Towards a Model of DNS Client Behavior*

Kyle Schomp[†], Michael Rabinovich[†], Mark Allman[‡]

[†]Case Western Reserve University, Cleveland, OH, USA
[‡]International Computer Science Institute, Berkeley, CA, USA

Abstract. The Domain Name System (DNS) is a critical component of the Internet infrastructure as it maps human-readable hostnames into the IP addresses the network uses to route traffic. Yet, the DNS behavior of individual clients is not well understood. In this paper, we present a characterization of DNS clients with an eye towards developing an analytical model of client interaction with the larger DNS ecosystem. While this is initial work and we do not arrive at a DNS workload model, we highlight a variety of behaviors and characteristics that enhance our mental models of how DNS operates and move us towards an analytical model of client-side DNS operation.

1 Introduction

The modern Internet relies on the Domain Name System (DNS) for two main functions. First, the DNS allows people to leverage human-friendly hostnames (e.g., "www.cnn.com") instead of obtuse IP addresses to identify a host. Second, hostnames provide a layer of abstraction such that the IP address assigned to a hostname can vary over time. In particular, Content Distribution Networks (CDNs) employ this late binding to direct users to the best content replica. Previous work shows that DNS lookups precede over 60% of TCP connections [14]. As a result, individual clients issue large numbers of DNS queries. Yet, our understanding of DNS query streams is largely based on aggregate populations of clients—e.g., at an organizational [6] or residential level [3]—leaving our knowledge of individual client behavior limited.

This paper represents an initial step towards understanding individual client DNS behavior. We monitor DNS transactions between a population of thousands of clients and their local resolver such that we are able to directly tie lookups to individual clients. Our ultimate goal is an analytical model of DNS client behavior that can be used for everything from workload generation to resource provisioning to anomaly detection. In this paper we provide a characterization of DNS behavior along the dimensions our model will ultimately cover and also anecdotally show promising modeling approaches.

Note, one view holds that DNS is a "side service" and should not be directly modeled, but rather can be well understood by deriving the DNS workload from applications such as web browsing and email transmission. However, deriving a DNS workload from application behavior is at best difficult because (i) client

* This work was funded in part by NSF grant CNS-1213157.

caching policies impact what DNS queries are actually sent in response to an application event, (*ii*) some applications selectively use pre-fetching to lookup names before they are needed and (*iii*) such a derivation would entail understanding many applications to pull together a reasonable DNS workload. Therefore, we take the approach that focusing on the DNS traffic itself is the most tractable way to understand—and eventually model—name lookups.

To motivate the need for a model, we provide an exemplar from our previous work. In [14], we propose that clients should directly resolve hostnames instead of using a recursive resolver. Ideally, an evaluation of this end system-based mechanism would be conducted in the context of end systems themselves. However, the best data we could obtain was at the level of individual households—which we know to include multiple hosts behind a NAT. Therefore, the results of our trace-driven simulations are at best an approximation of the impact of the mechanism we were investigating. Our results would have been more precise had we been able to leverage a model of individual client DNS behavior.

Broadly, the remainder of this paper follows the contours of what a model would capture. We first focus on understanding the nature of the clients themselves in §3, finding that while most are traditional user-facing devices, there are others that interact with the DNS in distinct ways. Next we observe in §4 that DNS queries often occur closely-spaced in time—e.g., driven by loading objects for a single web page from disparate servers—and therefore we develop a method to gather together queries into clusters. We then assess the number and spacing of queries in §5 and finally tackle the patterns in what hostnames individual clients lookup in §6. We find that clients have fairly distinct "working sets" of names, and also that hostname popularity has power law properties.

2 Dataset

Our dataset comes from two packet taps at Case Western Reserve University (CWRU) that monitor the links connecting the two data centers that house all five of the University's DNS resolvers—i.e., between client devices and their recursive DNS resolvers. We collect full payload packet traces of all UDP traffic involving port 53 (the default DNS port). The campus wireless network situates client devices behind NATs and therefore we cannot isolate DNS traffic to individual clients. Hence, we do not consider this traffic in our study (although, future work remains to better understand DNS usage on mobile devices). The University Acceptable Use Policy prohibits the use of NAT on its wired networks while offering wireless access throughout the campus, and therefore we believe the traffic we capture from the wired network does represent individual clients. Our dataset includes all DNS traffic from two separate weeks and is partitioned by client location—in the residential or office portions of the network. Details of the datasets are given in Table 1 including the number of queries, the number of clients that issue those queries, and the number of hostnames queried.

Validation: During the February data collection, we collect query logs from the five campus DNS resolvers to validate our datasets[1]. Comparing the packet

[1] We prefer traces over logs due to the better timestamp resolution (msec vs. sec).

Dataset	Dates	Queries	Clients	Hostnames
Feb:Residential	Feb. 26 - Mar. 4	32.5M	1359 (IPs)	652K
Feb:Residential (filter)	Feb. 26-27, Mar. 2-4	16.4M	1262 (MACs)	505K
Feb:Residential:Users		15.3M	1033	499K
Feb:Residential:Others		1.11M	229	7.94K
Feb:Office	Feb. 26 - Mar. 4	232M	8770 (IPs)	1.98M
Feb:Office (filter)	Feb. 26-27, Mar. 2-4	143M	8690 (MACs)	1.87M
Feb:Office:Users		118M	5986	1.52M
Feb:Office:Others		25.0M	2704	158K
Jun:Residential	Jun. 23 - Jun. 29	11.7M	345 (IPs)	140K
Jun:Residential (filter)	Jun. 23-26, 29	6.22M	334 (MACs)	120K
Jun:Residential:Users		5.81M	204	116K
Jun:Residential:Others		408K	130	4.13K
Jun:Office	Jun. 23 - Jun. 29	245M	8335 (IPs)	1.61M
Jun:Office (filter)	Jun. 23-26, 29	133M	8286 (MACs)	1.52M
Jun:Office:Users		108M	5495	1.42M
Jun:Office:Others		25.0M	2791	63.1K

Table 1. Details of the datasets used in this study.

traces and logs we find a 0.6% and 1.8% loss rates in the Feb:Residential and Feb:Office datasets, respectively. We believe these losses are an artifact of our measurement apparatus given that the loss rate is correlated with traffic volume.

Tracking Clients: We aim to track individual clients in the face of dynamic address assignment. Simultaneously with the DNS packet trace, we gather logs from the University's three DHCP servers. Therefore, we can track DNS activity based on MAC addresses. Note, we could not map 1.3% of the queries across our datasets to a MAC address because the source IP address in the query never appears in the DHCP logs. These likely represent static IP address allocations. Further, without any DHCP assignments we are confident that these IPs represent a single host.

Filtering Datasets: We find two anomalies that skew the data in ways that are not indicative of user behavior. First, we find roughly 25% of the queries request the TXT record for *debug.opendns.com*. (The next most popular record represents less than 1% of the lookups!) We find this query is not in response to users' actions, but is automatically issued to determine whether the client is using the OpenDNS resolver (indicated in the answer) [1]. We observe 298 clients querying this record, which we assume use OpenDNS on other networks or used OpenDNS in the past. We remove these queries from further analysis. The second anomaly involves 18 clients whose prominent behavior is to query for *debug.opendns.com* and other domains repeatedly without evidence of accomplishing much work. The campus information technology department verified that these clients serve an operational purpose and are not user-facing devices. Therefore, we remove the 18 clients as they are likely unique to this network and do not represent users. We do not attempt to further filter misbehaving hosts—e.g., infected or misconfigured hosts—as we consider them part of the DNS workload (e.g., since a resolver would be required to cope with their requests).

Timeframe: To more directly compare residential and office settings we exclude Saturday and Sunday from our datasets.

Table 1 shows the magnitude of our filtering. We find commonality across the partitions of the data, so we focus on the Feb:Residential:Users dataset for conciseness and discuss how other datasets differ as appropriate.

Marker	Clients	%
All	1262	100%
Google analytics	983	78%
Search engine	1010	80%
Google	1006	80%
Any other	602	48%
Gmail	881	70%
LDAP Login	840	66%
Any	1033	82%

Table 2. Feb:Residential clients that fit markers for general purpose devices.

3 Identifying Types of Clients

Since our focus is on characterizing general purpose user-facing devices, we aim to separate them from other types of end systems. We expect general-purpose systems are involved in tasks, such as (*i*) web browsing, (*ii*) accessing search engines, (*iii*) using email, and (*iv*) conducting institutional-specific tasks[2]. Therefore, we develop the following markers to identify general-purpose hosts:

Browsing: A large number of web sites embed Google Analytics [8] in their pages, thus there is a high likelihood that regular users will query for Google Analytics hostnames on occasion.

Searching: We detect web search activity via DNS queries for the largest search engines: Google, Yahoo, Bing, AOL, Ask, DuckDuckGo, Altavista, Baidu, Lycos, Excite, Naver, and Yandex.

Email: CWRU uses Google to manage campus email and therefore we use queries for "mail.google.com" to indicate email use.

Institutional-Specific Tasks: CWRU uses a single sign-on system for authenticating users before they perform a variety of tasks and therefore we use queries for the corresponding hostname as indicative of user behavior.

Table 2 shows the breakdown of the clients in the Feb:Residential dataset. Of the 1,262 clients we identify 1,033 as user-facing based on at least one of the above markers. Intuitively we expect that multiple markers likely apply to most general purpose systems and in fact we find at least two markers apply to 991 of the clients in our dataset. Results for our other datasets are similar.

We next turn to the 229 clients ($\approx 18\%$) that do not match any of our markers for user-facing clients. To better understand these clients we aggregate them based on the vendor portion of their MAC addresses. First, we find a set of vendors and query streams that indicate special-purpose devices: (*i*) 48 Microsoft devices that query for names within the *xboxlive.com* domain, which we conclude are Xbox gaming consoles, (*ii*) 33 Sony devices that query for names within the *playstation.net* domain, which we conclude are Sony Playstation gaming consoles, (*iii*) 16 Apple devices that have an average of 11K queries—representing 96% of their lookups—for the *apple.com* domain, even though the average across all devices that lookup an *apple.com* name is 262 queries, which we conclude are Apple TV devices and (*iv*) 7 Linksys devices that issue queries for *esuds.usatech.com*, which we conclude are transaction systems attached to the laundry machines in the residence halls (!).

[2] In our case, this is campus-life tasks, e.g., checking the course materials portal.

In addition to these, we find devices that we cannot pinpoint explicitly, but do not in fact seem to be general-purpose client systems. We find 41 Dell devices that differ from the larger population of hosts in that they query for more PTR records than A records. A potential explanation is that these devices are servers obtaining hostnames for clients that connect to them (e.g., as part of *sshd*'s verification steps or to log client connects). We also identify 12 Kyocera devices that issue queries for only the campus NTP and SMTP servers. We conclude that these are copy machines that also offer emailing of scanned documents.

For the IP addresses that do not appear in the DHCP logs (i.e., addresses statically configured on the hosts), we cannot obtain a vendor ID. However, we note that 97% of the queries and 96% of the unique domain names from these machines involve CWRU domains and therefore we conclude that they serve some administrative function and are not general purpose clients. The remaining 61 devices are distributed among 42 hardware vendors. In the remainder of the paper we will consider the general purpose clients (Users) and the special purpose clients (Others) separately, as we detail in Table 1. We find that our high-level observations hold across all of the Users datasets, and thus present results for the Feb:Residential:Users dataset only.

4 Query Clusters

Applications often call for multiple DNS queries in rapid succession—e.g., as part of loading all objects on a web page, or prefetching names for links users may click. In this section, we quantify this behavior using the DBSCAN algorithm [4] to construct clusters of DNS queries that likely share an application event. The DBSCAN algorithm uses two parameters to form clusters: a minimum cluster size M and a distance ε that controls the addition of samples to a cluster. We use the absolute difference in the query timestamps as the distance metric. Our first task is to choose suitable parameters. Our strategy is to start with a range of parameters and determine whether there is a point of convergence where the results of clustering do not change greatly with the parameters. Based on the strategy in [4], we start with an M range of 3–6 and an ε range of 0.5–5 seconds—note that $M = 2$ simplifies to threshold based clustering, but does not produce a point of convergence. We find that 96% of the clusters we identify with $M = 6$ are exactly found when $M = 3$ and hence at $M = 3$ we have converged on a reasonably stable answer which we use in the subsequent analysis. Additionally, we find that for $\varepsilon \in [2.5, 5]$, the total number of clusters, the distribution of cluster sizes, and the assignment of queries to clusters remain similar irrespective of ε value and therefore use $\varepsilon = 2.5$ seconds in our analysis. We define the first DNS query per cluster as the *root* and all subsequent queries in the cluster as *dependents*. In the Feb:Residential:Users dataset, we find 1M clusters that encompass 80% of the roughly 15M queries in the dataset.

To validate the clustering algorithm we first inspect the 67K unique hostnames the algorithm labels as noise. We find a variety of hostnames with the most frequent being: WPAD [7] queries for discovering proxies, Google Mail and Google Docs, software update polling (e.g., McAfee and Symantec), heartbeat signals for gaming applications (e.g., Origin, Steam, Blizzard, Riot), video

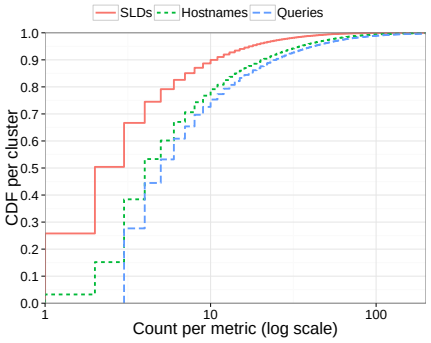

Fig. 1. Number of queries, hostnames, and SLDs per cluster.

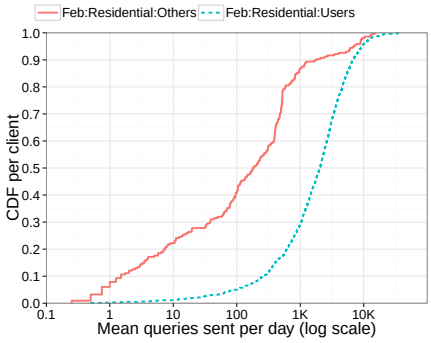

Fig. 2. Queries issued by each client per day.

streaming (e.g., Netflix, YouTube, Twitch), and the Network Time Protocol (NTP). All of these names can intuitively come from applications that require only sporadic DNS queries, as they are either making quick checks every once in a while, or are using long-lived sessions that leverage DNS only when starting.

To validate the clusters themselves, we observe that there are frequently occurring roots. Indeed, the 1M clusters have only 72K unique roots, with the 100 most frequently occurring roots accounting for 395K (40%) of the clusters. Further, the 100 most popular roots include popular web sites (e.g., *www.facebook.com*, *www.google.com*). These are the type of names we would expect to be roots in the context of web browsing. Another common root is *safebrowsing.google.com* [9], a blacklist directory used by some web browsers to determine if a given web site is safe to retrieve. This is a distinctly different type of root than a popular web site because the root is not directly related to the dependents by the page content, but rather via a process running on the clients. This in some sense means SafeBrowsing-based clusters have two roots. While use of SafeBrowsing is fairly common in our dataset, we do not find additional prevalent cases of this "two roots" phenomenon. From a modeling standpoint we have not yet determined whether "two roots" clusters would need special treatment.

Figure 1 shows the distribution of queries per cluster. While the majority of clusters are small, there are relatively few large clusters. We find that 90% of clusters contain at most 26 queries for at most 22 hostnames. Additionally, we find 90% of the clusters encompass at most 10 SLDs. The largest cluster spans 95 seconds and consists of 9,366 queries for names that match to the 3^{rd} level label. The second largest cluster consists of 6,211 queries for *myapps.developer.ubuntu.com*—which is likely a Ubuntu bug.

5 Query Timing

Next we tackle the question of when and how many queries clients issue. We begin with the distribution of the average number of queries that clients issue per day,

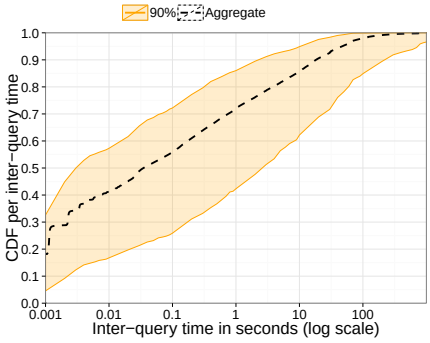

Fig. 3. Time between queries from the same client in aggregate and per client.

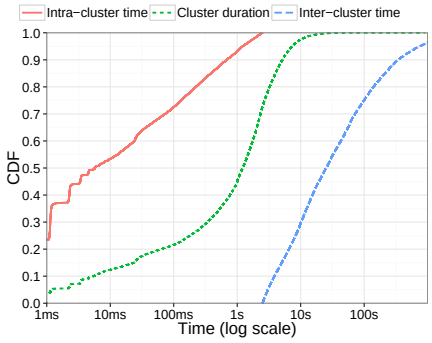

Fig. 4. Duration of clusters, inter-cluster query time and intra-cluster query time.

as given in Figure 2. We find that clients in Users issue 2K lookups per day at the median and 90% of clients in Users issue less than 6.7K queries per day. The Others datasets show greater variability where relatively few clients generate the lion's share of queries—i.e., the top 5% of clients produce roughly as many total DNS queries per day as the bottom 95% in the Feb:Residential:Others dataset.

A related metric is the time between subsequent queries from the same client, or inter-query times. Figure 3 shows the distribution of the inter-query times. The "Aggregate" line shows the distribution across all clients. The area "90%" shows the range within which 90% of the individual client inter-query time distributions fall. The majority of inter-query times are short, with 50% of lookups occurring within 34 milliseconds of the previous query. However, we also find a heavy tail, with 0.1% of inter-query times being over 25 minutes. Intuitively, long inter-query times represent off periods when the client's user is away from the keyboard (e.g., asleep or at class). The Others datasets show wide ranging behavior suggesting that they are less amenable to succinct description in an aggregate model.

For the Users dataset, we are able to model the aggregate inter-query time distribution using the Weibull distribution for the body and the Pareto distribution for the heavy tail. We find that partitioning the data at an inter-query time of 22 seconds minimizes the mean squared error between the data and the two analytical distributions. Next, we fit the analytical distributions—split at 22 seconds—to each of the individual client inter-query time distributions. We find that while the parameters vary per client, the empirical data is well represented by the analytical models as the mean squared error for 90% of clients is less than 0.0014. *Thus, parameters for a model of query inter-arrivals will vary per client, but the distribution is invariant.*

Next, we move from focusing on individual lookups to focusing on timing related to the 1M lookup clusters that encompass 12M (80%) of the queries in our dataset (see §4). Figure 4 shows our results. The "Intra-cluster time" line shows the distribution of the time between successive queries within the same cluster. This time is bounded to $\varepsilon = 2.5$ seconds by construction, but over 90% of the inter-arrivals are less than 1 second. On the other hand, the line "Inter-cluster

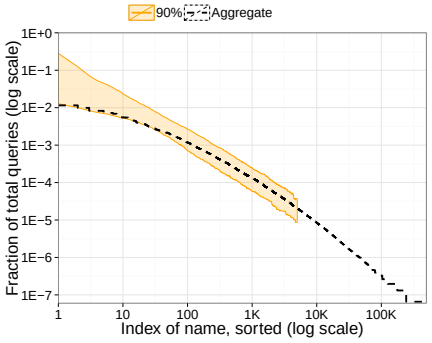

Fig. 5. Fraction of queries issued for each hostname per client.

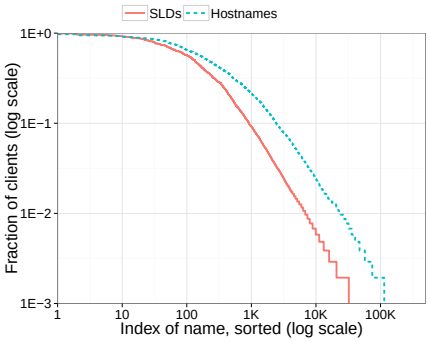

Fig. 6. Fraction of clients issuing queries for each hostname and SLD.

time" shows the time between the last query of a cluster and the first query of the next cluster. Again, most clusters are separated from each other by much more than ε time, the minimum separation by construction. The line "Cluster duration" shows the time between the first and last query in each cluster. Most clusters are short, with 99% less than 18 seconds. Additionally, we find that most of client DNS traffic occurs in short clusters: 50% of clustered queries belong to clusters with duration less than 4.6 seconds and 90% are in clusters with duration less than 20 seconds. For the Others datasets, a smaller percentage of DNS queries occur in clusters—e.g., 60% in the Feb:Residential:Others dataset.

6 Query Targets

Finally, we tackle the queries themselves including relationships between queries.

Popularity of Names: We analyze the popularity of hostnames using two methods—how often the name is queried across the dataset and how many clients query for it. Figure 5 shows the fraction of queries for each hostname (with the hostnames sorted by decreasing popularity) in the Feb:Residential:Users dataset. Per §5, we plot the aggregate distribution and a range that encompasses 90% of the individual client distributions. Of the 499K unique hostnames within our dataset, 256K (51%) are looked up only once. Meanwhile, the top 100 hostnames account for 28% of DNS queries. Figure 6 shows the fraction of clients that query for each name. We find that 77% of hostnames are queried by only a single client. However, over 90% of the clients look up the 14 most popular hostnames. Additionally, 13 of these hostnames are Google services and the remaining one is *www.facebook.com*. The plot shows similar results for second-level domains (SLDs), where 66% of the SLDs are looked up by a single client.

The distributions of both queries per name and clients per name demonstrate power law behavior in the tail. Interestingly, the Pearson correlation between these two metrics—popularity by queries and popularity by clients—is only 0.54 indicating that a domain name with many queries is not necessarily queried by a large fraction of the client population and vice versa. As an example, *update-keepalive.mcafee.com* is the 19^{th} most queried hostname but is only queried by

8.1% of the clients. At the same time, 55% of the clients query for *s2.symcb.com*, but in terms of total queries this hostname ranks as only the 1215^{th} most popular. This phenomenon may be partially explained by differences in TTL. The record for *s2.symcb.com* has a one hour TTL—limiting the query frequency. Meanwhile, *updatekeepalive.mcafee.com* has a 1 minute TTL. Given this short TTL and that the name implies polling activity, the large numbers of queries from a given client is unsurprising. Thus, a model of DNS client behavior must account for the popularity of hostnames in terms of both queries and clients.

The heavy tails of the popularity distributions represent a large fraction of DNS transactions. However, we cannot disregard unpopular names—even those queried just once—because together they are responsible for the majority of DNS activity therefore impacting the entire DNS ecosystem (e.g., cache behavior).

Co-occurrence Name Relationships: In addition to understanding popularity, we next assess the relationships between names, as these have implications on how to model client behavior. The crucial relationship between two names that we seek to quantify is frequent querying for the pair together. We begin with the request clusters (§4) and leverage the intuition that the first query within a cluster triggers the subsequent queries in the cluster and is therefore the *root* lookup. This follows from the structure of modern web pages, with a container page calling for additional objects from a variety of servers—e.g., an average web page uses objects from 16 different hostnames [10].

Finding co-occurrence is complicated due to client caching. That is, we cannot expect to see the entire set of dependent lookups each time we observe some root lookup. Our methodology for detecting co-occurrence is as follows. First, we define $clusters(r)$ as the number of clusters with r as the root across our dataset and $pairs(r, d)$ as the number of clusters with root r that include dependent d. Second, we limit our analysis to the case when $clusters(r) \geq 10$ to reduce the potential for false positive relationships based on too few samples. In the Feb:Residential:Users dataset, we find 7.1K (9.9%) of the clusters meet these criteria. Within these clusters we find 7.5M dependent queries and 2.2M unique (r, d) pairs. Third, for each pair (r, d), we compute the co-occurrence as $C = pairs(r, d)/clusters(r)$—i.e., the fraction of the clusters with root r that include d. Co-occurrence of most pairs is low with 2.0M (93%) pairs having a C much less than 0.1. We focus on the 78K pairs that have high C—greater than 0.2. These pairs include 98% of the roots we identify, i.e., nearly all roots have at least one dependent with which they co-occur frequently. Also, these pairs comprise 28% of the 7.5M dependent queries we study.

We note that intuitively dependent names could be expected to share labels with their roots—e.g., *www.facebook.com* and *star.c10r.facebook.com*—and this could be a further way to assess co-occurrence. However, we find that only 27% of the pairs within clusters with co-occurrence of at least 0.2 share the same SLD and 11% share the 3rd level label as the cluster root. This suggests that while not rare, counting on co-occurring names to be from the same zone to build clusters is dubious. As an extreme example, Google Analytics is a dependent of 1,049 unique cluster roots, most of which are not Google names.

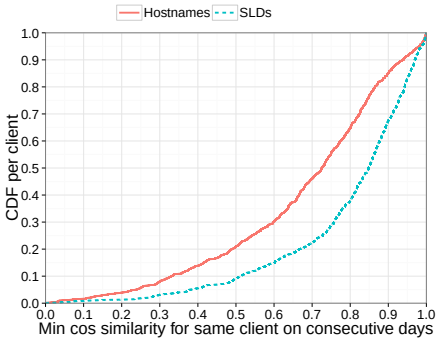

Fig. 7. Cosine similarity between the query vectors for the same client.

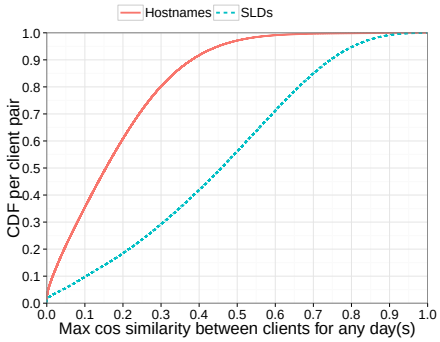

Fig. 8. Cosine similarity between the query vectors for different clients.

Finally, we cannot test the majority of the clusters and pairs for co-occurrence because of limited samples. However, we hypothesize that our results apply to all clusters. We note that the distribution of the number of queries per cluster in Figure 1 is similar to the distribution of the number of dependents per root where the co-occurrence fraction is greater than 0.2. Combining our observations that 80% of queries occur in clusters, 28% of the dependent queries within clusters have high co-occurrence with the root, and the average cluster has 1 root and 10 dependents, we estimate that at a minimum $80 * 0.28 * 10/11 = 20\%$ of DNS queries are driven by co-occurrence relationships. *We conclude that co-occurrence relationships are common, though the relationships do not always manifest as requests on the wire due to caching.*

Temporal Locality: We next explore how the set of names a client queries changes over time. As a foundation, we construct a vector $V_{c,d}$ for each client c and each day d in our dataset, which represents the fraction of lookups for each name we observe in our dataset. Specifically, we start from an alphabetically ordered list of all hostnames looked up across all clients in our dataset, N. We initially set each $V_{c,d}$ to a vector of $|N|$ zeros. We then iterate through N and set the corresponding position in each $V_{c,d}$ as the total number of queries client c issues for name N_i on day d divided by the total number of queries c issues on day d. Thus, an example $V_{c,d}$ would be $< 0, 0.25, 0, 0.5, 0.25 >$ in the case where there are five total names in the dataset and on day d the client queries for the second name once, the fourth name twice and the fifth name once. We repeat this process using only the SLDs from each query, as well.

We first investigate whether clients' queries tend to remain stable across days in the dataset. For this, we compute the minimum cosine similarity of the query vectors for each client across all pairs of consecutive days. Figure 7 shows the distribution of minimum cosine similarity per client in the Feb:Residential:Users dataset. In general, the cosine similarity values are high—greater than 0.5 for 80% of clients for unique hostnames—indicating that clients query for a similar set of names in similar relative frequencies across days. Given this result, it is unsurprising that the figure also shows high similarity across SLDs.

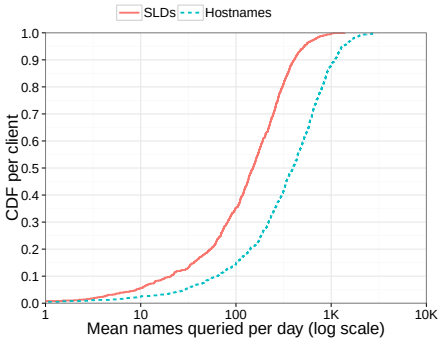

Fig. 9. Mean hostnames and SLDs queried by each client per day.

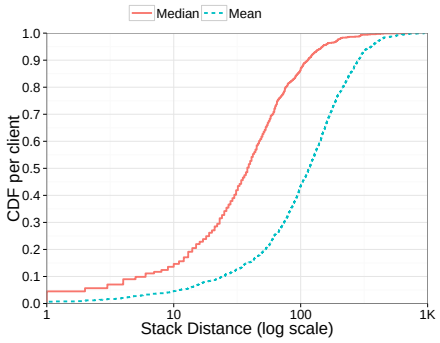

Fig. 10. Mean and median stack distance for each client.

Next we assess whether different clients query for similar sets of names. We compute the cosine similarity across all pairs of clients and for all days of our dataset. Figure 8 shows the distribution of the maximum similarity per client pair from any day. When considering hostnames, we find lower similarity values than when focusing on a single client—with only 3% showing similarity of at least 0.5—showing that each client queries for a fairly distinct set of hostnames. The similarity between clients is also low for sets of SLDs, with 55% of the pairs showing a maximum similarity less than 0.5. Thus, clients query for different specific hostnames and distinct sets of SLDs. *These results show that a client DNS model must ensure that (i) each client tends to stay similar across time and also that (ii) clients must be distinct from one another.*

A final aspect we explore is how quickly a client repeats a query. As we show in Figure 2, 50% of the clients send less than 2K queries per day on average. Figure 9 shows the distribution of the average number of unique hostnames that clients query per day. The number of names is less than the overall number of lookups, indicating the presence of repeat queries. For instance, at the median, a client queries for 400 unique hostnames and 150 SLDs each day. To assess the temporal locality of re-queries, we compute the stack distance [12] for each query—the number of unique queries since the last query for the given name. Figure 10 shows the distributions of the mean and median stack distance per client. We find the stack distance to be relatively short in most cases—with over 85% of the medians being less than 100. However, the longer means show that the re-use rate is not always short. *Our results show that variation in requerying behavior exists among clients, with some clients revisiting names frequently and others querying a larger set of names with less frequency.*

7 Related Work

Models of various protocols have been constructed for understanding, simulating and predicting traffic (e.g., [13] for a variety of traditional protocols and [2] as an example of HTTP modeling). Additionally, there is previous work on characterizing DNS traffic (e.g., [11,6]), which focuses on the aggregate traffic

of a population of clients, in contrast to our focus on individual clients. Finally, we note—as we discuss in §1—that several recent studies involving DNS make assumptions about the behavior of individual clients or need to analyze data for specific information before proceeding. For instance, the authors of [5] model DNS hierarchical cache performance using an analytical arrival process, while in [14], the authors use simulation to explore changes to the resolution path. Both studies would benefit from a greater understanding of DNS client behavior.

8 Conclusion

This work is an initial step towards richly understanding individual DNS client behavior. We characterize client behavior in ways that will ultimately inform an analytical model. We find that different types of clients interact with the DNS in distinct ways. Further, DNS queries often occur in short clusters of related names. As a step towards an analytical model, we show that the client query arrival process is well modeled by a combination of the Weibull and Pareto distributions. In addition, we find that clients have a "working set" of names that is both fairly stable over time and fairly distinct from other clients. Finally, our high-level results hold across both time and qualitatively different user populations—student residential vs. University office. This is an initial indication that the broad properties we illuminate hold the promise to be invariants.

References

1. OpenDNS. `http://www.opendns.com/`.
2. P. Barford and M. Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *ACM SIGMETRICS*, 1998.
3. T. Callahan, M. Allman, and M. Rabinovich. On Modern DNS Behavior and Properties. *ACM SIGCOMM Computer Communication Review*, July 2013.
4. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *AAAI International Conference on Knowledge Discovery and Data Mining*, 1996.
5. N. C. Fofack and S. Alouf. Modeling Modern DNS Caches. In *ACM International Conference on Performance Evaluation Methodologies and Tools*, 2013.
6. H. Gao, V. Yegneswaran, Y. Chen, et al. An Empirical Re-examination of Global DNS Behavior. In *ACM SIGCOMM*, 2013.
7. P. Gauthier, J. Cohen, and M. Dunsmuir. The Web Proxy Auto-Discovery Protocol. IETF Internet Draft. `https://tools.ietf.org/html/draft-ietf-wrec-wpad-01` (work in progress), 1999.
8. Websites Using Google Analytics. `http://trends.builtwith.com/analytics/Google-Analytics`.
9. Google Safe Browsing. `https://developers.google.com/safe-browsing`.
10. HTTP Archive. `http://httparchive.org`.
11. J. Jung, A. W. Berger, and H. Balakrishnan. Modeling TTL-Based Internet Caches. In *IEEE International Conference on Computer Communications*, 2003.
12. R. L. Mattson, J. Gecsei, D. R. Slutz, and I. L. Traiger. Evaluation Techniques for Storage Hierarchies. *IBM Systems Journal*, 1970.
13. V. Paxson. Empirically Derived Analytic Models of Wide-Area TCP Connections. *IEEE/ACM Transactions on Networking*, 1994.
14. K. Schomp, M. Allman, and M. Rabinovich. DNS Resolvers Considered Harmful. In *ACM Workshop on Hot Topics in Networks*, 2014.