Content Delivery and the Natural Evolution of DNS

Remote DNS Trends, Performance Issues and Alternative Solutions

John S. Otto Mario A. Sánchez John P. Rula Fabián E. Bustamante
Northwestern University
{jotto,msanchez,john.rula,fabianb}@eecs.northwestern.edu

ABSTRACT

Content Delivery Networks (CDNs) rely on the Domain Name System (DNS) for replica server selection. DNS-based server selection builds on the assumption that, in the absence of information about the client's actual network location, the location of a client's DNS resolver provides a good approximation. The recent growth of remote DNS services breaks this assumption and can negatively impact client's web performance.

In this paper, we assess the end-to-end impact of using remote DNS services on CDN performance and present the first evaluation of an industry-proposed solution to the problem. We find that remote DNS usage can indeed significantly impact client's web performance and that the proposed solution, if available, can effectively address the problem for most clients. Considering the performance cost of remote DNS usage and the limited adoption base of the industry-proposed solution, we present and evaluate an alternative approach, *Direct Resolution*, to readily obtain comparable performance improvements without requiring CDN or DNS participation.

Categories and Subject Descriptors

C.2.4 [**Communication Networks**]: Distributed Systems—*Distributed applications*; C.2.5 [**Communication Networks**]: Local and Wide-Area Networks—*Internet*; C.4 [**Performance of Systems**]: Measurement techniques

General Terms

Experimentation, Measurement, Performance

Keywords

CDN, Content distribution, DNS, DNS extension, Internet, measurement

IMC'12, November 14–16, 2012, Boston, Massachusetts, USA.

1. INTRODUCTION

Content Delivery Networks (CDNs) replicate content across geographically distributed sets of servers and redirect clients to nearby replicas to reduce web access time. Since first emerging in the mid 1990s, CDNs have become the primary vehicle for distributing content over the Internet. Today, 74% of the top 1,000 sites and 89% of their pageviews[1] use CDNs to deliver content to clients (Fig. 2). Over the last decade the CDN industry has grown to include a large and diverse set of over 49 companies including regional focused ones (e.g. Accelia) and global services such as Akamai and Level3.[2]

CDNs rely on the Domain Name System (DNS) for both dynamic request routing and replica server selection. For server selection, CDNs base their decision on the IP address of a client's local DNS resolver. This approach builds on the assumption that, in the absence of information about the client's actual network location, the location of its resolver provides a good approximation. The recent growth in remote DNS usage challenges this assumption.

This paper investigates the impact of recent DNS evolution on the web experience of end users. Over the past few years there has been a significant increase in the use of remote DNS services, including public DNS. Remote DNS offers a number of potential advantages to both service providers and end users. For service providers, strategically placed, remote DNS server clusters offer economies of scale and simplified management, among other benefits. For users, such services can yield better DNS performance, availability and security. The relative importance of such benefits to end users partially explains the observed growth in public DNS usage. OpenDNS has recently reported a 2× increase in users of their service over 2010-2012 [20, 21]. In our own survey (Fig. 3) based on data from the EdgeScope project [5], we found that the public DNS user base has grown by 27% annually over the last 21 months. As of December 2011, 8.6% of users in this dataset relied on a public DNS service.

While remote DNS services could indeed result in better DNS performance for end users, by breaking the assumption made by CDN mapping approaches they may yield worse end-to-end web performance. To illustrate the scope and extent of this effect, we compare the *minimum* time to fetch content from Akamai's CDN when using local and remote DNS servers from different locations in North America. Fig-

[1] As ranked by Alexa.com, accounting for 48% of all Internet pageviews.
[2] `http://cdnlist.com`; July 2011

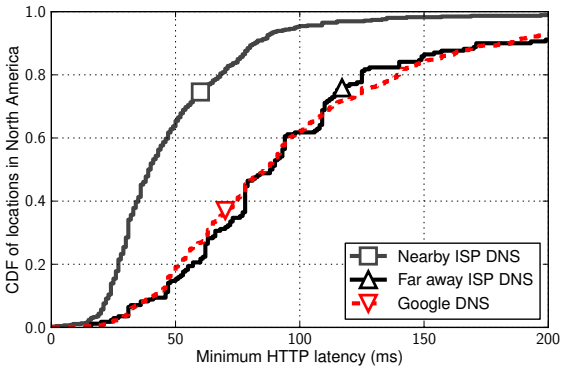


Figure 1: For North America, minimum HTTP latency to get Akamai content using near or far ISP DNS, and Google DNS. 27% of locations have "far away" ISP DNS (latency >50 ms). Median case's access latency is *doubled* for far-away and Google vs. nearby ISP DNS.

ure 1 plots the minimum latency for the different locations using *nearby* (latency <50 ms) and *faraway* ISP DNS and Google DNS.[3] Web access latency is a significant factor in user satisfaction [7, 26]. As the figure shows, remote DNS—both public DNS and far-away ISP DNS—has a major impact on CDN access latency. Latency doubles in the median case and *triples* for the worst 10% of locations. These results extend and are consistent with those in Ager et al. [1]; we find that a large fraction of ISPs' DNS services, as much as 27%, are distant from their users and therefore result in significant reductions in CDN performance.

In this paper, we present results from a large-scale study of the end-to-end impact of remote DNS usage on CDN performance (§ 4). Based on experiments conducted from 10,923 end-hosts across 99 countries, we show that for 90% of the sampled locations, the set of CDN replicas selected based on public DNS services has *no overlap* with those selected based on the client location. This error results in a 60% increase in end-to-end latency in the median case and up to 3× increase for 20% of the sampled location.

The potential negative impact of remote DNS usage on web performance motivates the recently proposed edns-client-subnet EDNS0 extension [3]. This extension, put forward by a collaboration of several CDNs and DNS service providers [28], allows DNS recursive resolvers to pass along clients' subnet information that could be used by CDNs to improve replica server selection. A key challenge to this approach is adoption, as it requires the commitment of both CDN and DNS services to be effective. We report on the first study of the adoption of the proposed edns-client-subnet extension and its potential benefits for public DNS service users (§4.3). We find that the proposed extension could be effective at reducing the impact of public DNS usage on CDN performance. Our results show that, assuming wide adoption, the proposed extension can reduce the impact of public DNS usage to less than 80% for 80% of locations (30% for the median case). However, in a survey of the top 1,000 most popular sites, we find that only 9% of sites use CDNs that support the extension.

We introduce *Direct Resolution* (*DR*) an alternative solution (§5) that obtains comparable performance improvements to the DNS extension. Our end-host solution enables incremental adoption by affected users, and does not rely on support from either DNS services or CDNs. The approach leverages the cache of an end host's recursive resolver to efficiently map a Canonical Name (CNAME) to an authoritative name server, but directly contacts the authoritative server to obtain a precise redirection for the client. We have implemented *DR* as part of *namehelp*, a tool based on a popular DNS benchmark utility [18] to both provide a comparative evaluation of DNS service and web performance and act as a DNS proxy that implements *DR* to improve CDN mappings. Naïvely using *DR* for all CDN queries improves end-to-end performance for 49% of locations. *namehelp* avoids this penalty and achieves strictly better performance than recursive DNS alone by only using *DR* when it has previously improved end-to-end performance.

Contributions

Recent studies [1, 9, 12] have shown that the use of public DNS services can complicate the mapping of clients to replica servers or data centers done by CDNs and cloud service providers, while Huang et al. [8] and Khosla et al. [12] discuss possible approaches to mitigate the performance penalty. Our work extends significantly these previous efforts, contributing:

- The first study of the end-to-end web performance impact of remote DNS on CDNs from users' perspectives in access networks.
- The first evaluation of the proposed edns-client-subnet extension, its potential performance and level of adoption.
- The design and experimental evaluation of a novel end-system solution that provides comparable benefits to the extension and is readily available for users to install.

After providing some background, we expand on each of these contributions. We close the paper with a discussion of closely related work in §6 and our concluding thoughts in §7.

2. BACKGROUND

Since their emergence in the mid 1990s, CDNs have become the primary vehicle for delivering content over the Internet. To estimate the reliance of popular sites on CDNs, we conducted a survey of the top 1,000 most popular sites, as ranked by Alexa.com. For each site, we download its index page and linked web objects. We used several techniques to determine if a site uses CDNs, including detection of HTTP redirection to CDN domains or use of CNAME entries in DNS resolutions. Since there are other methods of redirecting to a CDN (e.g. domain delegation) that we do not capture, these results are a lower bound on the use of CDNs in popular sites.

[3]To put this in context, the average network latency between U. Washington in Seattle, WA and Georgetown U. in Washington, D.C. is ≈ 83 ms.

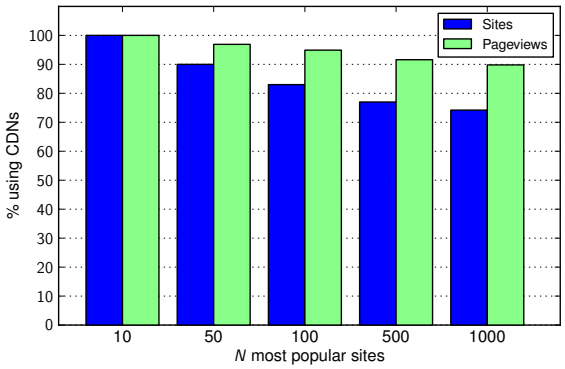


Figure 2: Fraction of the top N most popular sites and their pageviews that use CDNs. Over 70% of the top 1,000 sites and 89% of their pageviews utilize CDNs.

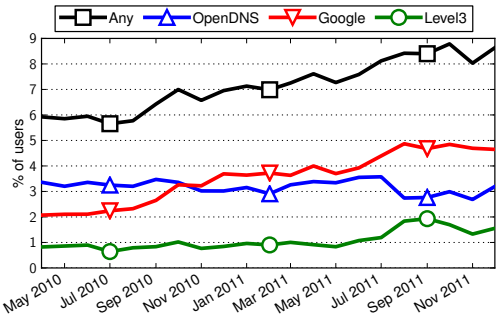


Figure 3: % of users with public DNS configured on their computer over the last 21 months. "Any" shows the combined adoption percentage of these services. Public DNS usage is growing by 27% annually.

Figure 2 summarizes our findings. The figure plots both the fraction of "Sites" (blue/dark) and "Pageviews" (green/light) using CDNs in the top N most popular sites. As the figure shows, all sites in the top 10 and over 70% of those in the top 1,000 rely on CDNs. Weighting CDN use by pageviews results in an even larger fraction of top sites using CDNs. Across all views to the top 1,000 sites—accounting for 48% of all pageviews—9 out of 10 require loading objects from a CDN.

In general, for a web client to retrieve content for a web page, the first step is to use DNS to resolve the server-name portion of the content's URL into the address of a machine hosting it. If the web site uses a CDN, the content will be replicated at several hosts across the Internet. A popular way to direct clients to those replicas dynamically is DNS redirection. With DNS redirection, a client's DNS request is redirected to an authoritative DNS name server that is controlled by the CDN, which then resolves the CDN server name to the IP address of one or more replica servers [13]. DNS redirection can be used to deliver full or partial site content. With the former, all DNS requests for the origin server are redirected to the CDN. With partial site content delivery, the origin site modifies certain embedded URLs so that requests for only those URLs are redirected to the CDN. CDNs typically select a (set of) replica to serve the request from based on the network and geographic region of the local DNS server. While a reasonable approximation when clients and their local DNS resolvers are topologically close, the use of remote DNS services can yield less than optimal redirections.

Over the past few years there has been a significant increase in the use of remote DNS services,[4] including public DNS. To analyze longitudinal trends in remote DNS usage, we use 21 months of user DNS configuration data from the EdgeScope project (April 2010 through December 2011). EdgeScope [5] collects data from BitTorrent [32] users, including host configuration and host and network measurement statistics. The dataset includes information reported by 47,119 users located in 197 countries and 4,613 ASes, giving us a diverse and global perspective on DNS usage by BitTorrent users.

We quantify public DNS adoption by counting the users having configured well-known IP addresses for these services at different points over the observed period. Figure 3 shows the observed trends in combined (labeled *Any*) and individual public DNS service adoption for the three most popular: Google, OpenDNS and Level3. Overall, we found a 27% annual growth rate in public DNS adoption, reaching 8.6% of sampled users in December 2011. Google's DNS service showed the most significant growth—a 74% annual increase—which resulted in it becoming the most-used public DNS service (overtaking OpenDNS) in November 2010. For 70% of these public DNS users, the public service is configured as their primary (22%) or only option (48%).

While subject to a potential "geek bias", our results are indicative of an overall increasing trend of public DNS use.[5] Indeed, our observed growth trends are consistent with those reported by OpenDNS showing a 2× increase in users of their service over 2010-2012 [20, 21].

2.1 Industry response

The increased use of remote DNS services, with its potential impact on web performance, has motivated a recent response from industry. Several companies, including Google, OpenDNS and EdgeCast, have proposed the "edns-client-subnet" DNS extension ("ECS") to the IETF [3] as part of the "Global Internet Speedup" initiative [28].

The proposed extension provides a mechanism for recursive DNS resolvers to pass client location information to CDN authoritative DNS servers. This enables CDNs to factor in the client's actual location in redirection decisions. The extension is specified as a EDNS0 [30] option that is appended to the DNS query and contains the client's network prefix. The length of the prefix is a parameter determined by the recursive resolver.

Assuming that *both the DNS service and the CDN* support the extension, the approach enables CDNs to transparently improve the quality of redirections. It is straightforward to determine whether a DNS service or CDN supports the

[4]While anecdotal, evidence of this trend can be seen in reports of DNS service outages in national ISPs, typically affecting large geographic regions [34, 35].

[5]`http://voices.washingtonpost.com/fasterforward/2010/11/comcast\_internet\_or\_any\_other.html`

extension by sending a request to recursive or authoritative DNS servers. If the `ECS` option is present in the response, the extension is supported.

3. METHODOLOGY

In this section, we describe our methodology to experimentally evaluate the interactions between remote DNS and CDNs.

We base our analyses on data contributed by 10,923 end hosts distributed across 99 countries and 752 ASes, providing a diverse set of vantage points. In terms of geography, 59% of the hosts are spread across 35 European countries, 21.9% are from the United States, 5.5% are located in Asia, and 3.9% are in Oceania. Our dataset was collected over a 127 day period between September 12th, 2011 and January 16th, 2012.

Each end host runs an instance of an ISP characterization plugin for the Vuze BitTorrent client [32]; users download our software and allow us to collect measurement results. Each host reports: local configuration information, the results of DNS resolutions, HTTP request timing statistics, and results of traceroutes and pings to CDN and DNS servers.

For our study, we chose different public DNS services and CDNs based on popularity and deployment architectures. The set of public DNS services we measured includes two of the most popular: Google Public DNS and OpenDNS. Our set of CDNs, which includes Akamai, EdgeCast, Google and Limelight, covers a variety of deployment models, ranging from servers sparsely deployed at points of presence to servers located inside the networks of end users. For each CDN, we manually selected a small (<10 KB) web object hosted by that CDN. This enabled us to evaluate the end-to-end performance using 2 steps: (1) clients query DNS for the object's hostname to obtain a CDN redirection, and then (2) request the object via HTTP from the CDN's edge server.

We define each /24 IP prefix from which we have measurements to be a vantage point "location" and aggregate measurements taken by nodes in the prefix. In the remainder of the paper, we analyze the distributions across these "locations". To ensure the significance of our results, we require at least 3 measurements from each location. For each combination of vantage point location, DNS service and CDN, we select the *best case results* (i.e. minimum latencies) for comparison.

Our experiments are designed to minimize their impact on the monitored services. We use caching at each node to avoid multiple probes to the same server, and randomly schedule experiments to reduce the likelihood of concurrent measurements launched from different hosts.

3.1 Obtaining CDN redirections

We use iterative DNS resolution to obtain CDN redirections that are not affected by the location of the ISP or public DNS servers.

End hosts typically resolve DNS names by querying a recursive DNS resolver operated by an ISP or public service. The recursive resolver makes several queries on behalf of the end host, caches the responses, and returns the answer to the end host.

Using iterative resolution, an end host can act as its own recursive resolver. Since the host directly contacts the authoritative DNS servers, CDNs can base their replica server selections on the client's *actual network location*.

To obtain CDN redirections that are effectively informed by client location, we follow two approaches to cover the cases where the edns-client-subnet DNS extension is *and* is not supported.

DNS extension supported. To query combinations of DNS services and CDNs where the extension *is supported*, we implement a stub resolver that adds the `ECS` option with a specified IP prefix to a query and forwards the request to a recursive resolver. This method of querying with the `ECS` option is possible due to a provision in the specification to support hierarchies of DNS resolvers; if the `ECS` option is already present in a query, it should not be modified by a recursive resolver. Since our stub resolver generates the option, we can evaluate the impact of providing different amounts of client information (i.e. prefix length) on CDN redirections.

DNS extension not supported. To evaluate the DNS extension's approach for DNS services or CDNs that *do not* support the extension, we emulate its characteristics: the latency to receive the answer, and the CDN's answer based on the client's location. We determine the latency to receive an answer by simply querying the DNS service for a name that is cached on the server. To obtain a CDN answer based on the client's location, we use iterative resolution to directly query the CDN's authoritative DNS server. As we show in §4.2, redirections via iterative lookup generally provide best-case HTTP performance. As a result, our emulated `ECS` query gives redirections equivalent to using the extension and providing the client's full IP address.

3.2 Measuring DNS services

Public DNS services are typically offered on BGP anycast-enabled IP addresses, allowing all users to configure the same server IP address. Probing public DNS services can present a challenge for network-level measurements, as some anycast-enabled addresses do not respond to ICMP pings.

To determine the recursive DNS resolver's globally routable *unicast IP address* so that we can probe it directly, we use a technique similar to that used by Huang et al. [9]. We operate an authoritative DNS server that answers queries with the IP address of the recursive DNS resolver sending the query. This approach works because resolvers that answer queries on an anycast IP interface use a *separate unicast interface* to communicate with other authoritative DNS servers; our service returns this unicast IP address for the recursive resolver. Clients send a DNS query to the public DNS service's anycast IP address and receive its globally routable unicast IP address in the DNS response; this gives the client a mapping between the public DNS service and the unicast IP address for the recursive resolver. Clients report these mappings, which enables us to identify public resolvers in our dataset.

Filtering configured public DNS services. To obtain ISPs' DNS servers in a scalable manner, we start out with users' DNS configurations and exclude known public DNS services. Since users may have public DNS services configured on either their computer or a DNS-proxying middlebox, we must filter out both cases. First, we conservatively filter well-known public DNS anycast IP addresses (e.g. 8.8.8.8 for Google DNS) configured on the user's computer. Then, using the result of our previously

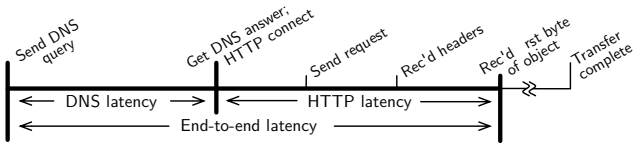


Figure 4: Timeline of DNS and CDN interactions, and the components we consider when measuring latency in this work.

described approach to determine a DNS server's *unicast IP address*, we can see through any proxies (e.g. middlebox) and determine the underlying DNS server. We exclude data when the underlying DNS server is in a /24 prefix where we have previously located public DNS servers via directly probing the public DNS services' well-known addresses.

We use ping latency to measure network distance. To reduce the impact of transient spikes in latency, we probe each DNS service's anycast and unicast addresses with three ICMP pings and select the minimum latency.

3.3 Measuring CDNs

We measure end-to-end latency for each pair of DNS and CDN services using the combined latency of DNS lookup and HTTP request (Fig. 4).

- DNS lookup: Time to obtain a CDN redirection, from querying a recursive DNS resolver until receiving an answer
- HTTP request: Time from initiating a connection to a replica server, to receiving the first byte of an object from an CDN edge server via HTTP

For each CDN studied, we select one of their customer's small web objects to request. To factor out any differences due to varying object size across CDNs, we compute access latency based on the time needed to receive *the first byte* of the object. We conduct each DNS and HTTP `GET` request twice in close succession so that the requested object will be served from the server's cache on the second request; we use the smaller latency.

3.4 Baseline for performance comparison

One issue we faced in our analyses was determining how to compare the performance of different approaches for obtaining CDN redirections. We initially considered using the performance when using ISP DNS as our baseline. However, several issues with ISP DNS mean that it is not always an appropriate choice as a baseline. For instance, the use of remote DNS architectures may break the assumption of proximity between client and resolver. Also, some ISP DNS services exhibit high latencies and poor cache performance due to load balancing [1]. Another alternative we considered is using the performance of redirections seen via iterative DNS resolution; however, factors such as CDN load balancing between replica server clusters may affect the redirections resulting in suboptimal performance.

We determine baseline performance by selecting each location's *best* performance from any CDN redirection mechanism, including using ISP DNS, public DNS services and iterative resolution. We compare the performance of CDN redirection approaches' aggregate distributions of performance relative to this baseline.

To compare the end-to-end performance of different approaches to obtaining CDN redirections, we compute *for each location* the baseline DNS latency to obtain a CDN redirection, and the baseline HTTP latency to download an object from the CDN. We define our baseline DNS performance as the minimum latency to obtain a DNS response from *any* ISP or public DNS service. This represents the latency to obtain a cached answer from the nearest recursive DNS resolver. For our baseline HTTP performance, we use the minimum observed HTTP latency for that CDN in that location. This value represents the best-possible CDN performance we observed from the location—typically, the latency to obtain a cached object from the nearest CDN server. Our end-to-end performance baseline is defined as the sum of the DNS and HTTP baseline latencies. This represents an idealized best-case scenario in which the nearest DNS returns a cached CDN redirection, the redirection yields the nearest CDN replica server, and the requested object is cached on the replica server.

4. DNS–CDN INTERACTION

To examine the impact that different DNS services have on several aspects of CDN performance, we first look at differences in the replica servers seen when obtaining CDN redirections using ISP DNS, public DNS services and iterative resolution. Next, we compare the impact of these approaches on the resulting HTTP performance. Finally, we evaluate the end-to-end performance benefit of using the edns-client-subnet extension and the degree of its adoption.

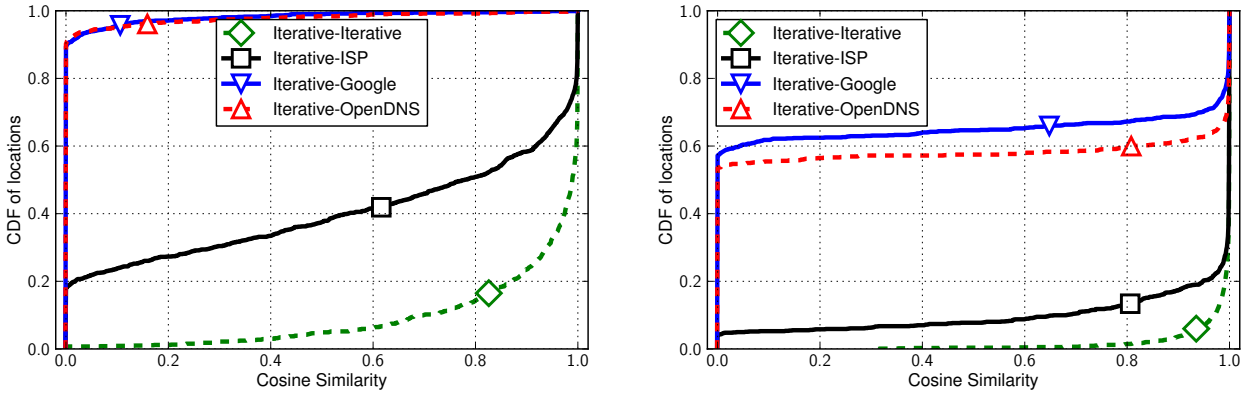
4.1 Redirection similarity

When a client requests a redirection, a CDN may consider factors beyond proximity, including server or network load. As a result of these transient effects, a client may see not one but rather a collection of replica servers. Over time, this collection reveals patterns in which some replica servers are seen more frequently than others. We aggregate redirections based on replica servers' /24 IP prefix since these servers are often deployed in clusters. We build a "ratio map" to represent these aggregated redirections, which maps each replica server cluster to the fraction of redirections specifying that cluster [2, 27].

We analyze these ratio maps of CDN redirections obtained via different DNS services, estimating the degree of similarity between them. Previous work has shown that this similarity can be used to estimate the proximity between nodes [2, 27].

We use *cosine similarity* to quantify the similarity between ratio maps on the assumption that clients with equivalent sets of redirections, i.e. client sees the same replica servers, should yield comparable HTTP performance. Given two ratio maps, we extract two equal-length vectors—one for each ratio map. Cosine similarity between the two vectors A and B quantifies the degree of overlap between the vectors computed as their dot product divided by the product of their lengths. Cosine similarity yields a value in the range $[0, 1]$:

$$cos\_sim = \frac{A \cdot B}{\|A\|\|B\|}$$



(a) Akamai

(b) Limelight

Figure 5: CDFs of cosine similarity for Akamai and Limelight CDN redirections by DNS lookup. Compared to Iterative, ISP DNS has some similarity in at least 80% of locations. However, for 90% of locations there is no similarity in Akamai redirections via public DNS.

When *cos_sim* = 0, the sets of redirections have no clusters in common. Values greater than 0 indicate that some clusters are seen in both sets; *cos_sim* = 1 means that the sets of clusters seen are equivalent. We ensure the significance of this analysis by requiring that each location's ratio map comprises *at least* 3 redirections (see §3). This enables us to capture load balancing behavior when computing the similarity of CDN redirections.

We use cosine similarity to estimate differences in redirections resulting from ISP and public DNS, to those resulting from iterative resolution. Redirections seen via ISP or public DNS are based on the location of the DNS resolver—not the client. Since iterative resolution yields CDN redirections based on client location, any differences indicate potential for redirections to suboptimal replica server clusters. Figure 5 plots the cosine similarities seen in Akamai and Limelight CDN redirections between iterative and ISP or public DNS services.

The "Iterative-Iterative" curves show similarity between random subsets of iterative redirections, and are the upper bound on similarity for a CDN's redirections. If CDN redirections were static, we would expect this value to always equal 1; however, they are dynamic and responsive to several factors including system load and network conditions. Cosine similarity values <1 reflect these variations in redirections. We find that, for the "upper bound" of similarity, we see higher similarity for CDNs with fewer data centers (e.g. Limelight) as there will likely be fewer variations in redirections.

Iterative and ISP redirections have the highest similarity, with 80% of locations having some Akamai replica server clusters in common. In contrast, the "Iterative-Google" and "Iterative-OpenDNS" curves reveal very low similarity—in 90% of locations, there is no similarity in the set of Akamai redirections.

These trends are the same for Limelight redirections, except that public DNS is able to match iterative's redirections in a larger fraction (43%) of locations. While ISP and public DNS resolvers both result in different redirections compared to iterative lookups, the effect is significantly larger for public DNS services.

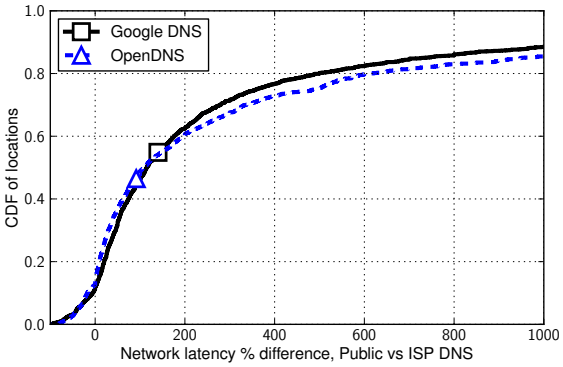
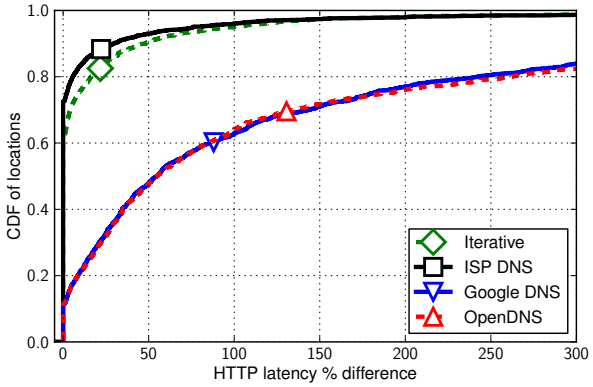


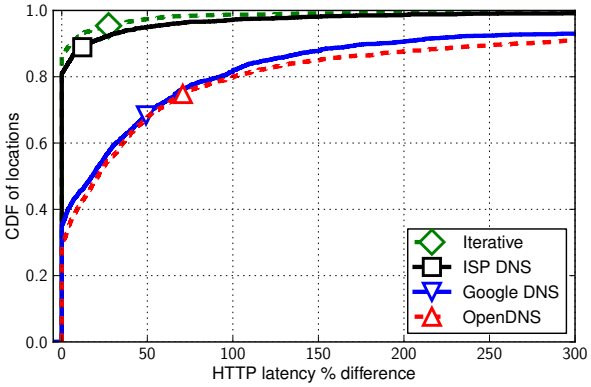
Figure 6: CDF of % difference in network latency to public vs. ISP DNS resolvers. Public DNS servers are farther away in 90% of locations and more than twice as far in 50%.

One possible explanation for the differences in cosine similarity seen with public DNS vs ISP DNS is their distance from the client. Since CDNs' replica server mappings are indicative of the location of the recursive DNS resolver, increased distance to a resolver is expected to correlate with lower similarity in redirections. To test this hypothesis, Fig. 6 plots the percent difference in latency to public DNS resolvers relative to ISP DNS latency. For half of our locations, public DNS servers are at least *twice as far away* as ISP DNS servers. This explains the lower similarity we find in redirections via public DNS.

Our cosine similarity analysis serves as a useful indicator for the *potential* impact of using a given DNS service. Having a cosine similarity value <1 is a *necessary but not sufficient* condition for a DNS service to affect performance. For instance, ISP DNS may provide different redirections compared to Iterative, but both of the CDN mappings could provide equivalent performance. In the next section, we directly compare the quality of HTTP performance using CDN redirections obtained via different DNS services.



(a) Akamai



(b) Limelight

Figure 7: CDF of % difference in HTTP latency using iterative, ISP or public DNS relative to our ideal baseline, for Akamai and Limelight. Replica servers seen via ISP DNS and iterative lookups generally provide the best-case HTTP latency. Public DNS services often yield redirections with higher latencies, with a 2× increase for 35% (Akamai) and 18% (Limelight) of locations.

4.2 HTTP performance

To understand the impact of the differences in CDN redirections between DNS approaches, we analyze the resulting HTTP performance obtained through each. As our metric, we use the latency to establish a connection to a CDN replica server and start downloading an object. This latency is an important measure of CDN performance. When loading web pages, low latencies are essential to attaining low page load times. In the case of high-bandwidth streaming (e.g. HD video), lower latency connections allow for higher transfer rates.

We evaluate the quality of CDN redirections via iterative, ISP and public DNS resolutions relative to our ideal baseline HTTP latency. Figure 7 plots the percent difference in HTTP latency—the time to receive the first byte of an object—of these DNS resolution approaches. Both iterative and ISP DNS redirections match the best-case HTTP performance in 70% (Akamai) and 80% (Limelight) of locations. In contrast, public DNS's redirections only achieve this for 15% (Akamai) and 40% (Limelight) of locations. Since Limelight's CDN architecture has relatively fewer data centers, the increased distance to the public DNS server is less likely to affect the client's replica server mapping. Still, public DNS results in at least double the HTTP latency for 35% (Akamai) and 18% (Limelight) of locations.

For some sampled locations, we find that iterative resolution does not always result in optimal performance. In these cases, several factors beyond client location may affect the redirections obtained from the CDN, including load balancing between server clusters [29]. In practice, this source of noise in CDN redirections is negligible in comparison to the overall results that we report.

4.2.1 *Cause of performance differences*

To better understand the components of HTTP latency, we examine two possible explanations: load and proximity of edge servers. These analyses will determine the reason for the differences in HTTP performance we observe between DNS services.

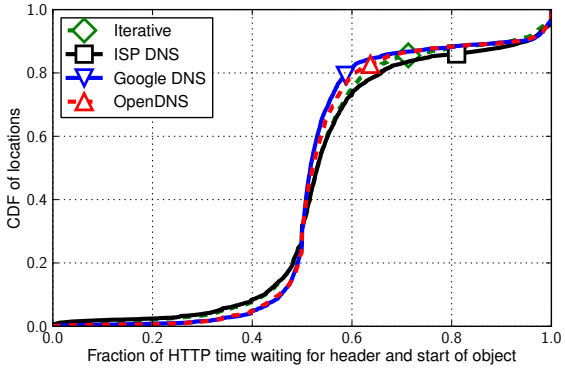


Figure 8: CDF of the fraction of the HTTP interaction spent waiting for a response, comparing redirections via several DNS services. Under load, servers queue requests, increasing this parameter from the expected value of 0.5. The distributions are nearly identical; there are no systematic differences in server load.

We test whether server load (and therefore response time) differ significantly between the edge servers seen via each DNS service. Since keeping edge server load low is one of the main goals of CDNs in general, we expect that our requests for cached objects should be served with minimal delay. We validate this hypothesis in Fig. 8, which plots CDFs of the fraction of the HTTP latency spent waiting to receive the response header and first byte of the requested object, for both lookups via public and ISP DNS. The intuition is that an HTTP request requires 2 round-trips to the server: the first to establish a connection, and the second to request the content. If the server is not heavily loaded, then the request will not be queued, and a request for a cached object can be returned immediately—in this case, the fraction of time

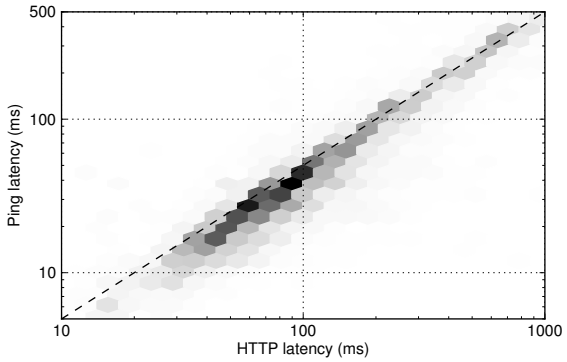


Figure 9: Heat map of HTTP and ping latency to replica servers aggregated across our experiments; darker bins indicate higher density of samples. The dashed black line shows the expected value, where HTTP latency is 2× ping latency. The variables are strongly correlated ($r = 0.88$). Network latency is the dominant factor in HTTP performance.

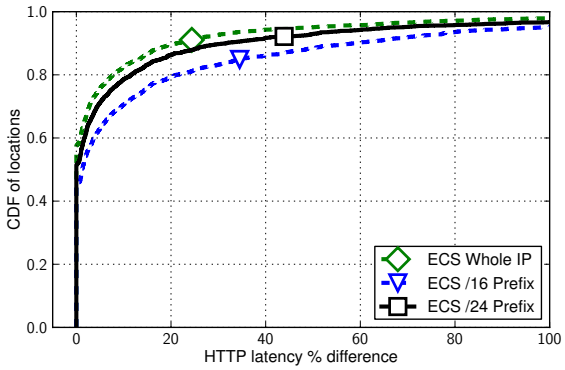


Figure 10: CDFs of HTTP latency relative to our ideal baseline, for varying amounts of client information included in an ECS-enabled lookup. Providing a client's /24 prefix provides nearly the same performance as giving the whole IP address for Google CDN. When only providing the /16 prefix of the client's address, HTTP latency is higher than when giving the /24 prefix for about 30% of locations.

spent on the second round-trip should be the same as the first, or 50% of the total request time. We find that in 70% of cases, this metric is within 10% of the expected value, and find no significant differences between these distributions. This indicates that server load is not systematically different for CDN replica servers seen via any DNS service.

Next, we evaluate network latency's relationship to overall HTTP latency. In an HTTP interaction, requesting and receiving an object requires 2 round trips: the first to establish the connection, and the second to request and receive the object. In Fig. 9 we plot a heat map showing the relationship between HTTP and network latency, with the black line showing the expected relationship in which HTTP latency is twice the network latency. We find a strong correlation between these two variables ($r = 0.88$, $n = 629252$), indicating that network latency has a significant impact on overall HTTP latency.

4.3 Better performance using client location

The previous analysis shows that using remote DNS services results in significantly different CDN redirections, often leading to reduced HTTP performance. The "edns-client-subnet" DNS extension [3] provides a way to expose client location information, part of the client's IP address, to CDNs as an approach to address this.

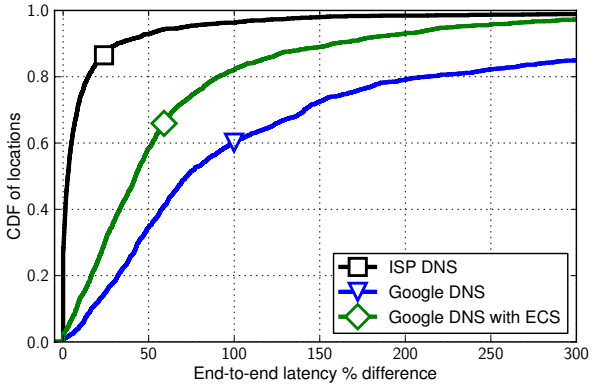
In the following paragraphs we present the first study on the efficacy of the proposed extension. For our evaluation, we follow the methodology for obtaining CDN redirections using the extension in §3.1. We close the section with a survey of the current extent of its adoption.

4.3.1 Performance improvement with client location

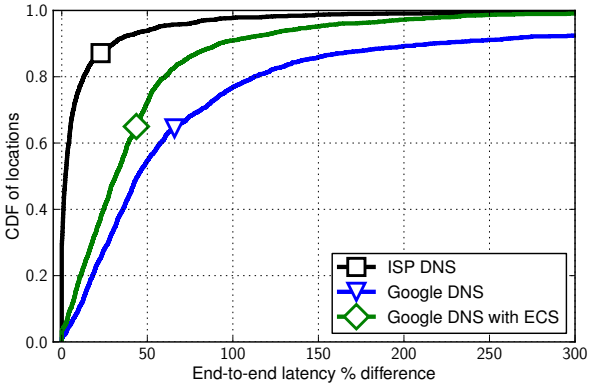
The ECS extension functions by passing part of the client's network address to the CDN's authoritative DNS server to aid in selecting an appropriate replica server. Choosing the amount of client information to provide requires several considerations. Exposing fine-grained client location information could enable the CDN to provide more accurate redirections, but might also result in the recursive resolver having to cache additional ECS-specific mappings—one for each client prefix—for every CDN domain name.

We evaluate the impact of providing different amounts of client location information via the extension on HTTP performance. For this experiment, we query Google DNS servers for a Google CDN name. Figure 10 plots the % difference in HTTP latency of redirections of ECS queries with either the client's /16 prefix, /24 prefix, or the client's whole IP address. For 60% of locations, giving the /16 and /24 prefixes are sufficient to obtain equivalent performance to giving the whole IP address. In the remaining locations, the /24 prefix provides slightly better mappings than the /16 prefix. For instance, giving the /16 prefix, 9% of locations have at least 50% increase in HTTP latency compared to the whole IP address—giving the /24 prefix, only 3% have a 50% increase. We also conducted this analysis for lookups to an EdgeCast CDN name, and found no difference between providing the /16 prefix, /24 prefix or the whole IP address. *For these CDNs*, we find that a /16 prefix is typically sufficient to provide equivalent mappings to the whole IP address. For a subset of locations, using /24 prefixes can result in marginal performance improvements compared to /16 prefixes.

We stress that these results are not generally applicable to CDNs due to variations in their deployments of edge servers. We would expect to see similar results for CDNs with *like deployments*, since users would be mapped to edge server locations with equivalent granularity in terms of their network location. For CDN architectures with edge servers in *more* locations (e.g. in end-users' networks), we expect that providing more prefix information would result in a greater performance improvement than we observed. However, lacking a CDN with such a deployment that also supports the extension, we cannot experiment directly with the impact of varying amounts of client location information on the performance of this CDN deployment model.



(a) Akamai



(b) Limelight

Figure 11: CDFs of end-to-end latency % difference, comparing ISP DNS and public DNS with and without the extension, for requests to Akamai and Limelight CDNs. Values are relative to the ideal end-to-end latency baseline. Using the extension significantly improves end-to-end performance. For Akamai, the extension improves median performance by 40%; for the top 20% of locations, it reduces latency by 60%.

To understand the impact of the extension in the context of end-to-end performance, we compare the performance of public DNS with and without `ECS` to ISP DNS. Figure 11 plots relative end-to-end latency for accessing objects on the Akamai and Limelight CDNs. Since neither of these CDNs currently support the `ECS` extension, we simulate it by using answers provided by iterative DNS lookups.

Using `ECS` provides a significant performance improvement over public DNS resolution without `ECS`. For instance, the median latency difference for Akamai lookups without `ECS` was 61%; with `ECS`, this was only 33%. The benefits of using the extension are most apparent in the upper tail of the distribution. Without the extension, 15% of locations had a 4× penalty from using public DNS; with the extension, this fraction of locations is less than 5%. Overall, the `ECS` approach provides a significant end-to-end performance advantage when using remote DNS services.

4.3.2 *Extension challenges*

Our analysis has demonstrated the potential benefits of the edns-client-subnet DNS extension for reducing the CDN performance impact of using remote DNS. To realize its performance benefits, however, both the DNS service and CDN must support the extension. In the following paragraphs, we evaluate and discuss the current level of support for the extension.

As part of our measurement experiments, we probed users' ISP DNS servers as well as several public DNS services to check if they supported the extension. Google DNS was the *only public service* that supported the extension. We also found *no ISP DNS services* that supported the extension, this despite the growing use of "remote DNS" architecture by ISPs and its potential performance cost.

To determine the impact of the extension on aggregate web performance, we test popular web sites to see if any of the CDNs they use support the extension. We determine the fraction of web sites and pageviews that utilize the extension to improve client performance. This analysis follows the model we used in §2 for evaluating the fraction of sites using

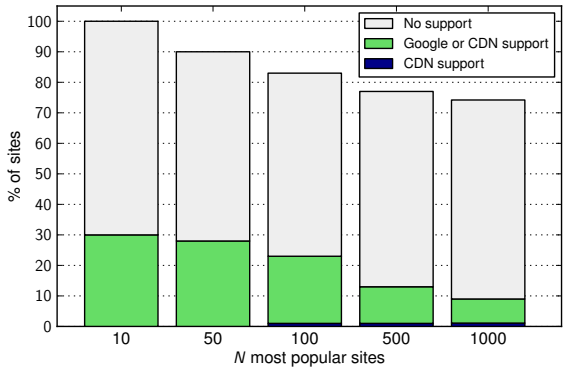


Figure 12: Breakdown of popular sites using CDNs, based on whether the CDNs used support the edns-client-subnet DNS extension. Each bar's overall height shows the % of sites using at least one CDN. The height of the middle region shows the % of sites using a CDN that supports the extension, while the bottom region shows the % of sites when excluding Google sites and services. Only 9% of sites in the top 1,000 use CDNs that support the extension.

CDNs; here, we also test whether each CDN supports the extension.

Figure 12 plots the fraction of sites using CDNs in the N most popular sites. Each site is categorized by whether the CDNs used support the `ECS` extension. Adoption to date is quite limited: out of the top 1,000 sites, only 9% employ a CDN supporting the extension. Most sites are in this category because they either are a Google site (e.g. `www.google.co.uk`) or they use a Google service such as advertising, site analytics or hosted libraries (e.g. jQuery). Setting aside these Google-related sites, we find that only 1% of sites use a non-Google CDN that supports the extension. For the remaining 65% of sites using CDNs without `ECS`

support, their clients are potentially receiving suboptimal CDN redirections and increased page load times.

There is a technical explanation for why CDNs may not have yet adopted the extension. Some large CDNs make their redirection decisions by mapping DNS servers onto a set of "core points" in the network, which are in turn mapped to the CDN's infrastructure [4]; this initial mapping step is non-trivial. Under this approach, the problem is that the CDN cannot directly utilize the information provided by the extension. Neither of the possible solutions—generating a much more extensive set of mappings or changing the overall approach for redirecting clients—are attractive to the CDN. This issue presents a significant barrier to adoption of the extension.

In summary, our analyses have shown that the edns-client-subnet DNS extension has significant potential to improve performance by conveying a client's location to CDNs' authoritative DNS servers to inform their redirection decisions. However, current adoption of the extension is limited to a few CDNs; only 9% of the top 1,000 most popular sites benefit from the potential performance improvements of the extension.

5. AN END HOST SOLUTION

In this section, we present *Direct Resolution* (*DR*), a readily available, end host solution that improves CDN performance when using remote DNS. Our approach is motivated by the potential performance benefits of the proposed DNS extension and its low adoption levels. *DR* does not require the participation of either DNS services or CDNs. We first describe *DR* and explain the intuition and process behind it. We evaluate the impact *DR* has on improving end-to-end CDN performance. Finally, we describe "*namehelp*", an implementation of *DR* that attain performance comparable to the edns-client-subnet DNS extension.

5.1 Approach

In our previous analysis we rely on iterative DNS resolution to attain the best possible redirections and, thus, the highest HTTP performance. The iterative resolution approach, however, is not a suitable solution by itself given its high resolution latency. With iterative resolution, a client must conduct several queries that traverse the DNS hierarchy and translate Canonical Names (CNAME) entries from customer domain names to CDN domains (e.g. `wwwimages.adobe.com` to `a1953.x.akamai.net`) before obtaining an answer. Recursive DNS servers can often answer a client's query faster than if the client used iterative resolution itself. This is because recursive servers answer queries for many clients, enabling them to answer many queries from cache or with few additional queries.

The *Direct Resolution* approach composes the best aspects of recursive DNS servers and iterative resolution to obtain improved CDN redirections. Figure 13 diagrams the high-level *DR* process. *DR* leverages the cache of a recursive DNS server to quickly translate customer domain names to CDN names and determine the CDN's authoritative CDN server to contact (Query #1). However, instead of using the CDN redirection obtained via the recursive server, the client *directly contacts the CDN's authoritative server* (Query #3) as in iterative resolution. The resulting CDN redirection is the same as that which iterative resolution would return but without the iterative latency penalty.

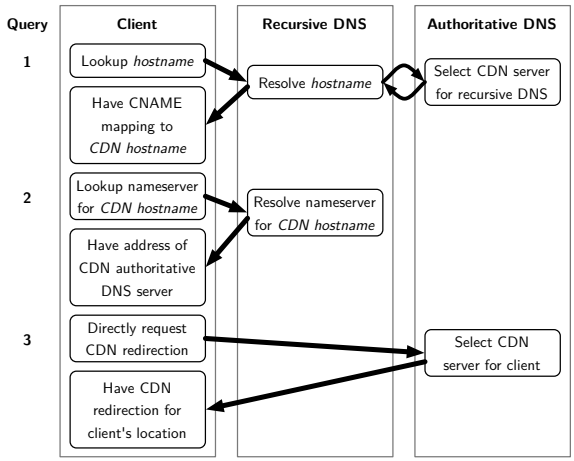


Figure 13: Sequence diagram for *Direct Resolution*.

5.2 CDN redirections using Direct Resolution

In Fig. 13, we outline the 3 steps to obtain a CDN redirection using *DR*. As an example to illustrate how *DR* works, we walk through the steps to obtain a CDN redirection for an object linked on a website (`website.com`) that is served by a CDN (`somecdn.com`).

Query #1 of this process is identical to the query a client makes when using an ISP or public DNS service: the client requests the IP address for `website.com`. The answer includes the requested IP address, as well as any Canonical Name (CNAME) entries to show how the queried domain name was resolved. CNAME entries tell us which CDN hosting the requested object (e.g. `somecdn.com`).

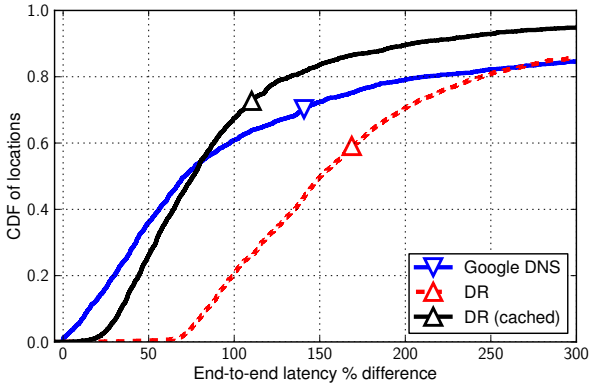
After the first query, the client has obtained a CDN redirection to the client; however, the redirection is based on the *location of the recursive DNS server*. The subsequent queries provide the client with a CDN redirection based on its own location.

To directly query the CDN for a redirection requires that client know the IP address of `somecdn.com`'s authoritative DNS server. Sometimes this information is provided as part of the information in the first query. Most times, however, the authoritative server is not typically included in the DNS response but we may have the answer in a local cache. Alternatively, If the client does not know the authoritative DNS server, then it issues Query #2 asking the recursive DNS server for the authoritative DNS server for `somecdn.com`.

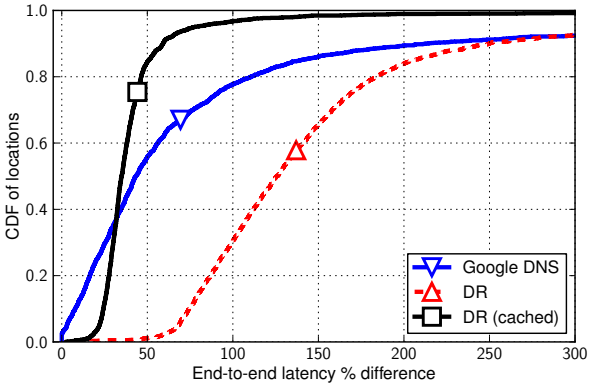
At this point, the client knows the address of the authoritative server and can directly obtain a CDN redirection (Query #3). The CDN's response contains a redirection based on the client's location. The client can then use the CDN redirection from Query #3 to address the server and download the objects for `website.com` from the CDN.

5.3 Performance evaluation

Since our *DR* approach for obtaining CDN redirections is functionally equivalent to iterative DNS resolution, we will



(a) Akamai



(b) Limelight

Figure 14: CDFs of end-to-end latency difference relative to the best-case latency, comparing "Google DNS" with *DR* when the CDN server is not known ("DR") or is cached ("DR (cached)"). "DR (cached)" can provide significantly improved performance compared to "Google DNS".

receive mappings to the same set of edge servers. Since HTTP requests to edge servers are agnostic to *how* the edge server mappings were obtained, we expect that the HTTP performance under *DR* will match that of iterative resolution. As we reported in §4.2, this yields best-case HTTP performance.

To evaluate the *DR* approach in terms of end-to-end performance, however, we must also account for the latency to obtain the CDN redirection. Depending on whether the client has the CDN's authoritative DNS server address in its cache, this entails either two or three queries. When the authoritative server's address is not known, the client must make two queries to the recursive server and one query to the CDN's authoritative server. If the authoritative server is cached, then the client contacts the recursive server and then the authoritative server.

We compare the end-to-end performance of our *DR* approach—with and without knowing the authoritative server—with the performance of simply using a remote DNS service. Figure 14 plots the distribution of % difference in end-to-end latency compared to our ideal baseline, across our measurement locations, for both Akamai and Limelight CDNs. For example, using Akamai, the median location's end-to-end latency when using Google DNS is 60% higher than the ideal baseline. When using *DR*, having the CDN's authoritative DNS server cached ("DR (cached)") provides significantly better performance than when it is not cached ("DR"). Compared to Google DNS, "DR (cached)" provides significant performance improvement, especially in the upper tail of the distribution. Comparing directly Google DNS and "DR (cached)", our solution provides lower end-to-end performance in 49% of locations.

This comparison between our approach and remote DNS shows that the HTTP performance benefits from better CDN redirection do not always outweigh the latency of *DR*'s additional DNS query. We address this by proposing that a client determines whether it should conduct *DR*'s additional queries to obtain a better redirection. For the purpose of our evaluation, we assume the existence of an oracle that determines whether it makes sense to use *DR* or simply the CDN redirection from remote DNS. One way to implement

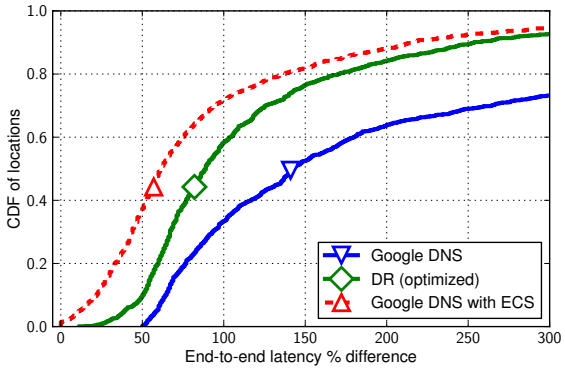


Figure 15: CDFs of end-to-end latency difference vs. best-case latency, for locations with $>$ 50% remote DNS penalty. "DR (optimized)" shows performance for using an oracle to decide whether to use *DR*. This significantly improves on remote DNS performance, and is within a constant factor of the performance of "Google DNS with ECS".

this would be to use the results of previous redirections; when a prior *DR* redirection resulted in the same answer as that seen via remote DNS, *DR* would not be used and the client would return the redirection seen via remote DNS in Query #1.

To evaluate *DR*, we focus on the set of locations where remote DNS usage has the most impact. We select those locations in which the penalty from using remote DNS is at least 50%. This subset comprises 65% of the total set of locations. For these locations, Fig. 15 plots the distributions of % difference in end-to-end latency for Akamai CDN relative to our ideal baseline for *optimized DR* and remote DNS with and without the edns-client-subnet DNS extension.

Generally, *optimized DR* provides significantly improved performance compared to remote DNS. In the median case, *optimized DR* provides 1.6× better performance than remote

DNS (88% vs. 142%) and is even better at the 90th percentile (2.5× better). One reason is that in the extreme cases of poor performance using remote DNS, the HTTP latency to distance CDN edge servers dominates; *optimized DR* can provide significant improvements via nearer edge servers.

We also examine the relative performance of *optimized DR* and remote DNS with the edns-client-subnet DNS extension, finding that the distributions of performance match closely. *Optimized DR* is within 42% of the edns-client-subnet extension's performance in the median case (88% vs 62%) and within 18% at the 90th percentile. The difference in performance between these two approaches is generally consistent across the distribution, and is primarily due to the additional DNS request that the *DR* approach must make to obtain an edge server mapping.

Overall, these results show that the *optimized DR* approach not only attains performance within a constant factor of remote DNS with the edns-client-subnet DNS extension, but that it also significantly improves performance relative to just using remote DNS.

5.4 Implementation

We have implemented our optimized *DR* approach in *namehelp*, a DNS proxy that runs on the user's computer as a daemon and requires no modifications to the operating system. Our implementation is available for download.[6] *namehelp* compares the performance of alternative DNS services with respect to CDN performance, and presents the results to the user. We consider additional implementations, such as integration into operating systems or middleboxes, as future work.

5.4.1 DNS proxy daemon

To provide the benefits of our optimized *DR* approach, *namehelp* functions as a DNS proxy running on the user's computer. Configuring the DNS proxy simply requires the user to specify her DNS server as "localhost", so that all DNS requests will be directed to *namehelp*.

namehelp listens on the default port (53) for DNS queries, and forwards them to a configured recursive DNS service. Simple domain name responses are returned immediately by the *namehelp* proxy with no latency penalty. If the response from the recursive DNS contains a CDN redirection (e.g. a CNAME record), we invoke our *DR* approach to improve the CDN redirection before returning an answer.[7]

5.4.2 Performance comparison utility

namehelp also includes a DNS benchmarking tool based on *namebench* [18]. To compare DNS services, we adopt *namebench*'s approach, which makes queries to both public and ISP DNS servers for a set of domain names. By default, a list of popular sites from Alexa.com is used, but users can use their web browser history to personalize the results. Using the results of these tests, it ranks DNS servers by mean response time and shows potential server-specific issues to the user (e.g. NXDOMAIN hijacking [31]).

namehelp benchmarking service extends the functionality of *namebench* to evaluate CDN performance via different DNS services. Tests can be run against a list of popular websites, or tailored with a user's browsing history. In addition, the tool also probes DNS services to test for edns-client-subnet support, given its potential performance benefits when available (§4.3).

To evaluate end-to-end CDN performance provided by a DNS service, *namehelp* benchmarking downloads the selected web pages and their linked objects (i.e. images, scripts and stylesheets) using the given DNS service to resolve domain names. We determine end-to-end latency using the methodology from §3.3. We adopt a single-threaded architecture for downloading web objects for simplicity. Although optimizations for downloading web objects (e.g. parallel connections and HTTP pipelining) could change the *overall time* necessary to download a web page, our approach yields *comparable* results of end-to-end performance between public DNS services. Caching of DNS lookups and HTTP objects when possible minimizes the test's run time and the bandwidth consumed.

The results of these tests are presented to the user, enabling her to make an informed decision regarding which DNS service to use based not only on DNS performance (as *namebench*) but also including the resulting CDN performance.

6. RELATED WORK

Our work builds upon and significantly extends a number of past studies on DNS, CDNs and their interaction.

CDNs. In an early study by Krishnamurthy et al. [13], the authors evaluate characterize the benefits of CDNs from the client perspective. Johnson et al. [10] studied the redirection performance of CDNs and conclude that they succeed not by selecting optimal servers, but rather by avoiding making bad decisions. Wang et al. [33] analyze methods of maximizing CDN's objectives of improving response time and system throughput under a wide range of loads. Nygren et al. [19] describe several aspects of the Akamai's network, and Triukose et al. [29] studies Akamai's performance. WhyHigh [14] is a tool used by Google to identify and diagnose client performance issues from the CDN's perspective. Although a detailed view of a CDN's network is a valuable perspective for evaluating a CDN, a persistent challenge is accurately capturing the end-user's perceived performance under the range of alternative settings. In this work, we capture this end user's perspective to accurately characterize the end-to-end performance impact of remote DNS usage.

DNS and CDNs. DNS is a critical component of the Internet infrastructure and, since Mockapetris and Dunlap's [17] retrospective study, has been the subject of several measurement analysis (e.g. [1, 11, 15, 16, 22, 25]) and proposed design alternatives (including [23, 24]). A number of studies [1, 9, 12, 16, 25] have evaluated different aspects of DNS-based redirections and CDNs. Mao et al. [16] determine that this is sufficient for coarse-grained (e.g. AS-level) server selection, but not as precise at the granularity of BGP prefixes. Shaikh et al. [25] find the ping time to a client's DNS server is a poor predictor of ping time to the actual client. The recent growth in usage of remote DNS services, such as OpenDNS [20, 21] and Google DNS [6], has motivated studies on the implications of remote DNS and its interactions with CDN redirections. In a preliminary study of DNS resolvers by Ager et al. [1], the authors report

[6] http://aqualab.cs.northwestern.edu/projects/namehelp

[7] *namehelp* can be extended to detect CDNs configured via domain delegation by using a whitelist containing CDNs' authoritative DNS servers.

that typically-employed load balancing techniques have a detrimental effect on the efficacy of DNS caching and that public DNS-based redirection is less likely to point users to an available copy of content within their own network. Huang et al. [9] documented that a user's nearest public DNS server is considerably farther away than that user's ISP-provided DNS server, and that using a more-distant public DNS server resulted in redirections to edge servers that were farther away. Our work builds on and extends these studies by evaluating the end-to-end performance impact of remote DNS on CDNs and conducting an extensive diagnosis of the results.

Solutions. Several approaches have been proposed to address the poor interaction between remote DNS and CDN redirections. Khosla et al. [12] studied the interaction between "cloud-based" remote DNS and Akamai redirections and discussed several possible approaches to mitigate the performance penalty. Huang et al. proposed an application-level solution to address the mismatch between client and local DNS resolvers [8]. Several industry players have formed the "Global Internet Speedup" [28] collaboration. This group advocates for the adoption of the proposed edns-client-subnet DNS extension [3], which passes along part of a client's IP address to CDNs to improve DNS-based server selection decisions. Our work presents the first evaluation of the performance benefits and adoption of this DNS extension. In addition, we are the first to propose and evaluate a general client-based solution that allows *unmodified applications* to enjoy the benefits of improved redirections. We present *Direct Resolution*, an alternative end-host approach to the industry proposed DNS extension, describe *namehelp*, an implementation of the *DR* approach and evaluate its performance benefits.

7. CONCLUSION

We explored the end-to-end impact of using remote DNS services on CDN performance and presented the first evaluation of an industry-proposed solution to the problem. We showed that remote DNS usage can indeed significantly impact client's web performance and that the proposed solution, if available, can effectively address the problem for most clients. Considering the performance cost of remote DNS usage and the limited adoption base of the industry-proposed solution, we present and evaluate *Direct Resolution*, an alternative approach to readily obtain comparable performance improvements without requiring CDN or DNS participation. We have implemented our approach as part of *namehelp*, a DNS proxy running on the client local host. *namehelp* also provides a benchmark service, comparing the impact of alternative DNS services on CDN performance. We have made *namehelp* publicly available. We continue to monitor the adoption of the edns-client-subnet DNS extension by DNS services and CDNs. We are also studying approaches for evaluating the potential performance benefits of this extension when providing partial client location information, while investigating alternative heuristics to improve *namehelp*'s performance.

Acknowledgements

We would like to thank our shepherd, Alan Mislove, and the anonymous reviewers for their valuable feedback and assistance. We are always grateful to Paul Gardner for his assistance with Vuze and the users of our software for their invaluable data. This work was supported in part by the National Science Foundation through Awards CNS 0644062, CNS 0917233 and CNS 0855253 and by a generous Google Faculty Research Award.

8. REFERENCES

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS resolvers in the wild. In *Proc. of IMC*, 2010.

[2] D. R. Choffnes and F. E. Bustamante. Taming the torrent: A practical approach to reducing cross-ISP traffic in peer-to-peer systems. In *Proc. of ACM SIGCOMM*, 2008.

[3] C. Contavalli, W. van der Gaast, S. Leach, and D. Rodden. Internet-draft: Client subnet in DNS requests, 2011. `http://tools.ietf.org/html/draft-vandergaast-edns-client-subnet-00`.

[4] G. Economou. How Akamai maps the net: an industry perspective, 2010. `http://www.akamai.com/dl/akamai/economu_mapping_the_internet.pdf`.

[5] EdgeScope – sharing the view from a distributed Internet telescope. `http://aqualab.cs.northwestern.edu/projects/EdgeScope`.

[6] Google Public DNS. `http://code.google.com/speed/public-dns/`.

[7] J. Hamilton. The cost of latency, October 2009. `http://perspectives.mvdirona.com/-2009/10/31/TheCostOfLatency.aspx`.

[8] C. Huang, I. Batanov, and J. Li. A practical solution to the Client-LDNS mismatch problem. *ACM CCR*, 42, 2012.

[9] C. Huang, D. A. Maltz, A. Greenberg, and J. Li. Public DNS System and Global Traffic Management. In *Proc. of IEEE INFOCOM*, 2011.

[10] K. L. Johnson, J. F. Carr, M. S. Day, and M. F. Kaashoek. The measured performance of content distribution networks. *Computer Communications*, 24, 2001.

[11] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. *IEEE/ACM Transactions on Networking*, October 2002.

[12] R. Khosla, S. Fahmy, and Y. C. Hu. Content retrieval using cloud-based DNS. In *Proc. of IEEE Global Internet Symposium*, 2012.

[13] B. Krishnamurthy, C. Wills, and Y. Zhang. On the use and performance of content distribution networks. In *Proc. of ACM IMW*, 2001.

[14] R. Krishnan, H. V. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. Moving beyond end-to-end path information to optimize CDN performance. In *Proc. of IMC*, 2009.

[15] R. Liston, S. Srinivasan, and E. Zegura. Diversity in DNS performance measures. In *Proc. of ACM IMW*, 2002.

[16] Z. M. Mao, C. D. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang. A precise and efficient evaluation of the proximity between web clients and their local DNS servers. In *Proc. of USENIX ATC*, 2002.

[17] P. V. Mockapetris and K. J. Dunlap. Development of the Domain Name System. In *Proc. of ACM SIGCOMM*, Oct. 1998.

[18] *namebench*. http://code.google.com/p/namebench/.

[19] E. Nygren, R. K. Sitaraman, and J. Sun. The Akamai network: a platform for high-performance Internet applications. *ACM SIGOPS Operating Systems Review*, 44, August 2010.

[20] OpenDNS. More than 1 percent of the world's Internet users now using OpenDNS for a safer, faster, smarter and more reliable connection, March 2010. http://www.opendns.com/about/announcements/160.

[21] OpenDNS. Security industry leader dan hubbard joins opendns as chief technology officer, March 2012. http://www.opendns.com/about/announcements/303.

[22] V. Pappas, D. Wessels, D. Massey, S. Lu, A. Terzis, and L. Zhang. Impact of configuration errors on DNS robustness. *IEEE J.Sel. A. Commun.*, April 2009.

[23] K. Park, V. S. Pai, L. Peterson, and Z. Wang. CoDNS: improving DNS performance and reliability via cooperative lookups. In *Proc. of USENIX OSDI*, 2004.

[24] V. Ramasubramanian and E. G. Sirer. The design and implementation of a next generation name service for the Internet. In *Proc. of ACM SIGCOMM*, 2004.

[25] A. Shaikh, R. Tewari, and M. Agrawal. On the effectiveness of DNS-based server selection. In *Proc. of IEEE INFOCOM*, 2001.

[26] S. Souders. High performance web sites: 14 rules for faster loading pages, June 2009. http://stevesouders.com/docs/velocity-20090622.ppt – Statistic attributed to Greg Linden.

[27] A.-J. Su, D. R. Choffnes, F. E. Bustamante, and A. Kuzmanovic. Relative network positioning via CDN redirections. In *Proc. of ICDCS*, 2008.

[28] The Global Internet Speedup. A Faster Internet. http://www.afasterinternet.com.

[29] S. Triukose, Z. Web, and M. Rabinovich. Measuring a commercial content delivery network. In *Proc. of WWW*, 2011.

[30] P. Vixie. Extension mechanisms for DNS (EDNS0), 1999. http://www.ietf.org/rfc/rfc2671.txt.

[31] P. Vixie. What DNS is Not. ACM Queue, November 2009. http://queue.acm.org/detail.cfm?id=1647302.

[32] Vuze, Inc. Vuze. http://www.vuze.com.

[33] L. Wang, V. Pai, and L. Peterson. The effectiveness of request redirection on CDN robustness. In *Proc. of USENIX OSDI*, 2002.

[34] L. Whitney. Comcast customers hit by another major outage. *CNET News*, December 6 2010. http://news.cnet.com/8301-1023_3-20024692-93.html.

[35] L. Whitney. Major outage hits Comcast customers. *CNET News*, November 29 2010. http://news.cnet.com/8301-1023_3-20023949-93.html.