

Understanding data-center driven content distribution

Vijay Kumar Adhikari, Sourabh Jain, Gyan Ranjan, and Zhi-Li Zhang*

Department of Computer Science & Engineering, University of Minnesota
Minneapolis, MN

viadhi@cs.umn.edu, sourj@cs.umn.edu, granjan@cs.umn.edu,
zhzhang@cs.umn.edu

ABSTRACT

Cloud services are radically changing the way we access the information from the Internet and how the “digital goods” are delivered to users. To enable such services, large scale data centers are deployed at various geographical locations, and the contents are replicated at multiple locations to achieve better availability and reliability. However, it is not clear given these multiple data centers how content providers place the content and how a user query is served from the cloud.

In this paper, we propose a comprehensive, yet simple and intuitive, approach that can be employed to understand the data center driven content distribution using active measurements. Additionally, applying this method on YouTube, we find that YouTube uses data-centers in more than 50 locations, and that it employs an interesting “3-step” approach to deliver videos to clients in a location-aware manner.

1. INTRODUCTION

Cloud driven services have witnessed tremendous growth in recent years. Content providers such as Google, Yahoo, MSN deploy large scale data centers and use these data centers to serve content to their clients.

Recent research [2,3] has focused primarily on building efficient data centers. Little is known about coordination amongst multiple data centers, load-balancing strategies and its possible impact on performance.

This paper tries to answer the following question. “How can we uncover the data-center cloud, in terms of identifying the nodes in the cloud, corresponding IP addresses and their geo-location, and understand the

*This work is supported in part by the NSF grants CNS-0721510, CNS-0905037, CNS-1017647 and the DTRA Grant HDTRA1-09-1-0050.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM CoNEXT Student Workshop, November 30, Philadelphia, USA.

Copyright 2010 ACM 978-1-4503-0468-9/10/11 ...\$5.00.

content distribution and load-balancing strategies *without having any inside information?*”

To answer this question, this paper makes two contributions: (a) We propose a framework for uncovering the clouds from scratch using active measurements, and (b) We apply proposed methodology to YouTube, and obtain interesting details on their data-centers and load-balancing strategies. Compared to the earlier strategy of proportional load-balancing among its 6 datacenters [1], YouTube’s video delivery framework has gone a major restructuring under Google.

2. METHODOLOGY

Our methodology consists of the following steps:

HTML parsing For HTML-based services, HTML pages can be downloaded and parsed to extract different hostnames being used by the content provider.

Binary content Binary content such as Adobe Flash videos present more challenges. They are hard to automate using text-based browsers or simple HTTP libraries. To deal with such problems, we propose a proxy-based approach as discussed in Sec 3.

Guessing hostnames From the structure present in the hostnames we try to guess more hostnames.

DNS look-ups Next, DNS queries can be made for all the hostnames from a large number of nodes in different geographic regions.

Geolocating IPs Once the IP addresses and hostnames have been obtained, the IPs can be geolocated in a number of steps such as latency and city codes in the names.

Load-balancing strategy The load balancing strategy can be studied by asking questions such as “Are the requests being distributed to all the hostnames?”, “Are the requests being served from locations close to the users?”, “Are they using some CDN services?”, and so on that can be answered based upon the information obtained in the earlier steps.

3. EXPERIMENTS & RESULTS

Extracting hostnames By parsing HTML pages, we obtained about 200 hostnames that can be put in the

following groups (m and n being integers).
s.youtube.com, s2.youtube.com, s.ytimg.com,
i(n).ytimg.com, v(m).lscache(n).c.youtube.com

Since playing a large number of videos on PL nodes was infeasible, we played the videos on computers inside our lab but configured the lab computers to use the PlanetLab nodes as proxy servers. We installed Squid proxy servers on PlanetLab nodes and each of them were used by exactly one client as a HTTP proxy server.

From this experiment, we were able to uncover even more hostnames that were delivering video contents. Those hostnames could be put in the following groups.

tc.v(m).lscache(n).c.youtube.com,
v(m).cache(n).c.youtube.com, r(code).c.youtube.com

As above, m and n are integers and “code” is a string that contains city/airport names among other things. These hostnames were not seen in the HTML parsing steps.

We were also able to guess hundreds of additional hostnames based upon the structure of the observed hostnames.

Extracting IP addresses and geolocation We resolved all the hostnames from about 460 PlanetLab nodes. We found 5169 IP addresses from 128 /24 prefixes. As we can see in Fig. 1, we covered a very large subset of the IP addresses being used by YouTube.

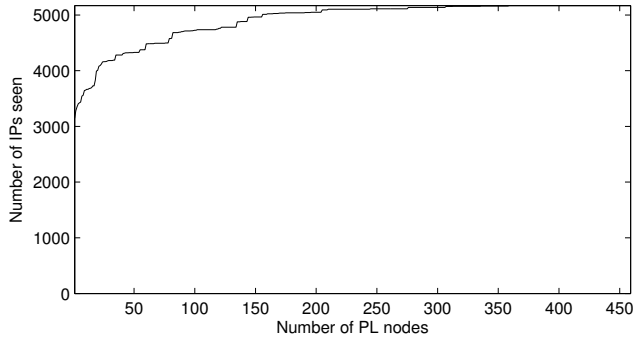


Figure 1: IP addresses seen vs number of PL nodes employed.

We used a multi-tiered approach to geolocate the IP addresses. We computed ping latency to all the IP addresses from multiple PL nodes. For the first step, we used the city codes in some hostnames to map IPs to cities. We also verified that the PL nodes close to that city had the lowest latency to those IP addresses. With these IPs with known location, we then used clustering approach to geolocate remaining IP addresses. We were able to geolocate YouTube IPs to more than 24 cities around the world.

Load-balancing Based upon the video playback data from proxy-based experiment, YouTube seems to be using 3-step load-balancing strategy. For the first step, each video id is uniquely mapped (irrespective of the

geographic location of the client) to one of 192 “logical” hostnames that the browser contacts for the video content. In fact, as Fig. 2 shows the hostnames were serving about the same percentage of videos.

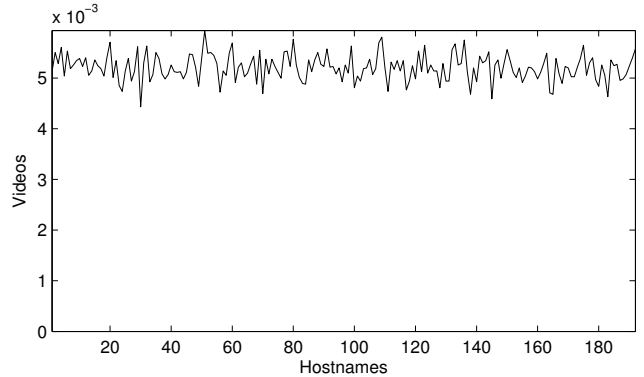


Figure 2: Percentage of videos mapped to each hostname.

As a second step, their DNS servers return different IP for those hostnames based upon the geographic location of the client. However, this location awareness is employed at a coarser level. There are multiple instances when the mapped IP addresses did not come from the location with lowest latency. In fact, 96% of the time PL nodes were talking to the nodes for which the latency was in the first quartile, whereas only 38% of the time, the nodes were talking to the host with the lowest latency. In the third step, the “logical” hostname redirects to a different hostname if it cannot serve a request. Initial findings suggest that there are more redirects for less popular videos. For instance, a dummy video uploaded by the authors was the one that generated most number of redirects during our study.

4. SUMMARY & FUTURE DIRECTIONS

In this paper, we proposed an intuitive methodology to uncover data-center and content-delivery strategies from large content-providers. We also employed the proposed methodology on YouTube as a case study. In the future, we plan to make this methodology more generic and try it on other content-providers.

5. REFERENCES

- [1] V. K. Adhikari, S. Jain, and Z. Zhang. YouTube Traffic Dynamics and Its Interplay with a Tier-1 ISP: An ISP Perspective. In *IMC*, 2010.
- [2] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. V12: a scalable and flexible data center network. *SIGCOMM Comput. Commun. Rev.*, 39(4):51–62, 2009.
- [3] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat. Portland: a scalable fault-tolerant layer 2 data center network fabric. *SIGCOMM Comput. Commun. Rev.*, 39(4):39–50, 2009.