A DNS Reflection Method for Global Traffic Management

Cheng Huang
Microsoft Research

Nick Holt
Microsoft Corporation

Y. Angela Wang
Polytechnic Institute of NYU

Albert Greenberg
Microsoft Research

Jin Li
Microsoft Research

Keith W. Ross
Polytechnic Institute of NYU

Abstract

An edge network deployment consists of many (tens to a few hundred) *satellite* data centers. To optimize end-user perceived performance, a Global Traffic Management (GTM) solution needs to continuously monitor the performance between the users and the data centers, in order to dynamically select the "best" data center for each user. Though widely adopted in practice, GTM solutions based on active measurement techniques suffer from limited probing reachability. In this paper, we propose a novel *DNS reflection* method, which uses the GTM DNS traffic itself to measure the performance between an arbitrary end-user and the data centers. From these measurements, the best data center can be selected for the user. We have implemented and deployed a prototype system involving 17 geographically distributed locations within the Microsoft global data center network infrastructure. Our evaluation of the prototype shows that the DNS reflection method is extremely accurate and suitable for GTM. In particular, at the 95 percentile, the measured latency is 6 ms away from Ping, and the selected data center is 2 ms away from the ground-truth best.

1 Introduction

In an era where 100 ms extra delay can cost 1% drop in sales [10], cloud service providers are examining all possible measures that can reduce end-user perceived latency. One aggressive strategy is to deploy *satellite data centers* in addition to the traditional mega "backbone data centers", so as to construct an *acceleration platform* close to the end-users. Based on these satellite data centers, planet-scale *edge networks*, such as Google's CDN and Microsoft's Edge Computing Network, go beyond distributing static content and speeding up large downloads. They are increasingly important for accelerating dynamic cloud services, including search, email, maps, online office productivity software, etc.

An edge network deployment consists of many (tens to a few hundred) satellite data centers. To optimize end-user perceived performance, the "best" data center needs to be dynamically determined for each end-user. By serving users from the best data center, static content can be delivered with lower latency and higher throughput (as well as with less load on the network backbones). In addition, these satellite data centers can proxy TCP connections to speed-up dynamic cloud services [15]. One key challenge here is to find, for each end user, the best data center, which is a dynamic real-time optimization problem. In practice, the optimal selection does not always correlate well with geographic distance, but rather with a combination of network latency, packet loss, and available bandwidth. Furthermore, optimality changes as Internet routes flap, ISP relationships change, and the connectivity of physical networks fluctuates. Dynamically and accurately determining the best data center is the cornerstone of the Global Traffic Management (GTM) solution.

To optimize end-user perceived performance, the GTM solution needs to continuously monitor the performance between the users and the satellite data centers, in order to dynamically select the best data center for each user. Though widely adopted in practice, we argue that existing GTM solutions based on active measurements [1] suffer from limited probing reachability, and those based on passive measurements [4, 11, 18] incur high overhead and degrade performance. In this paper, we propose a novel *DNS reflection* method, which uses the GTM DNS traffic itself to measure the performance between an arbitrary end-user and the data centers. The contributions of this paper are as follows:

- We first survey existing DNS-based GTM solutions, including those that pick the geographically closest data center, those that use IP anycast to direct users to a data center, and those that use active probing or passive measurements. We argue that these existing solutions can perform poorly for a non-negligible fraction of the users. As part of this analysis, as a side result, we estimate that there are approximately 862,000 Local DNS (LDNS) servers used by all Windows Vista and Windows 7 users in the Internet today.

- We then propose a novel *DNS reflection* method, which uses the GTM DNS traffic itself to measure the performance between an arbitrary end-user and the data centers. The basic idea is to (very) occasionally have a user's DNS query redirected to and reflected by the DNS servers located in the satellite data centers, which can in turn measure the performance between themselves and the user. From these measurements, the best data center can be selected for the user.
- We implement and deploy a prototype system involving 17 of the geographically distributed locations in the Microsoft global data center network infrastructure. In our evaluation, we first show that the DNS reflection method is extremely accurate. In particular, at the 95 percentile, the measured latency is 6 ms away from Ping. We then compare the GTM solution based on our DNS reflection method with solutions based on geographic and anycast selection. In our experiments, the reflection-based GTM method is 2 ms within optimal at the 95 percentile, while the geography and anycast based GTM solutions are 74 ms and 183 ms from optimal, respectively. In other words, for the users whose performance is most precarious, the benefit of reflection-based GTM is significant.

2 Brief Overview of GTM Solutions

GTM is often implemented through a DNS system. As a simple example, suppose `CloudService.com` has an infrastructure of mega and satellite data centers. When a user wants to connect to the service, it first performs a DNS resolution for `CloudService.com`. The authoritative DNS server for `CloudService.com` responds with the IP address of the "best" data center, which has been determined from its GTM system. The GTM system provides, via the authoritative DNS server, different satellite data centers for different users.

Before presenting our approach to GTM, in this section we briefly review various GTM solutions and discuss related work.

2.1 Geography-based GTM Solutions

This type of GTM system uses geographic locations to map clients to data centers [7, 9]. Using commercial GeoLocation databases provided by Akamai, Quova, MaxMind and so on, each client's IP address is mapped to a geographic location. The data center chosen for a client is simply the data center that is geographically closest. Such a solution can work reasonably well for a large fraction of the clients, as recently shown in evaluation [3]. However, geographic-based solutions are still subject to well-known issue of Triangular Inequality Violation (TIV) of Internet distances. Moreover, a geographic-based solution ignores the dynamic nature of the Internet, such as the variation of latency and packet loss, and always assigns the same data center to a particular client.

2.2 Anycast-based GTM Solutions

This type of GTM uses IP anycast [14], for which all the data centers announce the same anycast IP address. When a client sends a packet to the anycast address, the packet is routed to the *anycast-closest* data center. The anycast-closest is governed by both intra- and inter-domain routing algorithms and policies. Although the anycast-closest data center is often the best data center in terms of latency for many clients, there are a non-negligible percentage of violations [5, 8]. In addition, anycast-based GTM solutions ignore packet loss, which could severely impact many delay sensitive online services.

2.3 GTM Solutions based on Active Measurements

Commercial GTM solutions commonly rely on active probing techniques to measure the performance between data centers and clients. For instance, the F5 3-DNS system actively probes Local DNS (LDNS) servers and uses the response time to calculate the round trip time and packet loss between the LDNS and the data center [1]. Observations collected at Internet honey-pots also suggest commercial CDNs, such as Akamai, are conducting large scale Internet measurements [13]. However, as we will soon demonstrate, active probing suffers from limited reachability, as many LDNSes are configured to never respond to active probes. Because a large percentage of the LDNSes cannot be probed, the effectiveness of active measurements is limited.

2.4 GTM Solutions based on Passive Measurements

An alternative to active probing is to infer performance between clients and data centers through passive monitoring. For instance, latency can be calculated by examining the gap between SYN-ACK and client ACK during the TCP three-way handshake. In order to monitor the performance between clients and every data center, such solutions require redirecting clients to sub-optimal data centers from time to time [4, 11, 18]. Although only a small number of clients will be selected to probe remote data centers, these "unfortunate" clients could suffer significant performance degradation. Because even a simple response to a web search query can take 4-6 TCP rounds trips, the inflated RTT to a remote data center can significantly degrade the user's perceived performance. Furthermore, for large responses, such as online maps or documents, directing clients to remote data centers could further inflate the response time. In an era where half a second latency kills user satisfaction [10], such degradation can become unacceptable.

Moreover, in order to minimize the impact of suboptimal redirection to clients arriving subsequently, a small (or even 0) TTL should be set in the DNS response for the initial client. Unfortunately, as Pang et al. [2] discovered in a large-scale DNS study, a significant fraction of clients and LDNSes do *not* adhere to DNS TTLs. Responses could be used long after their expiration, even in excess of 2 hours. In those cases, suboptimal redirection can degrade the performance of a large number of subsequent clients.

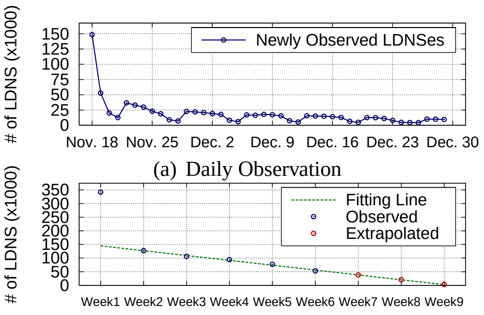
2.5 Other Factors

Most end-hosts are not far away from their LDNSes. Our latest evaluation of more than 1.6 million end-hosts accessing a popular Microsoft online service shows that the geographic distance between end-host and LDNS is less than 27 km in 60% of the cases and less than 428 km in 80% of the cases [6]. Still, there are nontrivial amount of end-hosts using LDNSes not in nearby locations, as similarly reported in earlier studies [12, 17]. The client-LDNS *mismatching* problem is being addressed independently [6]. In this paper, we focus on how to achieve good performance for those end-hosts that co-locate with their LDNSes.

Besides performance, there are many additional important factors to a GTM solution. The dynamic load on the data centers is one such factor: clients should not be directed to over-loaded data centers. ISP delivery cost is another factor. The data centers can use different ISPs, which may have different cost structures, due to complicated contractual relationships between ISPs and data center operators [8]. Taking into account delivery cost could bring significant savings to service providers, operational cost of data centers could also be explored. For instance, the power costs of the data centers can be explored so as to achieve additional savings [16]. Nevertheless, all these cost concerns are secondary and they can only be explored when they do not lead to performance degradation.

3 Limitation of Active Measurements

In this section, we set out to answer the following questions by examining a large LDNS population: how many LDNSes can be reached by active probing?



(a) Daily Observation

(b) Weekly Observation

Figure 1: Newly Observed LDNSes

3.1 A Large LDNS Population

Network Connectivity Status Indicator (NCSI) is a service running on Windows Vista or Windows 7 machines to detect the status of Internet connectivity. For instance, it shows as a system tray icon to notify users upon loss of Internet connectivity. Part of the NCSI service performs DNS queries for a special host name – `www.msftncsi.com`.

Between Nov. 18^{th} and Dec. 30^{th} 2009, we have sniffed 5% of the DNS traffic on the authoritative server of `msftncsi.com` for 6 weeks. A large collection of LDNS addresses is obtained. In particular, the NCSI collection contains about 795,000 LDNS addresses, located in 10,012 cities over 229 countries[1]. Figure 1(a) plots the number of uniquely newly-observed LDNSes every day. It is clear that a large number of LDNSes are observed in the first few days. However, new LDNSes keep being discovered over the entire course. A weekly pattern is also observed where the troughs correlate nicely with weekends.

To estimate the total LDNS population, Figure 1(b) plots the number of uniquely observed LDNSes every week. Except for the first week, there appears to be a clear linear trend. After simply curve fitting and extrapolation, we estimate the total number of LDNS to be around 862,000. Given the wide deployment of Windows machines, we expect this counts for a significant portion of the entire LDNS population on the Internet.

3.2 Reachability of Active Probing

We can use active probing to measure the performance between a LDNS and a data center. In this section, we study how many LDNSes can be reached via active probes.

To this end, we randomly select 50,000 LDNS addresses from the NCSI collection. Our evaluation shows that 24,660 LDNSes respond to Ping – about 49%. For

[1] from Akamai's GeoLocation database.

the rest, since they are DNS servers in nature, we issue DNS queries against them as a measure of active probing. Latency can be obtained by simply calculating the time difference between issuing a request and receiving the response. We experimented with three types of queries: 1) resolving DOT (the root DNS name); 2) reversely resolving localhost (i.e., 127.0.0.1); and 3) reversely resolving the LDNS' own IP address. Unfortunately, only 2896 (about 6% of the total) LDNSes respond to our DNS probes. Thus far, it is clear that a large percentage (about 45%) of the LDNSes are *closed* – they do *not* respond to either Ping or DNS queries from random clients.

To address the insufficiency of active measurements, in the next section, we proposed a much more involved passive DNS reflection method, which works for all LDNSes.

4 The DNS Reflection Method

4.1 The Key Idea

DNS reflection uses DNS traffic itself to measure the performance between a LDNS and a target data center. Very occasionally, the GTM DNS queries are redirected to the target data center. Since DNS traffic is UDP-based, getting one DNS query from the LDNS does *not* allow the target to infer the performance. Therefore, the target *reflects* the DNS query and responds in such a way that the LDNS is triggered to immediately issue another DNS query against the target. By examining the time difference between the two queries occurred on the target, the performance from the LDNS can be readily inferred.

DNS reflection is a passive measurement method and in this sense, similar to the approaches taken by [4, 11, 18], which redirect HTTP traffic from clients to the target. For all these methods, when there is no traffic from the clients or the LDNSes, there is no redirection and thus no measurement.

Beyond this similarity, however, the DNS reflection method differs fundamentally from [4, 11, 18] in a number of important ways: 1) DNS reflection only redirects DNS traffic, not HTTP traffic. Hence, the clients will always be served by the "best" data center (per the choice of the GTM). Although there is a latency incurred when a LDNS is elected to probe a remote date center, there is no latency inflation for subsequent HTTP transactions. 2) Each DNS reflection incurs two round trips between the LDNS and the target. This is a much smaller penalty, compared to redirecting HTTP transactions where the response time will be 4-6 times (or even more) of the inflated round trip time. 3) In HTTP traffic redirection, due to LDNS caching, subsequent arriving clients can be affected. While in DNS reflection, since the final DNS resolution result is *not* modified, it will *not* affect subsequent clients, which will always be served by the "best" data center.

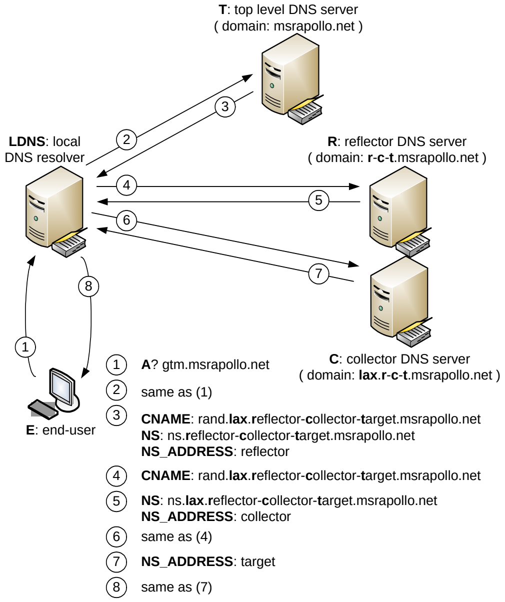

T: top level DNS server
(domain: msrapollo.net)
LDNS: local
DNS resolver
R: reflector DNS server
(domain: **r**-**c**-**t**.msrapollo.net)
C: collector DNS server
(domain: **lax**.**r**-**c**-**t**.msrapollo.net)
E: end-user
1
2
3
4
5
6
7
8
1 **A**? gtm.msrapollo.net
2 same as (1)
3 **CNAME**: rand.**lax**.**r**eflector-**c**ollector-**t**arget.msrapollo.net
NS: ns.**r**eflector-**c**ollector-**t**arget.msrapollo.net
NS_ADDRESS: reflector
4 **CNAME**: rand.**lax**.**r**eflector-**c**ollector-**t**arget.msrapollo.net
5 **NS**: ns.**lax**.**r**eflector-**c**ollector-**t**arget.msrapollo.net
NS_ADDRESS: collector
6 same as (4)
7 **NS_ADDRESS**: target
8 same as (7)

Figure 2: The DNS Reflection Method

4.2 Detailed Process

Figure 2 illustrates the details of each step of the passive DNS reflection method, as elaborated in the following:

Step 1 and 2: An end-user submits a DNS query for `gtm.msrapollo.net` to its LDNS, which then forwards the query to the top level authoritative name server of `msrapollo.net`.

Step 3: Instead of responding with a target IP address, the top level domain server decides to delegate the DNS resolution to a sub-domain, whose server locates in a target data center (e.g., the one in LAX). To this end, it constructs a CNAME (an alias in DNS parlance, which itself has to be recursively resolved by DNS), which embeds LAX as the target, as well as the IP addresses of two DNS servers in LAX, denoted as *reflector* and *collector*, respectively. In addition, it delegates the CNAME to be handled by a sub-domain server, by appending the sub-domain server name and its IP address (the address of the reflector) in the DNS response.[2]

Step 4: The LDNS follows the delegation by the top level authoritative name server and sends the query of the CNAME to the reflector.

Step 5: The reflector further delegates the DNS resolution to another sub-domain, which is a sub-domain of the previously delegated sub-domain. Similarly, the reflector appends the new sub-domain server name and its IP address (the address of the collector) in the DNS response.

[2] CNAMEs and name server records should always be unique such that the entire reflection process avoids caching at LDNS completely to ensure deterministic estimation.

Step 6: The LDNS continues to follow the delegation by the reflector and send the query of the CNAME to the collector.

Step 7 and 8: The collector examines the CNAME and responds with the address of target, which is embedded in the CNAME. The DNS resolution completes.

When the reflector and the collector are in the same physical location (LAX here), we can simply calculate the network latency between the LDNS and the LAX data center from the time difference between the reflector and the collector receiving request (4) and (6), respectively. The process can be further simplified by configuring one physical machine in LAX to own both IP addresses of the reflector and the collector.

Note that all the information regarding how to respond to a particular DNS query is embedded in the query itself. Therefore, neither the top level authoritative name server, nor the reflector or the collector, needs to maintain status at any step during the reflection process. This is an important design to simplify system implementation.

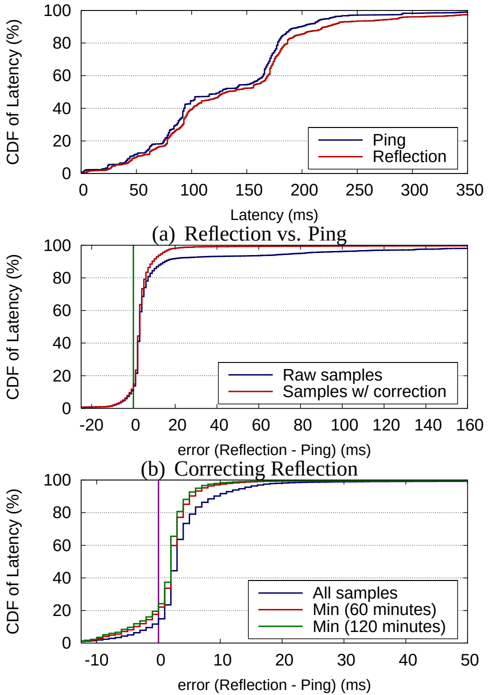
5 Evaluation

We implement a prototype system using C#, which consists of two types of DNS servers. The first type is a top level authoritative name server that responds to a GTM probing query (such as `gtm.msrapollo.net`) with a CNAME, following **step 3** in the previous section. The second type combines the reflector and the collector together and responds to queries targeting at either. It is deployed on a single physical machine configured with two IP addresses (one for the reflector and the other for the collector), in each of the 17 geographically distributed locations in the Microsoft global data center network (3 in Asia, 6 in Europe, 7 in US and 1 in Australia).

5.1 How Accurate is DNS Reflection?

In this section, we first evaluate whether the DNS reflection method gives correct latency measurement. After all, if a LDNS does *not* behave as we understand it would, or if it does *not* immediately send a second query after receiving the delegation response from a reflector, the reflection method could result in inflated or even completely wrong estimates.

To evaluate the correctness and accuracy of DNS reflection, we use 274 PlanetLab nodes as clients to issue DNS queries to our system. Every 15 minutes, each client generates 17 queries, which are redirected to all the 17 reflection locations, respectively. Upon receiving a DNS query, the reflector/collector also probes whether the requesting LDNS responds to Ping, and if it does, 6 Ping probes are sent to the LDNS. The experiment lasted 4 days during the first week of Jan., 2010. Among the 274 PlanetLab nodes, 240 of them are in the same location as their LDNSes (from Akamai's GeoLocation database). Among those co-located LDNSes, 162 of them respond to Ping. For comparison purpose, in the rest of the section, we focus on these 162 LDNSes.



(a) Reflection vs. Ping

(b) Correcting Reflection

(c) Applying Min Filter

Figure 3: Latency Comparison

For each reflection measurement, we compute the latency as outlined in the previous section. Also, we use the minimum of the 6 Ping probes as the ground-truth RTT. To compare the DNS reflection method and Ping, it is sufficient to use all the measurements collected from any single data center. Figure 3(a) shows the cumulative distributions of the two methods from one selected date center.

At the first sight, it appears that the two CDFs match each other quite well. However, if we calculate the difference between corresponding measurement samples and plot the distribution, as shown by the "Raw Samples" curve in Figure 3(b), it becomes clear that the latency measured by reflection and Ping do *not* really match well – the difference is 80 ms at the 95 percentile.

Manual examination of the samples reveals that whenever there is a large gap between reflection and Ping, the reflection latency is always twice as that of Ping. This triggered us to examine the logs of the top level authoritative name server. Finally, we discovered that some LDNSes do *not* use the delegated name server address returned by the reflector in **step 5**. Instead, it always resolves the name server address from the top level authoritative DNS server. This involves an extra round trip between the LDNS and the top level DNS server. In this particular case, the top level DNS server happens to be in the same data center, which is why the reflection latency is twice as that of Ping. Among the 162 LDNSes, there are 27 behaving this way. After we correct the samples from these LDNSes by halving the latency values, the curve "Samples w/ correction" in Figure 3(b) shows that the different with Ping is extremely small – 14 ms at the 95 percentile.[3]

The difference between reflection and Ping is even smaller if we apply an minimum filter on the measurement samples. As shown in Figure 3(c), if we take the minimum value of all the samples in a 2-hour window, then the difference with Ping is only 6 ms at the 95 percentile. Therefore, we conclude that DNS reflection is an extremely accurate measurement method.

5.2 How Good is a Reflection-based GTM?

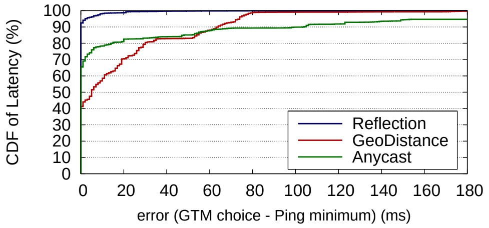


Figure 4: GTM Policy Comparison

Next, we evaluate the effectiveness of a reflection-based GTM by comparing it with geography-based GTM and IP anycast-based GTM. For the geography-based GTM, we use Akamai's GeoLocation database to find the latitude and longitude of each LDNS. The data center with the shortest great circle distance from the LDNS is selected as the best choice. For the IP anycast-based GTM, we setup an IP anycast address, which is announced from all the 17 locations. There are 17 DNS servers listening on the anycast address (one in each location). The PlanetLab nodes send DNS queries towards the anycast address. These queries are naturally routed to the anycast-closest data center. For the reflection-based GTM, we use the reflection measurements collected in every 2 hours to rank all data centers with respect to each LDNS. The minimum latency one is chosen as the best choice for the next 2 hours.

For each GTM, because of the co-location of the LDNS and its corresponding PlanetLab node, we use the Ping latency between the LDNS and the GTM-choice as the latency between the node and the data center, We use the minimum Ping latency of the 17 RTTs to all the data centers as the optimal latency. We calculate the difference between each GTM and the optimal. The cumulative distributions of the three GTMs are shown in Figure 4. We observe that the reflection-based GTM is 2 ms within the optimal (at the the 95 percentile), while the geography-based GTM is 74 ms and the anycast-based GTM is 183 ms. In other words, for the users whose performance is most precarious, the benefit of reflection-based GTM is significant!

6 Conclusions

In this paper, we argue that existing GTM solutions can perform poorly for a non-negligible fraction of the users. We propose a novel DNS reflection method, which uses the GTM DNS traffic itself to measure the performance between an arbitrary end-user and a data centers, with extremely good accuracy. We show that reflection-based GTM is very close to optimal and can significantly benefit a non-negligible fraction of the users.

References

[1] 3-DNS reference guide. WhitePaper, F5 Networks, Inc., 2002.

[2] Aditya, J. et al. On the Responsiveness of DNS-based Network Control. In *Proc. of IMC* (2004).

[3] Agarwal, S. et al. Match-Making for Online Games and Other Latency-Sensitive P2P Systems. In *Proc. of SIGCOMM* (2009).

[4] Andrews, M. et al. Clustering and Server Selection Using Passive Monitoring. In *Proc. of INFOCOM* (2002).

[5] Ballani et al. A Measurement-based Deployment Proposal for IP Anycast. In *Proc. of IMC* (2006).

[6] Huang, C. et al. Public DNS Systems and Global Traffic Management. *Work in progress* (2010).

[7] Gwertzman, J. et al. The Case for Geographical Push-Caching. In *Proc. of HotOS* (1995).

[8] Huang, C. et al. Measuring and Evaluating Large-Scale CDNs. In *Microsoft Research Technical Report MSR-TR-2008-106* (2008).

[9] Karger, D. et al. Web Caching with Consistent Hashing. In *Proc. of WWW* (1999).

[10] Kohavi, R. et al. Practical Guide to Controlled Experiments on the Web: Listen to Your Customers not to the Hippo. In *Proc. of KDD* (2007).

[11] Krishnan, R. et al. Moving Beyond End-to-End Path Information to Optimize CDN Performance. In *Proc. of IMC* (2009).

[12] Mao, Z. M. et al. A Precise and Efficient Evaluation of the Proximity between Web Clients and their Local DNS Servers. In *Proc. of USENIX ATC* (2002).

[13] Oberheide, J. et al. Characterizing Dark DNS Behavior. In *Proc. of DIMVA* (2007).

[14] Partridge, C. et al. RFC1546: Host Anycasting Service, 1993.

[15] Pathak, A. et al. Measuring and Evaluating TCP Splitting for Cloud Services. In *Proc. of PAM* (2010).

[16] Qureshi, A. et al. Cutting the Electric Bill for Internet-Scale Systems. In *Proc. of ACM SIGCOMM* (2009).

[17] Shaikh, A. et al. On the Effectiveness of DNS-based Server Selection. In *Proc. of INFOCOM* (2001).

[18] Stemm, M. et al. A Network Measurement Architecture for Adaptive Applications. In *Proc. of INFOCOM* (2000).

[3] We use DNS fingerprinting tool `fpdns` to find out the DNS server software and versions, but the extra name server resolution appears to happen independently. At the moment, we can only conjecture that it is likely due to specific configurations.

Fortunately, the extra resolution incurs fixed latency and thus will *not* affect the relative performance ranking with respect to all the locations. In addition, the latency can be reduced by deploying the top level authoritative DNS server in every data center and on IP anycast.