Putting DNS in Context

Mark Allman
International Computer Science Institute

ABSTRACT

Internet traffic generally relies on the Domain Name System (DNS) to map human-friendly hostnames into IP addresses. While the community has studied many facets of the system *in isolation*, this paper aims to study the DNS *in context*. With data from a residential ISP we study DNS along with both activity before an application needs a given mapping and the subsequent application transaction. We find that a majority of applications transactions (*i*) incur no direct DNS costs and (*ii*) for those that do the cost is minimal.

CCS CONCEPTS

• **Networks → Naming and addressing**; **Network performance analysis**.

KEYWORDS

Internet, DNS, performance, measurement

ACM Reference Format:
Mark Allman. 2020. Putting DNS in Context. In *ACM Internet Measurement Conference (IMC '20), October 27–29, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3419394.3423659

1 INTRODUCTION

The majority of Internet traffic relies on the Domain Name System (DNS) [17] to map human-friendly hostnames into IP addresses. This task and the DNS protocol itself seem relatively simple at first glance. However, the ecosystem that has emerged around resolving hostnames has become both complex and mysterious [25]. The community has focused much attention on developing an understanding of the operation and performance of the DNS ecosystem. Generally the previous work falls into two groups: (*i*) studies that deeply explore a particular facet of the DNS (e.g., performance), or (*ii*) studies that explore DNS' role within broader application transactions using synthetic traffic. Both classes of previous work have fundamentally advanced our understanding of DNS. However, both have a drawback: the experiments and measurements are conducted in *isolation*. For instance, studies that deeply explore the duration of DNS lookups may not consider how or when an application uses DNS information. A hefty DNS delay that happens once per hour of application use may not be a significant problem. Meanwhile a small DNS delay for a lookup that is associated with only a single short transaction, but repeats incessantly may sum to a large amount of waiting time for a user. While some of the previous work puts the DNS delays in the context of synthetically loading a web page, the download is isolated from other possibly germane activities. E.g., some previous download may have used a given hostname and populated the cache such that the subsequent access pays no direct DNS cost.

In this paper we aim to augment the DNS literature by evaluating DNS *within the context* of application transactions and user behavior. We use a one week dataset of in-situ DNS lookups and application transactions within a residential setting to understand DNS' role in real world traffic. In particular, with respect to the beginning of each application transaction, we use the following lenses:

Looking backward ... We first aim to understand the location from which DNS information is retrieved. While all DNS records ultimately originate at authoritative servers, the records can be cached at various locations. Our analysis allows us to understand how previous behavior populates such caches and aids subsequent transactions. We find that only 15.7% of application transactions require queries to an authoritative nameserver with the balance leveraging cached information.

Looking forward ... Our second contribution is to understand the magnitude of DNS' contribution to application transactions. For short transactions, the cost of a DNS lookup may be relatively high in terms of adding to the length of the transaction. On the other hand, for long transactions, a DNS lookup may not be noticeable to a user as the time required by the transaction itself dominates.

2 RELATED WORK

Our work relates to several broad classes of previous work.

Focused Studies: Much previous DNS work tackles a particular aspect of the system to understand or improve. Example topics—with a meager set of reference examples—include: delays [15], security [11, 13, 26, 31], caching [12, 18], prefetching [7], robustness [2, 29]. The result is a rich body of work that deeply explores many facets of the DNS in isolation. Our goal is to understand how a number of these aspects manifest in the context of DNS' actual use.

Measurement & Characterization: A second body of work eschews isolated studies in favor of measuring and characterizing many aspects of the DNS in the wild. Examples of such research are numerous. For instance, a number of studies characterize the traffic that arrives at the root nameservers, including aspects such as volume, request types and recursive resolver behavior [4, 6, 14, 33]. Meanwhile, other work focuses on understanding the components and behaviors of DNS' recursive resolution infrastructure [25]. Still more work uses passive DNS observation in edge networks to quantify many aspects of traffic—e.g., TTLs, TTL violations, caching efficacy, name popularity. Studies have been conducted in residential [5], campus [27] and mobile [16] edge networks.

Performance: More closely related to this paper is a body of work that uses active measurements to investigate DNS as a component of web page downloads. Examples of this include the web profiling

IMC '20, October 27–29, 2020, Virtual Event, USA
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8138-3/20/10…$15.00
https://doi.org/10.1145/3419394.3423659

(WProf) work [32], the Mirage work that leverages home routers as vantage points [30] and a study that uses the Lumen app to understand DNS' contribution to web page loading in mobile networks [16]. This previous work crucially begins to build an understanding of DNS' role in a broader context. Our study takes this notion a step further by both (*i*) considering applications in addition to the web and (*ii*) using in-situ instead of synthetic traffic.

Leveraging Slack in DNS: Two of our previous studies have developed techniques that increase the duration of DNS lookups [1, 24]. Part of these studies use passive measurements to illustrate that there is often a gap between a DNS lookup and its eventual use by an application—hence making longer DNS transactions more palatable. This paper is a more in-depth study of this phenomenon.

3 DATASET

Our dataset comes from monitoring the Case Connection Zone (CCZ)—an experimental FTTH network in a neighborhood adjacent to a Case Western Reserve University. Each of the roughly 100 residences in the neighborhood connects to the Internet via a bi-directional 1 Gbps link. This link serves as the primary Internet link for each house and previous work illustrates usage roughly matches expectations for general residential users [22, 23]. Roughly half the residences are student rentals, with the remainder being full-time non-student residents. From previous work in characterizing the traffic in the CCZ [22, 23], we believe that while the CCZ is small and provides high capacity, the traffic is akin to we would expect from a residential network. We do not believe the high capacity within the CCZ biases the results because for the most part our results are in terms of delay and not throughput. Further, in those cases where we do consider throughput, our comparisons are between transactions within the CCZ. The results presented in this paper are meant to add to our understanding and not be the definitive or final word. Results from different settings may vary.

Each house in the CCZ is supplied a router. While many home routers can act as DNS resolvers[1], the supplied routers in the CCZ do not take part in DNS lookups.[2] We monitor the network at the ISP's first aggregation point where we can view all traffic to and from all houses. The supplied home routers act as NATs and therefore our view of the traffic is at the house granularity. We cannot differentiate between devices or users within houses.

For this study we use two datasets captured by the Bro network monitoring system [21] from Feb. 6–12, 2019. A sketch of the ethical considerations of our data collection, storage and use is in § A.

DNS Transactions: The first dataset consists of 9.2M DNS transactions, including timestamps, local and remote IP address, query strings, all returned resource records and ancillary information from the transactions. In this study we consider only plain-text DNS transactions. At the time our dataset was collected encrypted DNS—either DNS-over-TLS (DoT) [9] or DNS-over-HTTP (DoH) [8]—was not mature enough to be in broad use. We validate this assertion empirically in § 5.1. Widespread use of encrypted DNS would render the study we conduct in this paper impossible and therefore future efforts to better understand DNS in context will likely need to be conducted from end systems.

[1]These home routers are generally DNS "forwarders" as they do not communicate with authoritative servers [25]. Rather, they send DNS requests to other ISP or institutional recursive resolvers and cache the results.

[2]This is the general CCZ configuration, not something we instituted for our experiment.

Resolver	% Houses	% Lookups	% Conns	% Bytes
Local	92.4	72.8	74.0	70.8
Google	83.5	12.9	8.3	9.2
OpenDNS	25.3	9.4	14.2	13.5
CloudFlare	3.8	3.9	2.9	5.7

Table 1: Use of resolver platforms in our dataset.

Application Transactions: The second dataset consists of summaries of 11.2M connections originated by hosts within the CCZ, including timestamps, IP addresses, port numbers, number of bytes transmitted in each direction, etc. The connection summaries do not include payload contents. For TCP, Bro tracks connection setup (SYN) and tear down (FIN & RST) indications to delineate connections. Bro defines a UDP "connection" as consisting of all packets involving a common set of IP addresses and port numbers.[3] A timeout of 60 sec after the last observed packet is used to signal the end of the "connection". The dataset includes 9.9M TCP connections (88%) and 1.3M UDP connections (12%).

The CCZ provides two DNS resolvers. Further, most residences use at least one third-party public DNS resolver platform. Table 1 lists the resolver platforms that are used for at least 1% of the DNS lookups in our dataset. The ISP's two resolvers are the most popular, being used—at least in part—by over 92% of the houses and handling over 72% of the queries. The table also shows the percentage of connections and traffic volume that are tied to each resolver—which are both roughly commiserate with query percentage.

Further, roughly 16% of the houses only use the ISP's resolver. The remaining houses both (*i*) leverage Google's public DNS resolvers for some of their lookups and (*ii*) use multiple resolvers. We believe the prevalent use of Google's DNS platform at the house level stems from a default setting in Android devices. Since the only houses that do not use Google's DNS resolver are those that *only* leverage the local ISP's resolvers, we hypothesize—without being able to prove—that these houses have installed a DNS forwarder that intercepts all DNS traffic and directs it to the ISP's resolvers.

4 METHODOLOGY

Our analysis depends on the following two building blocks:

Pairing: DNS lookups and application connections—e.g., fetching a web object—appear as independent transactions to a network monitor. However, to put DNS in context we use the DN-Hunter technique [3] to pair each connection with the DNS lookup it uses (if any). Consider an application connection originating from local IP address L and destined for remote IP address R. We pair that connection with the most recent non-expired DNS lookup conducted by L that contains R in the answer (if such exists). If all previous DNS lookups containing R are expired, we use the most recent.[4]

Given the widespread use of centralized hosting in the modern Internet it is possible that multiple hostnames resolve to the same IP address, thus confusing the DN-Hunter algorithm. In our dataset, we find there is only a single non-expired DNS response that includes the given target IP address for over 82% of the application transactions. For the remaining application transactions there exists multiple non-expired DNS records with the given IP address. Hence,

[3]We implicitly cover QUIC [10], as we treat such connections as a UDP "connection".

[4]Note, we discuss the use of expired DNS records in more detail in § 5.2.

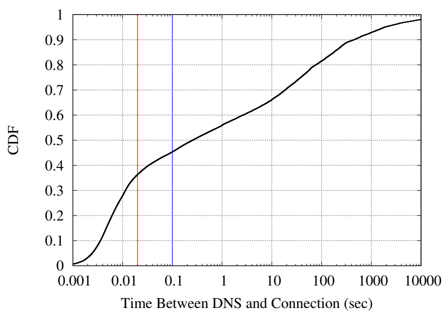

Figure 1: Dist. of time between completion of a DNS lookup and the start of an application connection.

given our vantage point we cannot be certain that the most recent DNS transaction is the correct choice for pairing—even if intuitive. To understand the impact of this ambiguity we ran an alternate version of the analysis in the remainder of the paper whereby we pair a random non-expired DNS transactions that contains the target IP address. Due to space constraints we do not present these results, but the magnitude of the deviations from the results we present are small and the high-level take-aways remain unchanged.

Blocking: To better understand DNS' role, we determine whether application connections (i) are *blocked* waiting on a DNS lookup to proceed or (ii) rely on previously looked up DNS information that is readily available. A network monitor has no concrete way to make this determination and therefore we design a heuristic to infer the relationship. Figure 1 shows the distribution of the interval between the start of each connection that relies on DNS and time the paired DNS transaction completes. The distribution shows two different regions—with a knee around 20 msec (red line on plot)—which follows our intuition that some connections are blocked awaiting the results of a DNS lookup while the DNS information is readily available for other connections. We find that 91% of the connections that start within 20 msec of the DNS lookup are the first to use the DNS lookup—i.e., these connections are less likely to be relying the device's local DNS cache. In contrast, only 21% of the connections that start more than 20 msec after the DNS lookup are the first to use a given lookup. Therefore, we use the two regions of the distribution to determine whether connections are blocked on DNS lookups or not. While 20 msec is visually around the knee of the curve, we use a more conservative threshold of connections starting within 100 msec of the DNS lookup completion—marked with a blue line on the plot—being called "blocked" in our analysis.[5]

5 DNS INFORMATION SOURCE

We now tackle the origin of DNS information for each connection.

5.1 No DNS Information Required

First, we focus on the first line of Table 2, or the 7.2% of connections that do not utilize DNS lookups—denoted as the $\mathcal{N}$ connections. We find that in 81.6% of these connections both ports are non-reserved "high ports". This is a hallmark of peer-to-peer traffic

[5]We call 100 msec "conservative" because our analysis generally considers DNS delays more important for blocked connections and, hence, the larger the threshold the more important DNS' role. We ran our analysis with a range of thresholds and find that while the numbers change slightly, the overall insights remain as we present them.

Class	Desc.	Conns (M)	% Conns
$\mathcal{N}$	No DNS	0.8	7.2
$\mathcal{LC}$	Local Cache	4.8	42.9
$\mathcal{P}$	Prefetched	0.9	7.8
$\mathcal{SC}$	Shared Resolver Cache	3.0	26.3
$\mathcal{R}$	Requires Resolution	1.8	15.7

Table 2: DNS information origin by connection.

where neither endpoint is a well-known server. Further, this type of traffic often does not depend on DNS information, but rather obtains IP addresses of peers via initial bootstrapping information (e.g., ".torrent" files) and gossiping among the known peers.

The balance of the connections use a reserved port. Ports 443, 123 and 80 account for nearly all these connections. These instances seem to largely come from IP addresses hard-coded into software in small devices. For instance, we find over 23K unsuccessful NTP connections to a particular IP address. From web searches we believe this likely stems from TP-Link devices being hard-coded to contact a now-retired public NTP server. Additionally, we find nearly 3K connections that contact one of two Ooma NTP servers which seem to be hard-coded into Ooma's residential VoIP devices. In a final example, we find over 11K HTTPS connections to one of two AlarmNet IP addresses, which we believe to be part of a monitoring service for residential security alarms.

As we note in § 3, we have no visibility into encrypted DNS lookups. We believe that at the time our dataset was collected, encrypted DNS was not prevalent in the CCZ. First, we find no transactions in $\mathcal{N}$ using the standard DoT port (853). Second, we find that only 1.3% of all application transactions are both (i) unpaired with a visible DNS lookup and (ii) not peer-to-peer traffic. Even if all these connections pair with an encrypted DNS lookup—which is unlikely the case, given the above discussion and examples—the impact of encrypted DNS on our overall insights will be negligible.

5.2 No DNS Lookup Required

The second set of connections listed in Table 2 are those that use DNS information, but get their records from a local cache and, hence, do not block awaiting DNS. While all of these connections start at least 100 msec after the paired DNS lookup has finished (see § 4), we sub-divide them further by whether the connection is the first to use the paired DNS record ($\mathcal{P}$) or not ($\mathcal{LC}$), as follows.

Local Caching: The $\mathcal{LC}$ category listed in Table 2 shows that 42.9% of the connections in our dataset rely on DNS information from the local cache.[6] The $\mathcal{LC}$ connections both (i) come more than 100 msec after the corresponding DNS transaction finishes and (ii) leverage a paired DNS record that has been previously used. This second criteria means the information has been locally cached based on previous activity. These connections do not block waiting for DNS information and, therefore, have no attendant DNS costs.

Of the 4.8M $\mathcal{LC}$ connections, nearly 1.1M (22.2%) use outdated DNS information that should have been expunged from the cache based on the DNS record's TTL. This aligns with previous work that shows some residential networking gear does not respect the TTL [25] and hosts often use DNS bindings for longer than allowed by the TTL [28]. We find that almost 82% of the violations happen

[6]Given our vantage point outside the residences, we cannot determine whether the DNS information comes from the on-device stub resolver's cache or from a home router acting as a DNS forwarder. However, in both cases the information is close to the application and the retrieval delay should be nearly irrelevant.

more than 30 seconds after the expiration. The median and 90th percentile of the distribution are 890 seconds and nearly 19K seconds, respectively. Hence, a non-trivial part of the efficacy of the local cache is based on violating DNS records' TTL.

Prefetching: Most modern web browsers speculatively issue DNS requests of hostnames they encounter that may need resolution in the future [7]. For instance, if a web page has an advertisement with a link to "www.cnn.com" the browser will lookup "www.cnn.com" as soon as noticing the link so that, if needed,there will be no delay in determining the IP address to contact. We determine that a connection benefits from speculative DNS lookups if it both (*i*) starts 100 msec after the paired DNS lookup concludes and (*ii*) leverages a previously unused DNS lookup. We denote this as $\mathcal{P}$ on Table 2 and find that 7.8% of the connections benefit from prefetching. The $\mathcal{P}$ connections do not incur a DNS cost because the DNS lookup happens in the natural lag time between noticing the name and using the mapping. Without prefetching the application would be forced to block awaiting a DNS response.

The benefit of DNS prefetching comes with the cost of speculative lookups that go unused. We find almost 3.1M DNS transactions (or 37.8%) are not used to facilitate an application transaction. Given our vantage point, we cannot conclusively determine that these are all traditional prefetching. For instance, an application could trigger a DNS lookup with the intention of using the result only to terminate before doing so. However, we believe that these DNS transactions are largely speculative lookups to aid performance. If we assume all the unused DNS lookups were caused by prefetching, then 22.3% of the speculative lookups were ultimately used.

Similar to the $\mathcal{LC}$ connections, we find that 12.4% of the $\mathcal{P}$ connections use expired DNS records. The rate of using expired DNS records is roughly 10% less for $\mathcal{P}$ connections compared with $\mathcal{LC}$ connections. We believe this discrepancy stems from $\mathcal{LC}$ connections being to destinations the user accesses with some frequency—as indicated by $\mathcal{LC}$ connections not being the first connection to use a given lookup. Meanwhile, the lookups behind $\mathcal{P}$ connections are speculative and so their use depends on users clicking a link. This naturally happens closer in time to the speculative lookup—e.g., before the user moves on to another web page—and therefore more often happens within the TTL. Indeed, the median time between DNS lookup and use is 310 seconds for $\mathcal{P}$ connections and 1,033 seconds for $\mathcal{LC}$ connections.

5.3 DNS Lookup Required

The previous two subsections detail connections that do not block on DNS lookups. Now we turn to the last two lines on Table 2 which deal connections that block awaiting DNS lookups to complete. These connections start within 100 msec of the end of the DNS transaction. These two categories are the $\mathcal{SC}$ connections that use DNS records from a shared resolver's cache and the $\mathcal{R}$ connections that require retrieving information from authoritative servers.

Given that we do not have visibility into DNS requests sent from the shared resolvers, we use a heuristic that uses the duration of a DNS lookup to place each connection in the $\mathcal{SC}$ or $\mathcal{R}$ set. For all resolvers utilized by at least 1,000 connections we calculate the distribution of the duration of the DNS transactions. We assume the minimum of this distribution is the network RTT to the given resolver. If a DNS record is cached by the resolver, we expect the lookup duration to be roughly the same as the minimum duration. However, the transaction duration will be longer than the network RTT when the resolver requires information from one or more authoritative servers. The delay distributions for all the popular resolvers indeed show a mode around the minimum delay, which we then use to set a threshold for each resolver. We conclude that lookups that do not exceed the threshold are served from the shared resolver's cache and the remainder require communication with some number of authoritative DNS servers. The threshold varies by resolver and reflects a small amount of rounding—e.g., we find a minimum RTT of about 2 msec for the local ISP's resolvers and therefore we use a threshold of 5 msec. For transactions not destined for one of the popular shared resolvers we use a threshold of 5 msec. The thresholds are specific to the monitored network, but the process for determining the thresholds is general.

Note, all DNS records we classify as coming from the shared resolver's cache nearly must be correct since there just isn't time to contact an authoritative server. However, we could mistakenly classify a lookup as requiring communication with authoritative servers when the shared resolver has the needed information in its cache, but extra delays creep into the process—e.g., network jitter, retransmission delays or extra processing delays.

Shared Cache: Table 2 shows that the shared cache is leveraged by 26.3% of the connections in our dataset. This represents a cache hit rate of 62.6% of all connections that query a recursive resolver (i.e., the $\mathcal{SC}$ and $\mathcal{R}$ connections). I.e., even when a connection is blocked waiting on a DNS record not available locally, the majority of the time the record is in the shared resolver's cache and the delay to obtain it is low—i.e., at most 20 msec in our environment.

Resolution Required: The final line in Table 2 shows that 15.7% of the connections in our dataset require the shared resolver to contact one or more authoritative servers before the connection can start. From our monitoring vantage point, we cannot determine how many authoritative servers the resolver must contact for each transaction. However, presumably, even these transactions that we have labeled as cache misses within the shared resolver generally benefit from some amount of caching. For instance, many transactions likely depend on the shared resolver's cached copy of the ".com" record obtained from the root nameservers. Therefore, this category encompasses a fairly wide variety of behavior.

6 DNS PERFORMANCE IMPLICATIONS

We now turn from the origin of the DNS records to the impact of lookups on performance. The $\mathcal{N}$, $\mathcal{LC}$ and $\mathcal{P}$ connections—57.9% of the total connections—either do not require DNS information or leverage records that are on hand locally when the connection is instantiated. Therefore, these connections incur no direct DNS costs. Instead, in this section we focus on the $\mathcal{SC}$ and $\mathcal{R}$ connections.

The top plot in Figure 2 shows the distribution of the duration of the DNS lookups in the $\mathcal{SC}$ and $\mathcal{R}$ sets. The modes at lower durations represent cache hits with the different modes indicating different resolvers. For instance, the large mode around 2 msec stems from the local ISP's resovlers, while the small mode just under 10 msec stems from cache hits at Cloudflare's resolvers. The figure shows a median lookup time of 8.5 msec and a 75th percentile

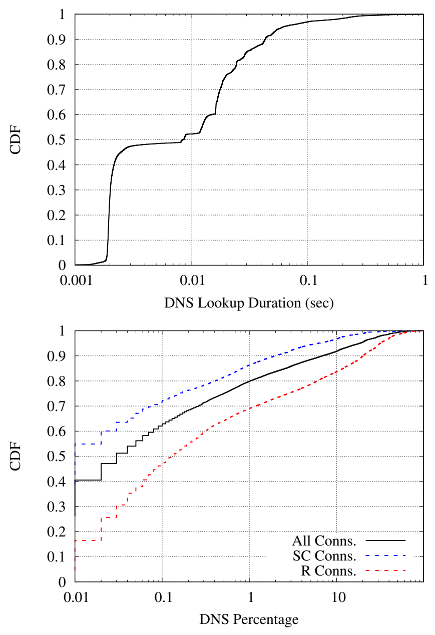

Figure 2: Dist. of DNS lookup delays (top) and DNS contribution to application transaction (bottom) for $\mathcal{SC}$ and $\mathcal{R}$.

of 20 msec. Further only 3.3% of the connections experience a DNS lookup delay over 100 msec. Our general takeaway is that in absolute terms DNS lookup durations are modest.

While the absolute delay is often small, we also aim to understand the delay in the context of the application transaction that depends on the DNS lookup. For each connection in $\mathcal{SC}$ and $\mathcal{R}$ we determine the total time T as the sum of the duration of the DNS lookup D and the duration of the application transfer A. The bottom plot in Figure 2 shows the distribution of DNS' percentage contribution to the total time (i.e., $\frac{100 \times D}{T}$). The plot shows that DNS' contribution is more than 1% for only 20% of all the transactions (black line). Further, in only 8% of the transactions do we find DNS contributes at least 10% of the overall duration. Unsurprisingly, the DNS portion of the transactions is generally less for the $\mathcal{SC}$ set (blue line) than for the $\mathcal{R}$ set (red line). However, even for the $\mathcal{R}$ case we find DNS contributes more than 1% in only 30% of the transactions.

Next, we pick two independent criteria that intuitively indicate the DNS cost is "*insignificant*": (*i*) an absolute lookup time of no more then 20 msec and (*ii*) a relative contribution of no more than 1%.[7] We find 64.0% of the $\mathcal{SC}$ and $\mathcal{R}$ transactions experience DNS lookups that are insignificant using *both criteria*. Also, we find that for 11.5% of the transactions the DNS lookup constitutes more than 1% of the transaction time, however, the absolute cost is no more than 20 msec. Similarly, we find that 15.9% of the transactions have an absolute DNS lookup time more than 20 msec, but the DNS lookup represents no more than 1% of the total transaction. This leaves 8.6% of the $\mathcal{SC}$ and $\mathcal{R}$ transactions that pay a significant DNS lookup cost—i.e., the delay is both (*i*) $>$ 20 msec and (*ii*) $>$ 1% of the overall transaction time. In terms of the overall set of connections this means only 3.6% suffer a significant DNS lookup delay.

[7]While we can quibble about the chosen constants, when run with alternate values—elided due to space—our analysis shows similar high-order insight.

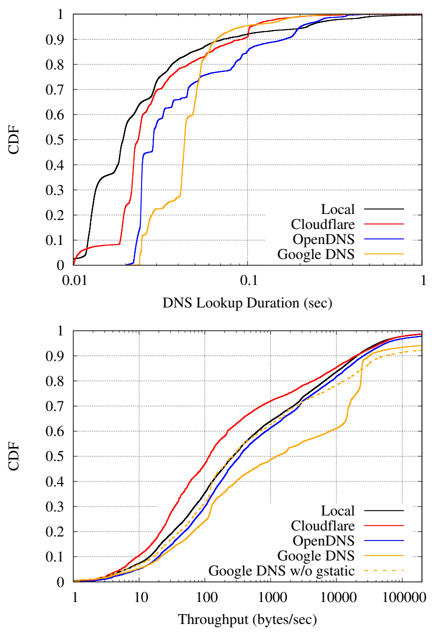

Figure 3: Dist. of DNS delays for $\mathcal{R}$ connections (top) and throughput for $\mathcal{SC}$ and $\mathcal{R}$ connections (bottom).

7 PERFORMANCE VS. RESOLVER

As we discuss in § 3, a handful of resolver platforms are used by at least 1% of the connections in our dataset (see Table 1). We now use our framework to perform an analysis of the relative performance of each platform across various metrics.

First, we consider the shared cache hit rate—i.e., the percentage of $\mathcal{SC}$ connections relative to all $\mathcal{SC}$ and $\mathcal{R}$ connections for each resolver platform. The hit rate varies considerably across resolver platforms, as follows: (*i*) Cloudflare, 83.6%, (*ii*) local ISP resolvers, 71.2%, (*iii*) OpenDNS, 58.8%, and (*iv*) Google, 23.0%.[8] All platforms except Google use their cache for the majority of the lookups.

We next turn to the duration of lookups to various resolver platforms. The lookup duration in the $\mathcal{SC}$ set are dictated by the RTT between the monitored clients and the resolver. Therefore, we focus on delays for the $\mathcal{R}$ set of lookups. The top plot in Figure 3 shows the distribution of the duration of lookups used by $\mathcal{R}$ connections. The plot shows that (naturally) the local ISP's resolver generally has the lowest delay, followed by Cloudflare and OpenDNS. The difference between the local ISP, Cloudflare and OpenDNS at the median point in the distribution can largely be explained by the RTT between the clients and the resolver. That is, these resolvers generally perform their lookups similarly, but their distance from the client is the factor in making them "better" or "worse". Google on the other hand has generally longer lookups than the other resolvers until the 75^{th} percentile. However, the tail of Google's distribution is generally shorter than the other platforms.

A final metric we use to assess resolvers is the performance of application transactions. Content Distribution Networks (CDNs) often

[8]Note, we only assess the first-level cache that is co-located with the resolver the client accesses. Some platforms may have second-level caches that provide the needed answers instead of relying on the authoritative servers, however, we cannot detect this from our vantage point.

convey which edge server clients should access via DNS responses. One of the pieces of information CDNs use to make this decision is where a client is located. Since clients rarely send DNS requests directly to the CDNs' authoritative servers, the CDNs leverage the location of the resolver as a proxy for a client's location. This means that the resolver can impact the quality of CDN decisions [19].

The bottom plot in Figure 3 shows the distribution of throughput for $\mathcal{SC}$ and $\mathcal{R}$ connections for each resolver platform. First, note that Google has two lines on the plot. We find that 23.5% of the connections that stem from lookups to Google's resolver are for "*connectivitycheck.gstatic.com*". This hostname is responsible for 0.3% of the connections that stem from non-Google resolvers. Looking up this hostname is part of the process Anroid devices use to detect captive portals. We removed the connections that use the "*connectivitycheck.gstatic.com*" hostname and include the dashed orange line to show the performance of Google without this artifact. Comparing the solid and dashed orange lines on the plot shows that these connections skew the distribution. The plot also shows that for 75% of connections Cloudflare-based connections have lower performance than the other three platforms—which are roughly equivalent. For the 25% tail of the distributions we find that (*i*) Cloudflare-based connections converge to being roughly equivalent with the local ISP's resolver and OpenDNS, while (*ii*) Google-based connections (dashed line) perform better than the others (corresponding to their lower delay shown in the top plot).

Ultimately, due to conflicting metrics we are unable to determine which resolver is "the best".

8 POSSIBLE DNS IMPROVEMENTS

In Table 2 we show that 42.1% of the connections in our dataset block awaiting DNS information. While we find in § 6 that blocking adds significant delay for only 3.6% of the connections, we now briefly explore two local mechanisms to further reduce DNS' cost.

A Whole-House Cache: The first improvement we study is use of a cache in each home's router. While many modern residential routers act as caching forwarders [25], those used in the CCZ do not. However, we take cases where multiple requests for the same record within the record's TTL as hints that multiple devices require the same record and a whole-house cache could be helpful.

We used the traffic in our dataset to simulate a whole-house cache for each house in the observed network. The connections in the $\mathcal{N}$, $\mathcal{LC}$ and $\mathcal{P}$ categories will not benefit from further caching, hence, we focus on the $\mathcal{SC}$ and $\mathcal{R}$ connections. We find that 9.8% of the connections in our dataset would move from $\mathcal{SC}$ or $\mathcal{R}$ to $\mathcal{LC}$—i.e., would benefit from a whole-house cache. We find that 22% of the $\mathcal{SC}$ connections and 25% of the $\mathcal{R}$ connections would see benefit from a whole-house cache—i.e., the savings is fairly uniform across the two sets of connections that block on DNS lookups.

Refreshing: Another local approach we investigate to aid performance is speculatively refreshing cache entries as they are about to expire in a whole-house cache. This is similar to browser prefetching in that we use a cue that a user might need a given record in the future so we request and cache the record a priori.

To study this, we built a simple trace-driven simulator that uses our traffic dataset as the workload. For each query retrieved over our dataset we use the maximum TTL in the observed responses as the authoritative TTL. This is a conservative approximation in that if the TTL we observe is lower than the actual authoritative TTL we will expire the entry too soon and see more work required to keep the entry fresh. Further, we do not refresh records with an authoritative TTL under 10 seconds as this is logistically problematic (more below). Finally, note, that some of the results from our simulations do not exactly match the analysis in the previous sections. This stems from our ideal use of the TTL, which does not always align with reality (e.g., see § 5.2).

	Standard	**Refresh All**
Conns.	10.4M	10.4M
DNS Lookups	8.4M	1.2B
Lookups/sec/house	0.2	25.2
Cache Hits	61.0%	96.6%
Cache Misses	39.0%	3.4%

Table 3: Efficacy of refreshing expiring names.

Table 3 shows the results of our simulations. First, note, we only use the 10.4M connections in our corpus that use DNS in some fashion—i.e., we do not include the $\mathcal{N}$ connections. The second column shows the performance of a standard cache that conducts DNS queries as needed. We find 61.0% of the connections are served from the cache. The number of lookups equates to an average of roughly 0.2 lookups per second across all the houses in our dataset.

The final column of Table 3 shows the performance of a whole-house cache that refreshes all previous lookups as their entries expire—for records with TTLs > 10 seconds. In this case, the cache hit rate is 96.6%, showing that name use is highly predictable by previous user behavior. Achieving this cache hit rate, however, takes roughly 1.2B additional DNS requests—or roughly 144x more than the standard cache. The average query rate is about 25 queries per second per house—an approximately 126x increase over the standard cache. While the cache hit rate increased by more than 50%, the query load seems impractical for most situations. Further, the query load will increase if we include names with lower TTLs. An open question for future work is whether we can design ways to achieve close to the 96.6% cache hit rate that is possible, while incurring costs that are commiserate with the standard cache.

9 SUMMARY

This paper provides a study of DNS use within the context of application traffic. We make several contributions, including (*i*) finding nearly 58% of connections do not block awaiting DNS lookups, (*ii*) determining that only 3.6% of the connections pay a significant DNS cost, and (*iii*) finding no clear cut winner across several resolver platforms. We believe our analysis adds to the significant existing body of work on understanding DNS.

ACKNOWLEDGMENTS

This work benefits from discussions with Misha Rabinovich, Kyle Schomp and Rami Al-Dalky over the years. The IMC reviewers and our shepherd, John Heidemann, provided many useful comments that improved the paper. This work was supported in part by National Science Foundation awards CNS-1647126 and CNS-1815876. Our thanks to all!

REFERENCES

[1] Rami Al-Dalky, Michael Rabinovich, and Mark Allman. 2018. Practical Challenge-Response for DNS. *ACM Computer Communication Review* 48, 3 (July 2018).

[2] Mark Allman. 2018. Comments on DNS Robustness. In *ACM Internet Measurement Conference.*

[3] Ignacio N. Bermudez, Marco Mellia, Maurizio M. Munafo, Ram Keralapura, and Antonio Nucci. 2012. DNS to the Rescue: Discerning Content and Services in a Tangled Web. In *Internet Measurement Conference.*

[4] N. Brownlee, k. claffy, and E. Nemeth. 2001. DNS Measurements at a Root Server. In *IEEE Global Telecommunications Conference (GLOBECOM).*

[5] Tom Callahan, Mark Allman, and Michael Rabinovich. 2013. On Modern DNS Behavior and Properties. *ACM Computer Communication Review* 43, 3 (July 2013).

[6] S. Castro, D. Wessels, M. Fomenkov, and k. claffy. 2008. A Day at the Root of the Internet. *ACM SIGCOMM Computer Communication Review (CCR)* 38, 5 (Oct 2008).

[7] E. Cohen and H. Kaplan. 2003. Proactive Caching of DNS Records: Addressing a Performance Bottleneck. *Computer Networks* 41, 6 (2003).

[8] P. Hoffman and P. McManus. 2018. DNS Queries Over HTTPS (DoH). RFC 8484.

[9] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. 2016. Specification For DNS Over Transport Layer Security (TLS). RFC 7858.

[10] J. Iyengar and M. Thomson. 2020. QUIC: A UDP-Based Multiplexed and Secure Transport. Internet-Draft draft-ietf-quic-transport-30.txt (work in progress).

[11] Ben Jones, Nick Feamster, Vern Paxson, Nicholas Weaver, and Mark Allman. 2016. Detecting DNS Root Manipulation. In *Passive and Active Measurement Conference.*

[12] Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. 2001. DNS Performance and the Effectiveness of Caching. In *ACM SIGCOMM Internet Measurement Workshop.*

[13] Amit Klein, Haya Shulman, and Michael Waidner. 2017. Internet-Wide Study of DNS Cache Injections. In *Proc. of IEEE InfoCom.*

[14] Matthew Lentz, Dave Levin, Jason Castonguay, Neil Spring, and Bobby Bhattacharjee. 2013. D-mystifying the d-root Address Change. In *ACM Internet Measurement Conference.*

[15] Chaoyi Lu, Baojun Liu, Zhou Li, Shuang Hao, Haixin Duan, Mingming Zhang, Chunying Leng, Ying Liu, Zaifeng Zhang, and Jianping Wu. 2019. An End-to-End, Large-Scale Measurement of DNS-over-Encryption: How Far Have We Come?. In *ACM SIGCOMM Internet Measurement Conference.*

[16] M. M. Almeida, A. Finamore, D. Perino, N. Vallina-Rodriguez, and M. Varvello. 2017. Dissecting DNS Stakeholders in Mobile Networks. In *ACM CoNext.*

[17] P.V. Mockapetris. 1987. Domain Names - Concepts And Facilities. RFC 1034.

[18] Giovane C. M. Moura, John Heidemann, Ricardo Schmidt, and Wes Hardaker. 2019. Cache Me If You Can: Effects of DNS Time-to-Live. In *ACM SIGCOMM Internet Measurement Conference.*

[19] J. Otto, M. Schez, J. Rula, and F. Bustamante. 2012. Content Delivery and the Natural Evolution of DNS: Remote DNS Trends, Performance Issues and Alternative Solutions. In *ACM SIGCOMM Internet Measurement Conference.*

[20] Craig Partridge and Mark Allman. 2016. Addressing Ethical Considerations in Network Measurement Papers. *Commun. ACM* 59, 10 (Oct. 2016).

[21] Vern Paxson. 1999. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks* 31, 23-24 (Dec. 1999).

[22] Matt Sargent and Mark Allman. 2014. Performance Within A Fiber-To-The-Home Network. *ACM Computer Communication Review* 44, 3 (July 2014).

[23] Matt Sargent, Brian Stack, Tom Dooner, and Mark Allman. 2012. *A First Look at 1 Gbps Fiber-To-The-Home Traffic.* Technical Report 12-009. International Computer Science Institute.

[24] Kyle Schomp, Mark Allman, and Michael Rabinovich. 2014. DNS Resolvers Considered Harmful. In *ACM SIGCOMM HotNets.*

[25] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. 2013. On Measuring the Client-Side DNS Infrastructure. In *ACM Internet Measurement Conference.*

[26] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. 2014. Assessing DNS Vulnerability to Record Injection. In *Passive and Active Measurement Conference.*

[27] Kyle Schomp, Michael Rabinovich, and Mark Allman. 2016. Towards a Model of DNS Client Behavior. In *Passive and Active Measurement Conference.*

[28] Craig Shue, Andrew Kalafut, Mark Allman, and Curtis Taylor. 2012. On Building Inexpensive Network Capabilities. *ACM Computer Communication Review* 42, 2 (April 2012).

[29] Craig Shue, Andrew Kalafut, and Minaxi Gupta. 2007. The Web is Smaller than it Seems. In *ACM Internet Measurement Conference.*

[30] Srikanth Sundaresan, Nick Feamster, Renata Teixeira, and Nazanin Magharei. 2013. Measuring and Mitigating Web Performance Bottlenecks in Broadband Access Networks. In *ACM Internet Measurement Conference.*

[31] Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. 2014. DNSSEC and Its Potential for DDoS Attacks: A Comprehensive Measurement Study. In *ACM Internet Measurement Conference.*

[32] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2013. Demystifying Page Load Performance with WProf. In *Proc. of NSDI.*

[33] D. Wessels and M. Fomenkov. 2003. Wow, That's A Lot of Packets. In *Passive and Active Network Measurement Workshop (PAM).*

A ETHICS

The dataset we use for this paper is based on observing a real user population and, as such, there are privacy concerns with the data. While we cannot guarantee no harm ever comes from this dataset, we have taken a number of precautions to conduct the work in an ethical manner, including:

- The monitored FTTH network is adjacent to Case Western Reserve University and was setup for the explicit purpose of experimenting with giving large capacity to residential users. Each house connected to the network has explicitly agreed to participate in the experimental program.
- Case's Institutional Research Board (IRB) has reviewed our data collection, storage and use processes and ultimately has classified our research as not being human subject research. That is, our research focus is on the system and not the people. Of course, we do not take this finding as carte blanche, but rather understand that the dataset must be treated with care as it is in fact sensitive.
- We use only passively collected data in this paper. Therefore, our experiments cannot cause direct harm, but instead the data represents only "potential harm", as outlined in [20]. In other words, we do not make any actual changes to the network, we only watch what naturally happens. That said, we treat the recorded data as highly sensitive such that any potential harm we record does not turn into actual harm.
- Access to the actual network monitor we use to gather data is tightly controlled. It is generally available only to faculty and senior research scientists. Occasionally, a student is given temporary access to debug an issue. This latter did not happen for the data gathered for this paper.
- The body of this paper uses only aggregate numbers and we do not provide enough information to identify any user of the monitored network. While it may be possible to identify people with the raw data, we have not done so in our investigation.
- From the traffic patterns we observe, we believe all houses likely have multiple devices and multiple users. The gateway provided for each residence in the experimental network does Network Address Translation and this makes it even more difficult to readily identify actual users from the recorded traffic.
- All data collected is kept encrypted. Access is given only (*i*) to those with a direct need and (*ii*) to the narrowest possible set of data for a given project.
- For the application transactions (web, email, etc.) we do not record the payload of the transactions. We use Bro's [21] standard connection summary policy file to record various information about the connections (endpoints, start time, duration, bytes transmitted, etc.).
- For DNS transactions we use Bro's [21] DNS policy script to record traffic. We do not collect DNS payloads as they appear

on the wire, but the summaries do contain query strings and the resource records from responses.

- While we could have anonymized portions of the traffic—e.g., IP addresses, query strings—described in the last two bullets before storing the data, we chose not to do so. While most of the analysis in this paper could have been conducted on anonymous data, we believe that such anonymization unduly degrades the research value of the data.
  E.g., without the real IP addresses of the DNS resolvers the quality of the analysis in § 7 would have been degraded—or, perhaps, not possible. For instance, each resolver platform uses multiple IP addresses and it would have been exceedingly difficult to tease apart Google and OpenDNS because they both had an RTT of roughly 20 msec from the monitored network.
  As we have done in much previous work, we put our trust in limiting access to the data and working in a disciplined way that calls for only looking at only what we need to complete a given analysis.

Ultimately, we believe the ethical decisions we made for our work are consistent with much previous work in the community and align with the community's (unwritten[9]) norms.

[9] ...alas...