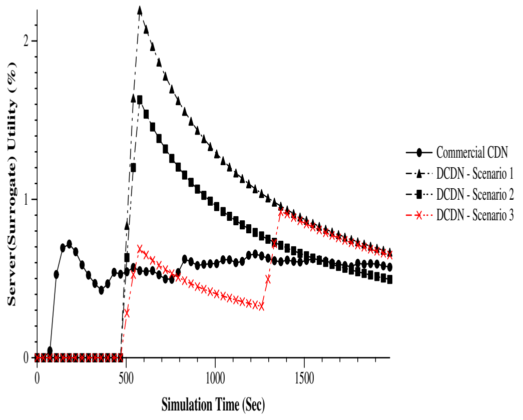
The results show that this scenario could produce the same performance as that of commercial CDN which handled 150 clients. It gives us a valuable inference: DCDN architecture can replicate the performance of commercial CDN, if the critical parameters are adjusted well enough.

When we scale the simulations results, we can say that to replicate the performance of commercial CDN using DCDN architecture, a server in commercial CDN setup has to be replaced with 2 x 5 = 10 surrogates in DCDN setup (for, the number of surrogate servers was doubled and the number of clients was reduced to 1/5) when the link capacity of DCDN surrogate is reduced to 1/10 of that of commercial CDN.

Further research and experiments have to be done to verify this exact ratio. However, the above results suggest that the



(a)



(b)

Fig. 5. (a)SPage Response Time.(b) Surrogate (or Server) CPU Utilization

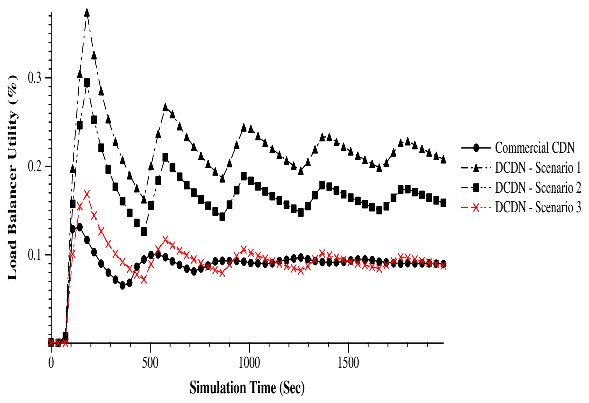


Fig. 6. DCDN Server (Load Balancer) Utilization

ratio between the number of commercial CDN servers and DCDN surrogates will fall within a reasonable limit. It also suggests that the DCDN architecture could fall well within financial viability limits for the number of surrogates needed to support the system will not be that high.

Surrogate CPU utilization in DCDN setup also shows that it will not take a big chunk of the processing capacity of the surrogate work-stations. DCDN server CPU utilization is as negligible as that of load balancer in the commercial CDN setup. It suggests that running Local as well as Master DCDN servers within the Internet cloud will not be a bottleneck.

VII. Conclusion and Future Work

The simulation results show that by using an efficient load balancing algorithm at DCDN servers, we could run an effective DCDN for fast content retrieval, provided we keep proper balance between the volume of requests and the number of surrogates available to handle it. In this architecture, the Internet users will take their share of responsibility in maintaining DCDN and will inherit their share of revenue.

A. Future Work

Protecting the integrity and security of the content being distributed through DCDN is of utmost importance since the content is being heavily decentralized and vulnerable to attacks from hackers. Unique and effective protocols and security measures are to be developed against possible content tampering in DCDN surrogates. Accounting and billing mechanism which will measure the service of the DCDN surrogates to award them proportionate remuneration is another important future research area.

References

[1] Content distribution networks(cdn) - research directory, May 2007.

[2] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4), December 2004.

[3] R. Burns, R. Rees, and D. Long. Efficient data distribution in a web server farm. *IEEE Internet Computing*, 5(5):56 – 65, September - October 2001.

[4] J. Coppens, T. Wauters, F. D. Turck, B. Dhoedt, and P. Demeester. Design and performance of a self-organizing adaptive content distribution network. IEEE/IFIP Network Operations Management Symposium 2006, Vancouver, Canada, April 2006.

[5] J. Dilley, B. Maggs, J. Parlkh, H. Prokop, R. Sitaraman, and B. Welhl. Globally distributed content delivery. *IEEE Internet Computing*, page 50 to 56, September-October 2002.

[6] F. Douglis and M. Kaashoek. Scalable internet services. *IEEE Internet Computing*, 5(4):36 – 37, 2001.

[7] D. Gross and C. M. Harris. *Fundamentals of queueing theory*. Wiley, London, 1998.

[8] B. Halderen and G. Pierre. *Globule User Manual – Version 1.3.1*, February 2006.

[9] I. Lazar and W. Terrill. Exploring content delivery networking. *IT Professioinal*, 3(4):47 – 49, July - August 2001.

[10] B. Molina, V. Ruiz, I. Alonso, C. Palau, J. Guerri, and M. Esteve. A closer look at a content delivery network implementation. In *Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean*, volume 2, pages 685 – 688, Dept. of Commun., Univ. Politecnica de Valencia, Spain, May 2004.

[11] G. Pierre and M. van Steen. Globule: A collaborative content delivery network. *IEEE Communications Magazine*, pages 127 – 133, August 2006.

[12] F. Presti, N. Bartolini, and C. Petrioli. Dynamic replica placement and user request redirection in content delivery networks. In *IEEE International Conference on Communications*, volume 3, pages 1495–1501, Dubrovnik, Croatia, May 16 - 20 2005.

[13] J. B. R. Buyya, A. Pathan and Z. Tari. A case for peering of content delivery networks. *IEEE Distributed Systems Online*, 5(5):56 – 65, October 2001.

[14] B. H. S. Sivasubramanian and G. Pierre. Globule: a user-centric content delivery network. poster presented at 4th International System Administration and Network Engineering Conference, September-October 2004.

[15] A. Vakali and G. Pallis. Content distribution networks - status and trends. *IEEE Internet Computing*, pages 68 – 74, November - December 2003.

[16] Wikipedia. Content delivery networks (cdn), April 2007.