On Modern DNS Behavior and Properties

Tom Callahan†, Mark Allman‡, Michael Rabinovich†
†Case Western Reserve University, ‡International Computer Science Institute
{trc36,michael.rabinovich}@case.edu, mallman@icir.org

ABSTRACT

The Internet crucially depends on the Domain Name System (DNS) to both allow users to interact with the system in human-friendly terms and also increasingly as a way to direct traffic to the best content replicas at the instant the content is requested. While previous efforts have characterized DNS, the DNS ecosystem has evolved over time and this paper is therefore an initial study into the behavior and properties of the modern DNS system. We passively monitor DNS and related traffic within a residential network in an effort to understand the impact of DNS server behavior—as viewed through DNS responses—as well as client behavior surrounding DNS lookups—as viewed through both DNS requests and traffic that follows from DNS responses.

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]:
Network Protocols

General Terms

Measurement

Keywords

DNS, Measurement

1. INTRODUCTION

The Internet's protocol stack includes the notion of addressing so that traffic can be directed from one host to another. However, the stack contains no notion of naming, even though use of raw IP addresses is cumbersome at best. This was first dealt with by sharing lists of hostname-to-IP address mappings that were installed on each host such that users could deal with human-friendly host names that would be turned into the IP addresses required by the protocol stack. Maintaining one master list of such mappings became burdensome and this method eventually gave way to the Domain Name System (DNS), which is a distributed, hierarchical naming system [15, 16]. The DNS calls for organizational nameservers to hold the authoritative binding between IP addresses and hostnames only for their own hosts. Clients then query for this mapping as needed. This provides human-friendly hostnames without the logistical burdens of maintaining a single master list of all hosts on the network.

Over the years, the DNS infrastructure has grown and morphed in a number of dimensions. For instance, DNS is commonly used as a query mechanism for various blacklists of compromised or otherwise misbehaving actors on the Internet (e.g., [13]). Also, DNS supports named services as well as hostnames to support discovery of transport port numbers. A final—yet crucial—example is that DNS mappings foster agility within the network such that traffic can be directed with fine granularity. There are alternate methods for directing traffic through the network—i.e., routing changes—but these are more cumbersome than using DNS to direct users. First, such changes would be constantly pushing updates through the routing system causing churn. Second, DNS responses are at the immediate and sole control of the content provider, and can be crafted to match content providers' needs at the instant a client is interested in their content. The observation that DNS can deal with late binding of names to specific servers has lead to all manner of innovation in terms of traffic engineering.[1] For instance, Content Distribution Networks (CDNs) often make use of DNS to direct users to exactly the content replica the CDN believes will provide the best experience at the instant the user is interested in the content—and hence triggers the DNS request. The DNS architecture inside the CDN itself may even contain multiple levels. For instance, to mitigate the cost of repeated DNS lookups for content, one CDN DNS server provides long-lived delegations to another layer of CDN DNS servers near client resolvers, which in turn issue short-lived answers to individual lookups to retain fine-grained control over the traffic flow [12].

At its core DNS is a simple protocol with requests and responses each generally contained in a single UDP packet. Further, resolving a hostname requires only a small number of transactions. For instance, finding the IP address corresponding to "www.google.com" first requires finding the authoritative servers for ".com", and then for ".google.com" and finally looking up the IP addresses of "www.google.com". The results from each step can be cached such that future lookups may require even fewer steps. The simple protocol and process, however, belies much complexity in the modern DNS ecosystem. DNS transactions are now mere building blocks for complex systems involving load balancing, caching, traffic engineering and careful management of the user experience. A DNS request triggered by a user clicking a link in a web browser may now travel through multiple layers of DNS resolvers—from the home wi-fi router which often acts as a simple DNS forwarder to several layers of resolvers [9] or a public DNS service provider (e.g., [17]). Therefore, while the DNS protocol is itself simple, much of the resolution process and the actors involved are mysterious and largely hidden from view.

In this paper we describe an initial study of modern DNS behavior as observed from the vantage point of clients within a small residential network over a recent 14 month span. While some of our analysis is a reappraisal of previous work conducted by others [11, 19], we are aware of no recent analysis that passively assesses DNS

[1]Note, we use the term "traffic engineering" loosely in this paper to indicate content providers directing users to specific resources.

behavior even though the DNS ecosystem is constantly evolving—e.g., due to increased use of open global DNS platforms, the popularity of DNS pre-fetching, the increasingly complex structuring of the recursive lookup process that involves pools and hierarchies of servers. This paper proceeds in four steps: (i) in § 3 we study client behavior as reflected in DNS requests, (ii) in § 4 we turn to the DNS server behavior that manifests in DNS responses, (iii) in § 5 we briefly discuss DNS transmission characteristics and finally (iv) in § 6 we turn to investigating the use of resolved hostnames by clients.

2. DATASETS AND METHODOLOGY

For this study we monitor DNS traffic within the "Case Connection Zone" [1], which is an experimental network that connects roughly 90 homes in a neighborhood adjacent to Case Western Reserve University to the Internet via bi-directional 1 Gbps fiber links. The connections from each house come together in a switch. We have a packet-level monitor that receives all traffic via a mirroring port on the switch. The CCZ network is atypical in terms of the capacity afforded to users. However, in a companion study [21] we find that the usage patterns are fairly consistent with those other residential network studies have shown. Further, we find the average CCZ user exceeds 10 Mbps for 1.2 minutes per day. Therefore, we do not believe CCZ users' behavior is to a first order driven by their ultra-broadband links.

We have found our measurement apparatus to drop on the order of 0.01% of packets in the worst case [21]. At this measurement-based loss rate we do not believe the insights from our analysis are skewed by our apparatus. Additionally, we experience measurement outages where no traffic is observed. These are caused by mundane logistical failures (e.g., temporarily unplugging the monitor from the network or disks filling) and from what we can tell not by network phenomena that would suggest a biasing of the resulting data. Our vantage point lies between the users' devices and the recursive DNS server provided for the users and therefore we directly observe client behavior and not the behavior of the recursive DNS servers.

We use the protocol analyzers found in the Bro intrusion detection system [20] to reconstruct DNS queries and responses from the observed packet stream and log these to text-based logs via Bro's standard `dns.bro` policy. The DNS logs include timestamps, involved IP addresses, error code, contents of the question and answer sections of DNS requests and responses, and a summary of the number of records in the authority and additional answers sections of DNS responses. Some of our analysis aims to understand traffic resulting from DNS lookups. For this we use Bro's standard connection log—as produced by the `conn.bro` policy—which summarizes each transport layer connection and includes timestamps, durations, involved IP addresses/ports, amounts of data transferred, application protocols, as well as Bro's standard notion of the connection's ending state (e.g., "normally terminated") and history of key actions observed during the connection (e.g., three-way handshake completed, TCP reset observed, etc.).

The dataset used in this paper was collected continuously from January 25, 2011 through March 31, 2012. During this period we collect just over 200 million DNS queries and about 162 million DNS responses that yield an IPv4 address. These queries were made by roughly 85 client IP addresses.[2] In addition, the connection logs contain 1.1 billion flows over the entire dataset. Of the connections we link with preceding DNS traffic we find 92.2% are HTTP—including ports 80 and 443. This is not surprising since the bulk of the connections we observe are either web connections or peer-to-peer connections [21], and peer-to-peer systems generally use DNS sparingly. For instance, while a BitTorrent client may use the DNS to access a tracker web site to obtain a description of a torrent of interest to the user, the IP addresses of the peers in the given swarm are given by the tracker[3] without relying on the DNS.

Req. Type	5^{th} perc.	Median	95^{th} perc.
A	76.7%	87.5%	90.0%
PTR	5.3%	8.7%	16.9%
AAAA	2.4%	4.1%	9.7%
OTHER	0.0%	0.0%	0.2%

Table 1: Queries by type per month.

3. DNS REQUESTS

Communication across the Internet generally starts with a DNS query from a user-facing device. Some applications continue to heavily rely on DNS over the course of their operation (e.g., web browsers), while others only use DNS for bootstrapping (e.g., peer-to-peer applications, as discussed above). Fundamentally the request to resolve a hostname into an IP address is straightforward and the DNS protocol is likewise uncomplicated. In subsequent sections we will show that the complexity of the system increases when we start to study how these simple requests are handled by the DNS ecosystem and then in turn how the responses are dealt with by the end hosts. The results in this section are basic context for the remainder of the paper.

Table 1 shows the most prevalent DNS query types in our dataset. The table shows that type A queries for IPv4 addresses associated with some hostname are the most prevalent with over 87% of the requests per month at the median. The PTR lookups appear to be predominantly Bonjour discovery traffic. We also find that AAAA lookups for IPv6 addresses constitute a median of 4.1% of the queries per month even though we find almost no actual IPv6 use by CCZ hosts. We believe this is caused by some operating systems making A and AAAA queries simultaneously regardless of whether or not the host intends to use IPv6.

We next turn to understanding the breadth of DNS hostnames requested by the CCZ users. Figure 1 shows the distribution of popularity by hostname and by registered second-level domains (SLD)[4] over our entire dataset. We find heavy-tailed distributions with approximately 63% of hostnames and 50% of domains requested only once over the entire 14 month measurement period. However, we find millions of requests for the most popular hostnames and domains. We also note that some domains use a large number of subdomains—we observe over 100K distinct subdomains for each of google.com, blogspot.com, and tumblr.com.

Finally, we note that roughly 97% of the DNS requests we observe traverse one of the two local resolvers provided for CCZ clients. The primary resolver is used for approximately 81% of the overall requests and the secondary resolver handles 16%. These resolvers are not specifically for CCZ users, but are more broadly used within the Internet service provider supplying access to the

[2]Note, the CCZ project provides a router to each house and this device NATs internal hosts. Therefore, while we observe 85 client IP addresses, this represents more than 85 users. See [21] for a deeper analysis of the number of users.

[3]Peers within the swarm can identify additional peers by IP address after the client joins the swarm.

[4]While technically second-level domains include only the two most senior labels in a hostname (e.g., "google.com"), we also include a third component in our analysis if that third component is what an organization buys from a DNS registrar (e.g., "theregister.co.uk").

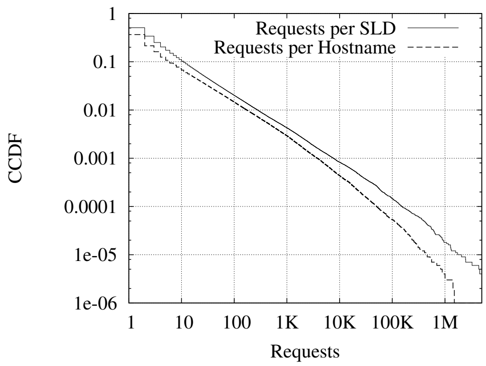

Figure 1: Distribution of # of requests per hostname and SLD.

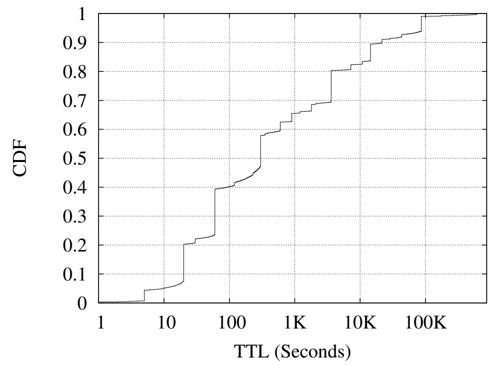

Figure 2: Max. TTL for each distinct answer

CCZ network (OneCommunity). Google's public DNS resolver handles just over 1% of the requests we observe. The remaining 2% of the requests are dealt with by myriad resolvers, including Level3's open resolver, OpenDNS and a Case Western Reserve University resolver.

4. DNS RESPONSES

We next analyze various aspects of DNS responses to understand how content providers leverage the DNS to direct traffic.

4.1 TTLs

We first study the time-to-live (TTL) associated with hostnames in DNS responses. The TTL is assigned by the authoritative DNS server and indicates the time period for which the response is valid. The TTL aims to provide consumers with the ability to cache the result of a lookup such that future requests for the same name can be handled locally, while at the same time allowing the name-to-address associations to expire at some future point such that they can be changed. Longer TTL values allow for higher cache hit rates (and therefore decreased DNS traffic), while lower TTL values allow a service provider more agility in terms of changing their mappings and directing traffic. Precisely where to place the TTL is a complex engineering tradeoff that is ultimately a policy decision for operators.

Most of the user-facing devices within the CCZ use a local recursive DNS server to traverse the DNS hierarchy and find hostname-to-IP address mappings. Therefore, responses can be served from the cache of the recursive resolver and hence the TTL can be less than that originally assigned by the authoritative nameserver. We seek to understand the TTLs being assigned by the authoritative servers. While our dataset does not contain direct interactions with authoritative nameservers, the recursive resolver looking up a name not already in its cache should return the TTL assigned by the authoritative server. The recursive resolver will return smaller TTLs for subsequent lookups. Hence, we use the maximum observed TTL for each name as an approximation of the value assigned by the authoritative server.

In independent work we have observed that some resolvers return bogus TTLs or incorrectly decrement the TTL when responding from the local cache. We therefore tested the TTL handing of the resolvers monitored in this study to ensure our maximum TTL approximation is reasonable. We sent the two CCZ resolvers queries for hostnames from our own domain. When these queries arrive at our authoritative DNS server we respond with a record containing TTL values ranging from 10 to 1M seconds. We then probe for the same names repeatedly to assess the resolvers' TTL decrementing behavior. We find that both monitored resolvers report correct TTLs for values up to one week. TTLs over one week are set to one week. Further, both servers correctly decrement TTLs when responding from the cache.

Figure 2 shows the distribution of the maximum TTL for each hostname in our dataset. The figure shows multiple modes that indicate popular TTL values—e.g., 20 seconds, 60 seconds, 1 hour, etc. We find a median TTL of 5 minutes and that only roughly 1% of the TTLs exceed one day. We also find roughly 40% of the hostnames have TTLs of at most 1 minute—which is suggestive of fine-grain traffic engineering.[5] We also analyze TTLs across time in our dataset and we find that the 5^{th} percentile, median and 95^{th} percentile per month are relatively stable across the 14 month dataset at roughly 5–20 seconds, 300 seconds and 1 day, respectively.

4.2 Equivalent Answers

Individual DNS responses may contain multiple mappings in response to a single query for two reasons. First, multiple IP addresses give clients recourse if they cannot communicate with a particular IP address. Second, this aids load balancing across a set of replicas as recursive DNS servers will rotate the order of the IP addresses returned when serving records from the cache.[6] Figure 3 shows the distribution of (i) the total number of IP addresses returned for each hostname across all responses in the entire dataset, (ii) the average number of IP addresses returned in each individual response for each hostname in the dataset and (iii) the average number of nameservers (NS records) returned in each response for each hostname in the dataset.

We find that roughly 75% of the hostnames map to a single IP address over the entire dataset, with another roughly 8% mapping to two IP addresses. Further, we find that over 11% of the hostnames associate with at least five IP addresses over the course of our dataset. So, while traffic engineering may not be in play for most hostnames resolved within our dataset, some hostnames clearly use a variety of replicas. In terms of the average number of IPs in each DNS response, we find 79% of the hostnames have an average of one IP address in their DNS responses.

[5]We have no insight into the reasoning behind the TTL values in various responses. The values ultimately come from policy decisions encoded by authoritative nameservers. However, these low values seem likely used to enable flexibility in terms of directing traffic as many of these are from well-known CDNs (e.g., Akamai uses TTLs of 20 seconds).

[6]We are not aware of this behavior specified anywhere. However, [6] discusses the behavior in terms of the popular DNS server *bind*. Further, we do observe this behavior from the predominant local

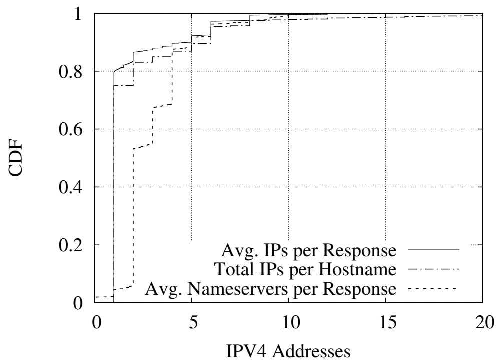

Figure 3: A & NS Record Distributions

The final line on the plot shows the distribution of the average number of nameservers identified for each hostname. We often find more nameserver records in a response than IP addresses. Since NS records form the basis of queries for IP addresses of content servers we hypothesize that providers work hard to ensure resolvers have a reliable and up-to-date set of NS records. While only two nameservers are required to register a domain, almost 50% of responses include at least four NS records in their responses—underscoring the crucial nature of these records.

4.3 Proximity

We next turn to understanding the geographic distance between returned IP addresses and the requesting user device. It is well known that popular services attempt to provide content from nearby servers and that directing traffic via DNS is a popular way to drive clients to close servers (e.g., this is what the Akamai CDN attempts to do [7]). We use a geolocation database from MaxMind to map IP addresses to locations at a city-level granularity [14].[7] Since DNS responses can have multiple answers we consider the geographic distance to each position in the DNS response independently. For each DNS response we calculate the average distance to each position in the response. For all such averages we then compute the quartiles for each position and each DNS response size. For instance, for responses with two IP addresses we calculate the quantiles for the first and second positions independently. We determine the quantiles for DNS responses containing 1–4 IP addresses. We find the distance quantiles are the same regardless of position in the DNS response. This underscores the equivalence of these addresses, as discussed above. There is no proximity preference in the ordering of answers in a response. As we will show below, providers typically attempt to list a set of equally close addresses.

Given that the quantiles for the positions within responses of a given size are the same, we ignore position and report the distance as a function of the number of IP addresses in a response in Table 2. We find that hostnames that map to only a single IP address are less popular and hence not optimized for locality and are roughly 2K miles away from the CCZ clients at the median point. Hostnames that use two or three IPs per response are the closest to the clients. This follows our intuition given that Akamai is heavily used for CCZ traffic [21], returns two IPs per response [24] and is known to use geography as a criteria in choosing replicas for clients [7]. Finally, we note that DNS responses listing four IP addresses

# IPs	25^{th} perc.	Median	75^{th} perc.
1	851	2,036	2,159
2	851	851	1,741
3	851	1,119	2,159
4	2,154	2,154	2,156

Table 2: Distances (miles) to returned IPs.

are generally further away from the clients than those with fewer addresses. We have no ready explanation for this observation.

Since we find the distance between the CCZ and the location of the IP addresses provided in DNS responses to be invariant of the position within the response we hypothesize that the IPs in a given response are generally within the same geographic area. We test this hypothesis by computing the average of the pairwise geographic distances between each pair of IPs in each DNS response. We find the average pairwise distance to be less than 10 miles in over 93% of the cases—showing that indeed the IPs returned as part of a given response are largely within the same location.

4.4 Case Studies

As discussed above, the modern DNS ecosystem is widely used for traffic engineering and replica selection as opposed to simple hostname lookups. The subsections above provide information concerning this traffic engineering process along a number of dimensions, but these do not fully capture the dynamic nature of the DNS resolution process. While in this initial work we do not yet have a comprehensive model for capturing highly dynamic DNS traffic engineering activity, it is a topic of our near term future work. As a glimpse into the process we briefly study the replica selections of two popular CDNs: Akamai and Google. We should stress that our discussion here is in terms of the IP addresses returned in DNS resolutions; because of a possibility of multiple addresses assigned to the same physical machine, or addresses belonging to virtual rather than physical machines, or dynamic IP addresses, we cannot exclude a possibility that the distinct IP addresses may not correspond to distinct physical replicas.

We first look at the answers returned by Akamai on behalf of iTunes. Previous work shows that Akamai uses a large number of servers [7] and performs fine-grained replica selection based upon network conditions and load [24]. Figure 4 shows a scatter plot where each point represents an IP address from a DNS response. The x-axis shows the time of the response and the y-axis shows the IP address—where the addresses have been mapped to integers based on the order of first observation. We observe 140K responses during the course of our dataset and find nearly 300 server IP addresses. We find a general base set of addresses across most of our observation period as shown by the two thick black lines in the plot. However, we also find many additional servers throughout our measurement period. As we will show in the next section, clients use the first IP address in DNS responses with higher likelihood than subsequent addresses. We constructed a plot similar to figure 4 but considering only the first IP in each response. We elide this plot as it is largely indistinguishable from figure 4. However, the close similarity between the two plots indicates that the IP address rotation performed by the recursive DNS server provided for the CCZ users is in fact working to distribute load across the advertised servers.

A second case we examine is the replica selection over time by the Google CDN serving YouTube content. Although YouTube employs a complex combination of HTTP redirections and DNS resolutions to distribute user requests [3], we focus purely on DNS resolutions. Figure 5 shows the timeline of nearly 40K responses to

resolvers used by CCZ hosts.

[7] As our study spans 14 months and GeoIP data is not static, we use archived GeoIP databases corresponding to the month of each DNS transaction.

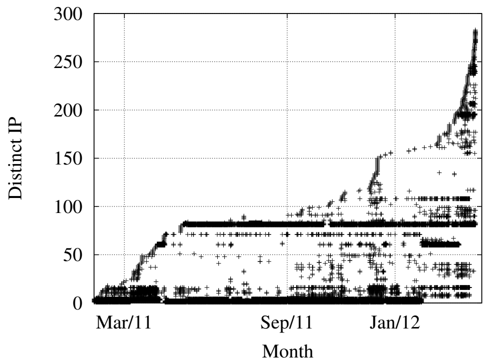

Figure 4: Akamai IPs Returned Over Time

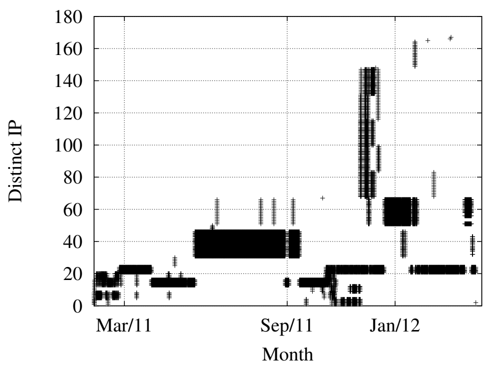

Figure 5: YouTube IPs Returned Over Time

CCZ clients. While Google maintains its own CDN, we do not observe the variety of redirection as we do with Akamai. We find just over half as many server addresses in use for YouTube as we find for iTunes/Akamai. Further, while we find no base set of servers in use throughout our dataset, the responses are more stable across time than those of Akamai. We do find shifts in server addresses at various points in the dataset, but server assignment between shifts is more orderly than for the iTunes replicas shown above.

These two cases are illustrative and not necessarily indicative of any general patterns, but rather show that different strategies for leveraging the DNS to direct clients to replicas are in use by different providers. By considering request routing from a client site's perspective, our observations compliment studies aimed at mapping the content delivery infrastructures as a whole such as [5, 25].

5. TRANSMISSION

While we aim to understand various facets of clients and servers via DNS requests and responses above, we now turn to the process of transmitting DNS mappings. The first order property of DNS interaction is speed and therefore we analyze the response time of DNS lookups, i.e., the time between a DNS request and its response. For this analysis we first remove the 3% of the transactions in the dataset that do not involve the CCZ recursive resolvers (as discussed in § 3). Using an alternate resolver makes the measured response times not directly comparable with those transactions using the CCZ-provided resolvers. To make our analysis tractable we then uniformly sample 10% of the transactions from across the dataset to further analyze.[8]

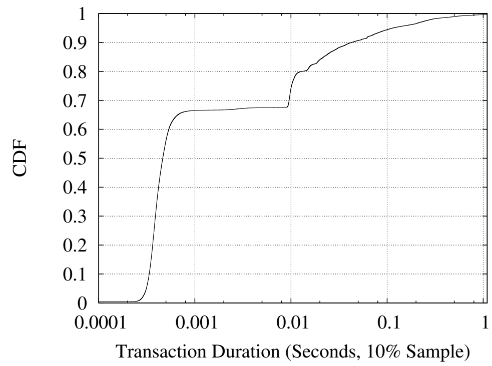

Figure 6: All DNS Response Times

Figure 6 shows the distribution of the response times of the DNS lookups. Our first observation is that roughly two-thirds of the transactions complete in under 1 msec.[9] This suggests that these DNS requests are being serviced from the cache at the CCZ resolver. Another roughly 10% of the transactions take approximately 10 msec. These transactions are likely associated with traffic to nearby institutions (e.g., traffic with the Case network or one of the nearby hospitals). The third region of the plot shows that the longest 25% of the transactions are well spread across the interval between 10 msec and 1 second. These are transactions that have to make their way through the global DNS ecosystem to a variety of authoritative DNS servers around the world.

Figure 7 shows the DNS response time for lookups that are not served from the CCZ resolver cache (i.e., the tail of figure 6). We find the median response time of a lookup not utilizing the cache is about 20 msec. While this is in fact lower than previous studies report [11, 4], we are not able to determine any sort of concrete trends due to the difference in vantage points involved. We also compute the minimum DNS response time to each SLD in our dataset. The distribution of these minimums is also shown in figure 7. We find that the minimum per-SLD distribution shows longer lookup times than when considering all transactions. This indicates that the overall distribution of lookups is skewed towards closer DNS servers. Or, put differently, popular hostnames are served from closer authoritative servers than less popular hostnames. In fact, we find that 23.5% of DNS responses include a redirection to an Akamai server and 13.4% to a Google server. Given these well connected heavy hitters, it is unsurprising that the overall response times are skewed to lower values.

6. UTILIZING RESPONSES

We now tackle questions pertaining to how user-facing clients employ the information conveyed in DNS responses.

6.1 Use of DNS Responses

The first aspect of clients' reaction to DNS responses is simply whether or not the responses are used for subsequent communication. We find that across our entire dataset only about 60% of the DNS responses contain IP addresses that are used in any way

[8] We tried different sampling rates and the results hold and therefore we do not believe the sampling introduces bias into the analysis.

[9] Recall that our vantage point is at the aggregation switch, and thus our observed durations do not include the delay between the switch and the client, which we found to also be on the order of a millisecond. However, for purposes of identifying behavioral regions, the absolute value of the duration is not important.

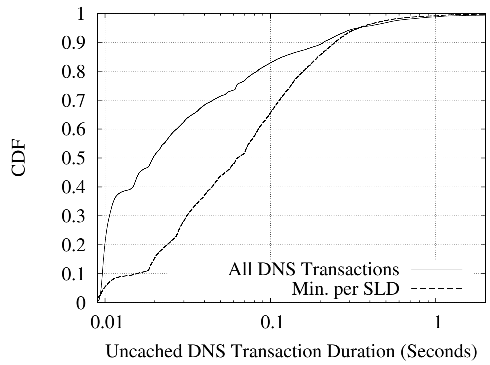

Figure 7: Uncached DNS Response Times

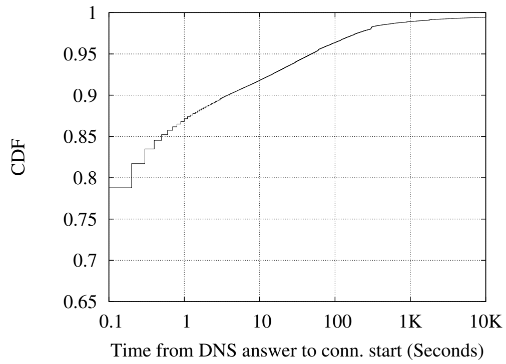

Figure 8: Time From DNS Response to First Connection

for follow-on traffic. In other work we use OS fingerprinting techniques to roughly quantify the number of users behind the NATs in the CCZ [21]. During that work we have determined that users within the CCZ make heavy use of the Chrome web browser [23], which employs DNS prefetching to reduce the web page load time experienced by users [2]. While we cannot distinguish between DNS prefetching and other behavior that may cause DNS responses to go unused, it seems likely that DNS prefetching is at least partially responsible for the 40% of responses that go unused.

Given that we suspect DNS prefetching is in use, we next examine the time between a DNS response arriving and subsequent traffic involving an IP address included in the DNS response for the 60% of the DNS responses with follow-on traffic. Figure 8 shows the distribution of the elapsed time between the DNS response and connection initiation. We find that connections are initiated within 100 msec roughly 79% of the time and within 1 second roughly 87% of the time. These are likely cases where the application is directly using the DNS responses it receives and not trying to optimize future communication by prefetching mappings that may or may not be needed. Therefore, for DNS lookups that are ultimately used, at most 13% are the result of prefetching. Without additional information about application logic this bound may be a tenuous presumption at the lower delays. However, we note that in 4.6% of the cases the use of an IP comes at least 1 minute after the DNS lookup. Such delays are difficult to attribute to anything other than speculative DNS lookups.

As discussed above, DNS responses often contain more than one IP address for a particular hostname. In principle, these IP addresses are equivalent and clients can use any one of the set. We now turn to understanding which IP addresses clients actually do use in subsequent communication. We first exclude all connections that follow from DNS responses that contain only one IP address. Then for each connection we determine the position of the destination IP address and the number of addresses in the corresponding DNS response. Figure 9 shows the percentage of use we find for each position within DNS responses, as a function of the number of addresses in the response. Each bar is divided into regions that represent the percentage of use for each position in the answer. For instance, the third bar represents all responses containing four addresses and is divided into four regions. The bottom region of each bar represents the use of the address in the first position with the address positions progressing up the bar with the top region representing the use of the last address found in the responses.

We find that more than 60% of the connections are made to the first IP address listed in the response across all response sizes with the exception of responses with five addresses—and, even in that case the first address is used in just under 60% of the subsequent connections. Further, for responses with 3–6 IP addresses the use of all but the first position is fairly uniform, which suggests that some applications are choosing randomly from the returned set of addresses. This uniformity generally holds for responses that contain 7–10 addresses, as well, with one exception. For responses with 8–10 addresses the exception is the last address listed—which is used more often than the others, suggesting a "use last" policy. For responses with 7 addresses the most common address (after the first) we find in subsequent traffic is the 4^{th} address. We have no firm explanation for how addresses are chosen in general. However, we note that common programming languages often contain shortcuts to provide the first IP address in a DNS response [8, 18].

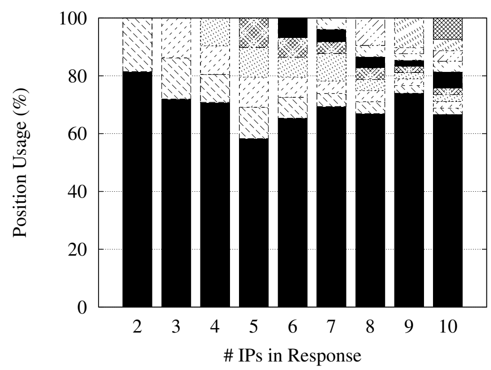

Figure 9: Used Answer Position

6.2 TTLs

We next focus on how well clients honor the TTLs given in DNS responses. Previous work observes that around 47% of clients and their resolvers in combination violate the TTL and that most violations extend beyond two hours after the mapping expires [19]. Further, [22] finds that 8–30% of connections utilize expired DNS records across several datasets from different vantage points. In our dataset we find that 13.7% of TCP connections use addresses from expired DNS responses—significantly less than indicated in [19]. We speculate the cause is modern browsers "pinning" IP addresses to hostnames in an effort to mitigate the impact of cross-site scripting attacks [10]. Figure 10 shows the distribution of the elapsed time between a record's expiration and when it is used. Some connections come seconds after the expiration. These are likely unavoidable—having to do with an application looking up a short-lived mapping and then using it slightly after it expires or the expiration process of a local cache being somewhat coarse. We find that roughly 60% of the violations come 1 minute—1 day after the expiration. The median elapsed time between expiration and use is 168 seconds. Again, our results show significantly smaller violation sizes than in [19], which reports that over 95% of violations exceed 1K seconds.

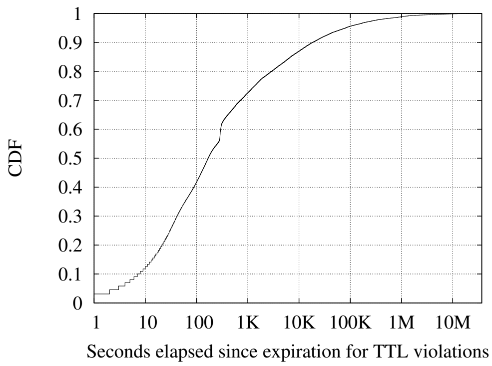

Figure 10: Distribution of TTL Violations

Our results show that while in some cases content providers and CDNs are trying to shape traffic at a fine granularity by using DNS mappings with short TTLs, web browsers are actively working in the opposite direction and violating the specified TTL to protect against attacks. Ignoring TTLs can work because even if a DNS mapping is no longer optimal for a particular client, the mapping generally continues to be valid well past the TTL expiration. Still, this points to a need for better security mechanisms that do not call for clients to violate protocols in order to protect themselves.

Finally, we revisit the length of TTLs in the context of the traffic that is derived from DNS responses. In § 4.1 we consider the TTLs set by content providers as a function of the hostname (Figure 2)—hence weighing all hostnames equally. We now take a different approach to understand TTLs based on client usage. For each maximum TTL we find in our dataset, we count both the number of connections made and bytes downloaded by the CCZ clients that are associated with the given TTL. This indicates the amount of traffic "shapable" at a given time granularity. Figure 11 shows the distributions of these two metrics as a function of maximum TTLs. We find that using this metric skews the tail of the TTL distribution to somewhat lower values. While roughly 55% of hostnames have TTLs of at most 350 seconds, we find that 68% of connections and 87% of traffic volume is associated with DNS records with TTLs of at most 350 seconds. Further, while 80% of hostnames have TTLs of at most one hour (§ 4.1), we find that 85% of connections and 95% of bytes associate with DNS TTLs of at most one hour.

7. SUMMARY AND FUTURE WORK

The contribution we make in this paper is an initial empirical understanding of a broad range of characteristics of modern DNS behavior. Specifically, we focus on facets of DNS behavior that pertain to the phenomenon of enabling traffic engineering using DNS mappings. While previous studies have tackled empirical assessment of DNS, we believe that since the DNS is an evolving system and these studies were conducted many years ago that our reappraisal is timely. To underscore this point, we note that our analysis covers aspects rarely or never seen during previous studies (e.g., prefetching).

While this paper represents an updating of our understanding, we also stress that it is only an initial step. Our data comes from a small network and broader data would be useful. We have gathered data from larger networks, however, it is not from the same user-centric vantage point as we have in the CCZ—making it more difficult to understand client behavior from this broader data. That said, we are actively analyzing this additional data to gain understanding where possible. Finally, as discussed in § 4.4 we are starting to work on a replica selection model that we can apply to all traffic, rather than relying on manual case studies as we have herein.

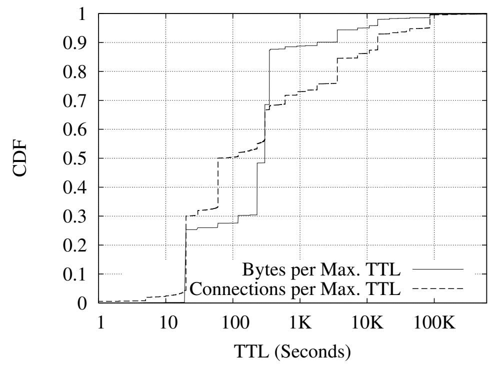

Figure 11: Weighted TTLs

Acknowledgments

Lev Gonick, Mike Chalkwater, Lou Changeri, Tu Ouyang and Marv Schwartz facilitated our monitoring of the CCZ network. Comments from the anonymous reviewers improved the paper. The work was funded in part by National Science Foundation grants CNS-0831535, CNS-0831780, CNS-1213157, and CNS-0831821. Our thanks to all!

8. REFERENCES

[1] Case Connection Zone. http://caseconnectionzone.org/.

[2] DNS Prefetching - The Chromium Projects. http://www.chromium.org/developers/design-documents/dns-prefetching.

[3] V. K. Adhikari, S. Jain, Y. Chen, and Z.-L. Zhang. Vivisecting youtube: An active measurement study. In *INFOCOM*, pages 2521–2525, 2012.

[4] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS Resolvers in the Wild. In *Proceedings of the 10th annual conference on Internet measurement*, pages 15–21. ACM, 2010.

[5] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Web content cartography. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet measurement*, pages 585–600, 2011.

[6] P. Albitz and C. Liu. DNS and BIND, Fourth Edition. http://docstore.mik.ua/orelly/networking_2ndEd/dns/ch10_07.htm, April 2001.

[7] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. Globally Distributed Content Delivery. *Internet Computing, IEEE*, 6(5):50–58, 2002.

[8] GNU. C library: Host names. http://www.gnu.org/software/libc/manual/html_node/Host-Names.html.

[9] Google. Google public dns performance benefits. https://developers.google.com/speed/public-dns/docs/performance.

[10] C. Jackson, A. Barth, A. Bortz, W. Shao, and D. Boneh. Protecting Browsers from DNS Rebinding Attacks. *ACM Trans. on the Web*, 3(1), 2009.

[11] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. *Networking, IEEE/ACM Transactions on*, 10(5):589–603, 2002.

[12] T. Leighton. Improving performance on the internet. *Communications of the ACM*, 52(2):44–51, 2009.

[13] J. Levine. Rfc 5782: Dns blacklists and whitelists. 2010.

[14] MaxMind. City-Level GeoIP Database. `http://dev.maxmind.com/geoip/geolite`.

[15] P. Mockapetris. Domain Names - Concepts and Facilities, Nov. 1987. RFC 1034.

[16] P. Mockapetris. Domain Names - Implementation and Specification, Nov. 1987. RFC 1035.

[17] OpenDNS - A Technical Overview. http://www.opendns.com/technology.

[18] Oracle. Java api: Inetaddress.getbyname(). `http://docs.oracle.com/javase/6/docs/api/java/net/InetAddress.html#getByName(java.lang.String)`.

[19] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan. On the Responsiveness of DNS-based Network Control. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. ACM, 2004.

[20] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, Dec. 1999.

[21] M. Sargent, B. Stack, T. Dooner, and M. Allman. A First Look at 1 Gbps Fiber-To-The-Home Traffic. Technical Report 12-009, International Computer Science Institute, Aug. 2012.

[22] C. Shue, A. Kalafut, M. Allman, and C. Taylor. On Building Inexpensive Network Capabilities. *ACM SIGCOMM Computer Communication Review*, 42(2), Apr. 2012.

[23] B. Stack. Personal Communication, May 2012.

[24] A. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante. Drafting Behind Akamai (Travelocity-based Detouring). In *ACM SIGCOMM*, 2006.

[25] S. Triukose, Z. Wen, and M. Rabinovich. Measuring a commercial content delivery network. In *Proceedings of the 20th International Conference on World Wide Web*, pages 467–476, 2011.