2022 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT) | 978-1-6654-6090-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/BDCAT56447.2022.00044

Understanding DNS Query Composition at B-Root

Jacob Ginesin, *Northeastern University,* Jelena Mirkovic, *USC/ISI*

***Abstract*—The Domain Name System (DNS) is part of critical internet infrastructure, as DNS is invoked whenever a remote server is accessed (an URL is visited, an API request is made, etc.) by any application. DNS queries are served in hierarchical manner, with most queries served locally from cached data, and a small fraction propagating to the top of the hierarchy – DNS root name servers. Our research aims to provide a comprehensive, longitudinal characterization of DNS queries received at B-Root over ten years. We sampled and analyzed a 28-billion-query large dataset from the ten annual "Day in the Life of the Internet (DITL)" experiments, from 2013 through 2022. We sought to identify and quantify unexpected DNS queries, establish longitudinal trends, and compare our findings with published results of others. We found that unexpected query traffic increased from 39.57% in 2013 to 67.91% in 2022, with 36.55% of queries being priming queries. We also observed growth and decline of Chromium-initiated, random DNS queries. Finally, we analyzed the largest DNS query senders and established that most of their traffic consists of unexpected queries.**

***Index Terms*—Domain Name System, DNS root, security, DITL, measurement**

I. Introduction

THE Domain Name System (DNS) is the Internet's system for mapping between alphanumeric resource names (e.g. the name of a web or a mail server) and their respective values (in most cases an IP address). This system is critical for basic functioning of the Internet. DNS queries are issued whenever a remote server is accessed by a client application, usually to obtain an IP address for a given server's name. A missing or an incorrect reply to such queries can halt all communication between the server and the client.

In many cases DNS queries are answered locally by a *caching resolver* at the client's network, which oftentimes has the full response in its cache. If the local caching resolver does not know the answer to a query, it will interact with other DNS participants to obtain, cache, and return the answer to the client. The DNS system is organized as a hierarchy of DNS name servers, with servers at the higher levels of the hierarchy containing information about servers at one level lower. At the top of the hierarchy resides 13 DNS roots. Most DNS queries are satisfied by lower levels of the DNS hierarchy, but some propagate to DNS roots.

We call malformed or too frequent queries that propagate to the DNS roots *unexpected queries*. Previous work analyzing historical DNS traces data has revealed a surprising amount of unexpected queries hitting the root nodes, including queries that are malformed, random-looking, or repeated at high frequency [10], [11], [19]. Yet, none of the previous work provides a full characterization of unexpected traffic into disjoint and meaningful categories. This classification would help us better understand root causes of different types of unexpected traffic. Our research aims to develop a comprehensive classification of DNS queries, and use it to study trends in DNS query traffic at B-root over the past ten years. We make the following contributions in this paper:

1) We propose a detailed, comprehensive DNS query classification scheme to cover main root causes of unexpected DNS traffic.
2) We quantify unexpected DNS query traffic at B-root, one of 13 DNS roots, both in aggregate and per class of interest. We study longitudinal trends in unexpected DNS queries over the course of ten years, using annually collected "Day in the Life of the Internet (DITL)" data [4]. We find an increase in unexpected traffic — from 39.57% in 2013 to 67.91% in 2022. We additionally find 36.55% of traffic in 2022 is due to priming (empty) queries.
3) We identify top senders of DNS queries to B-root, then classify the traffic coming from each sender. We reveal most traffic from top senders consists of unexpected queries.

II. Background and Related Work

In this section we provide more details about DNS hierarchy and query resolution, as well as discuss prior work characterizing traffic at DNS query resolvers.

A. DNS Hierarchy

DNS queries are issued by applications and operating systems whenever a connection is established with a remote server. For example, if a user types the URL `www.example.com` into their browser, a DNS query containing the aforementioned name is sent to discover the corresponding IP address. The query is first sent to a *caching resolver* – usually a server in the same local network as the query sender. The caching resolver (*resolver* for short) will attempt to respond by searching for the query's answer in its cache. If the full answer is not in the cache, the resolver will interact with different *authoritative name servers* to try to determine the full answer. Such an answer will be returned to the client, then saved in cache to respond to potential future queries.

The DNS utilizes a distributed, hierarchical zoning system in order to designate authority, ensure the system's robustness, and effectively distribute query traffic across servers. Each name, e.g. `www.example.com`, can be viewed as collection of name segments separated by dots [2]. Each name segment resides in a separate name space, which genral has a designated *authoritative name server*. Such servers will answer queries about names within that name space, either by providing the full answer or by directing the query sender to an authoritative name server for a subset of the given name space. In our example, the caching resolver trying to find the IP address for `www.example.com` may have in its cache the name and

978-1-6654-6090-3/22/$31.00 ©2022 IEEE
DOI 10.1109/BDCAT56447.2022.00044

IP address of the name server authoritative for `.com` *top-level domain (TLD)*. The resolver may repeat its query to the TLD name server, and receive back the name and IP address of the name server authoritative for `example.com` *second-level domain (sTLD)*. The resolver will cache this new information, then repeat its query to this sTLD name server and receive a full response, which will be cached and returned to the client.

If the resolver from our example does not have information about the relevant TLD server (e.g., `.com`), it will send its query to one of DNS root name servers. The names and IP addresses of root name servers are often hard-coded in operating system releases, thus a resolver always knows how to reach a DNS root. The root zone is served by 13 logical root name servers (13 root server names, A–M) and hundreds of physical servers.[1] The root name server will provide the name and IP address of the relevant TLD server, which will be cached by the caching resolver. The rest of name resolution continues as described in the previous paragraph.

Responses from DNS replies carry a "time-to-live (TTL)" value, a number of seconds that the authoritative server suggests they should be cached. A resolver can decide to cache a DNS record for a shorter time than the recommended TTL. DNS records for names higher in the hierarchy, such as sTLD and TLD servers, usually have TTL values in hours or even days. Caching should ensure that a resolver can quickly reply to most client DNS queries, and that higher levels of DNS hierarchy do not receive too frequent queries from any resolver.

Two recent extensions to DNS protocol introduce changes to the query resolution process. Query minimization (QMIN) RFC 7816 [9] instructs DNS resolvers to protect their clients' privacy by only asking each authoritative name server for the name segment that the resolver is currently trying to resolve. In our example, the resolver would not send the full `www.example.com` query to each authoritative name server. Instead, it would send a `com` query to a root name server, a `example.com` query to the TLD name server, and the full query to the sTLD name server. RFC 8109 [13] introduces *priming queries*, i.e., queries of type nameserver (NS) for the root zone ".". Such queries can be sent to any root, and the reply should specify all root server names and IP addresses. Priming queries can help a resolver learn a new IP address for a root name server. In practice, the mapping between root server names and IP addresses has been stable enough as to not require additional root servers to be introduced [12].

Finally, while the most popular DNS query maps a DNS name into an IP address (query type A for IPv4 address, query type AAAA for IPv6 address), there are other types of DNS queries. A NS query returns the name and often the IP address of the authoritative name server for the specified query name. A pointer record (PTR) query asks for reverse mapping from an IP address into a DNS name. A start-of-authority (SOA) query requests some metadata about the name, such as the email address of the administrator, when the domain was last updated, and how long the server should wait between refreshes. Mail exchanger (MX) queries are for mail servers serving a given name. There exists several other, less frequent, query types [1].

[1]The number 13 used to be the maximum number of root servers due to the size limitation of UDP reply packets. After the introduction and rollout of anycast, the number of physical servers has increased, while there are still 13 root server names.

B. DNS Query Classification

Castro et al. [10] analyzed DITL datasets at eight root servers from years 2006 through 2008. They uncovered very high volumes of unexpected traffic at the root zone–almost 98% of queries were identified as unexpected. In this previous work, unexpected queries were specified to fall into any one of the following categories: invalid query class (a field in a DNS query with five valid values), A or AAAA queries where the query name is an address, queries with invalid TLDs, queries with non printable characters or underscores, PTR queries for a private IP address, identical queries (same class, type, name and ID), repeated queries (same class, type, and name but different ID), and queries where referral records (TLD and sTLD) have not been properly cached. Our study is in a sense a modern sequel to this previous work. We extend Castro et al. study in a few ways: (1) we dive deeper into invalid query categories, characterizing them by their root cause, (2) our analysis covers newer DNS query trends, like query minimization and priming queries, (3) our analysis spans ten years of DITL data albeit at only one root – B-root, and (4) we analyze top senders in several invalid query categories to investigate if there are any commonalities between them that would explain their querying behavior.

C. DNS Sender Analysis

A recent study measured centralization in senders to B-Root, with a specific focus on tracking the 5 top cloud providers [14]. This study reveals that in 2020, more than 30% of all queries to two TLD name servers and B-root were sent from five large cloud providers: Google, Amazon, Microsoft, Facebook, and Cloudflare. We extend upon this work by examining and ranking all senders at B-root instead of just cloud providers. We find senders which often send at rates that are too high often send predominantly queries with invalid TLDs. A similar trend was observed by Castro et al [10] in 2006–2008.

III. Dataset

Each year, most root servers and several TLD servers collect and publish all their query traffic on a specific, predetermined day. This effort is known as "Day in the Life of the Internet" or DITL, and is undertaken to produce useful data for research [4]. Although DITL data has been collected at other root name servers since 2006, B-Root joined the experiment in 2013; likewise, our sample covers just ten years of traffic (2013 through 2022) [16]. Ideally, we would have analyzed all roots' data from DITL collection. However, this data is only available on OARC servers, which have limited computational power. For this reason, we started with B-root data, which was available locally at our servers, and we plan to extend our analysis to other roots' data in the future.

To speed up our analysis, we analyzed samples from B-Root's DITL data. For each year of DITL experiment data (2013 through 2022), we utilized the sample function from Python's random package [17] to generate 10 year-specific

subsets of data. For each of our 10 subsets, we additionally generate 4 subsets each denoting one of four time zones: 12-1am, 6-7am, 12-1pm, and 6-7pm. Table I shows the details of the dataset we analyze in this paper. In total, we study 28 billion DNS queries spread over 10 years, one day per year.

Date (Y/M/D)	Main	12-1am	6-7am	12-1pm	6-7pm
2013/05/28	1.00B	19.96M	37.68M	69.44M	48.53M
2014/04/28	0.98B	255.10M	61.24M	11.19M	46.80M
2015/04/13	0.96B	35.87M	42.82M	63.72M	29.55M
2016/04/05	4.22B	148.62M	154.42M	344.51M	172.56M
2017/04/11	3.50B	134.81M	128.26M	183.75M	172.21M
2018/04/10	3.90B	97.01M	121.28M	255.69M	181.40M
2019/04/08	3.63B	93.38M	157.30M	295.90M	103.52M
2020/05/05	3.52B	67.86M	144.47M	267.39M	116.75M
2021/04/13	2.07B	79.78M	69.06M	112.40M	93.40M
2022/04/12	4.11B	92.40M	105.70M	326.66M	87.48M

TABLE I: Evaluated Datasets

IV. Methodology

In this section we describe our methodology to determine query classes, and how we implemented our approach.

A. Classification Goals

While previous works have primarily focused on opportunistically measuring some aspects of unexpected queries, our goal is to provide a more comprehensive, general classification. We seek to define a method to allow us to stratify DNS queries into sections denoting different root causes of unexpected traffic. To do this, we consider two qualities to be of interest when creating our classification method: full-coverage and mutual exclusivity. A method that has full-coverage places every single query of a given dataset into a single, defined category at each level of classification. A method that is mutually exclusive ensures there's no overlap between query categories at the same classification level.

We opt to primarily classify queries based on query names; yet, we also recognize it's possible to classify queries by other features (e.g. query types, cached/uncached queries, repeated queries) and we hope to do so in the future. In developing our name-based method that fulfills the aforementioned qualities, we consider the DNS zoning hierarchy as defined by RFC 1035 [2]. In moving down the zoning hierarchy from the root zone (right to left in the context of a textual DNS query), we recognize three mutually exclusive possibilities: the query is empty (“.”), the query ends following some text (e.g. “foo.”), or the query continues (e.g. “[more query].foo.”). This method is recursive on the latter query case, which is relevant for processing queries whose subdomains are multi-level [2].

In accordance with our method, we stratified our data into three all-encompassing, mutually exclusive categories - empty (“.” – these are likely priming queries), has-TLD (e.g. “example.com”), and one-word (e.g. “foobar”). Next we split the has-TLD category into valid-TLD and invalid-TLD by comparing the TLD of each query to IANA's maintained valid-TLD list [3]. Within one-word category we also attempt to detect presence of valid TLDs, which can occur due to query minimization [9]. We further stratified the valid-TLD category by categorizing valid TLDs by frequency.

Previous to stratifying invalid-TLD queries by TLD frequency, we separated out classifications of queries we deemed interesting. We quantified queries that contained top-level domains consisting of entirely numbers because they're deemed invalid by RFC1034 [1]. We quantified queries from Appletalk, a discontinued proprietary suite of networking protocols for Apple products, as it could potentially indicate legacy Apple product usage [8], leaking private data into the public Internet. We quantified queries with TLDs containing “bad encoding” (ASCII depicted as “\xxx\xxx”) because of its high frequency in DITL data. Because Chromium-initiated queries are known to occasionally contain an invalid-TLD [15], we quantified those as well (the importance of Chromium-initiated queries is discussed further in Section V-E).

We separated Chromium-initiated queries from within the one-word category, due to their overabundance in certain years. Chromium-initiated queries are discussed further in Section V-E. We quantified minimized queries (minimized queries at root servers look like one-word queries whose content is a valid top-level domain) in our collections after the technique's introduction in March of 2016 [9]. Minimized queries and their importance are discussed further in Section V-F.

Our implementation of our classification method involves use of dictionary-based matching and regular expressions. We achieve exclusivity by enforcing the order in which we apply classification criteria within a Python program.

V. Results

In this Section we present our results. We show the breakdown of DNS query traffic in 2013 and 2022 in Section V-A. We analyze trends in query types in Section V-B. We analyze longitudinal trends in Section V-C. We explore top senders of queries to B-Root in V-D. We specifically explore Chromium-initiated queries in Section V-E. We quantify the increasing presence of minimized queries in Section V-F. We explore empty queries in Section V-G.

A. 2013 & 2022 B-Root Traffic Breakdown and Comparison

We applied our classification method to DITL collections of B-Root DNS traces from 2013 and 2022 with the intent of revealing trends over the past ten years (see Figures 1 and 2 respectively). After splitting our data into the four previously designated categories - empty, valid-TLD, invalid-TLD, and one-word - we did additional work to further stratify each category. Within valid-TLD, we quantified the highest frequency valid top-level domains. Within invalid-TLD, we quantified Appletalk queries [8], queries with top-level domains that are incorrectly encoded (e.g. ”[query].\xxx\xxx\xxx”), queries with all-number top-level domains [2], and Chromium-initiated queries (see Section V-E). Beyond these specific categories within invalid-TLD, we quantified the highest frequency unique invalid top-level domains. Within the One-Word category, we quantified Chromium-resulting queries. Because minimized

[2]All-number top-level domains are specified as invalid by RFC1034.

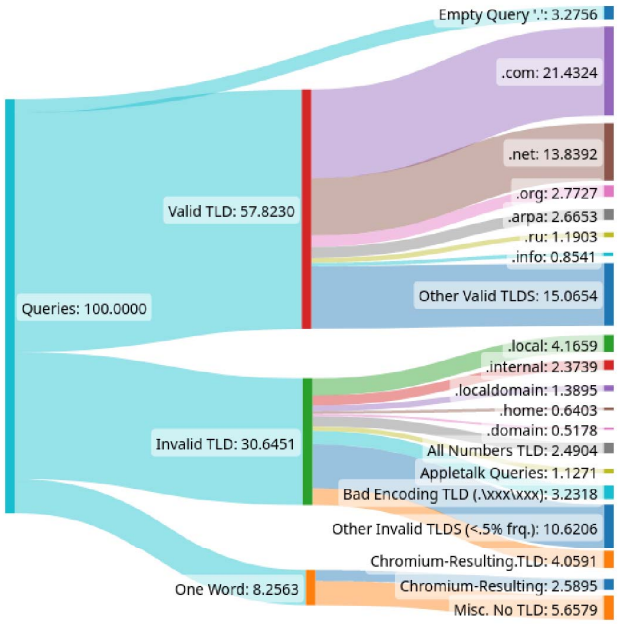

Fig. 1: Stratification of 1.00 billion DNS traces at B-Root in 2013

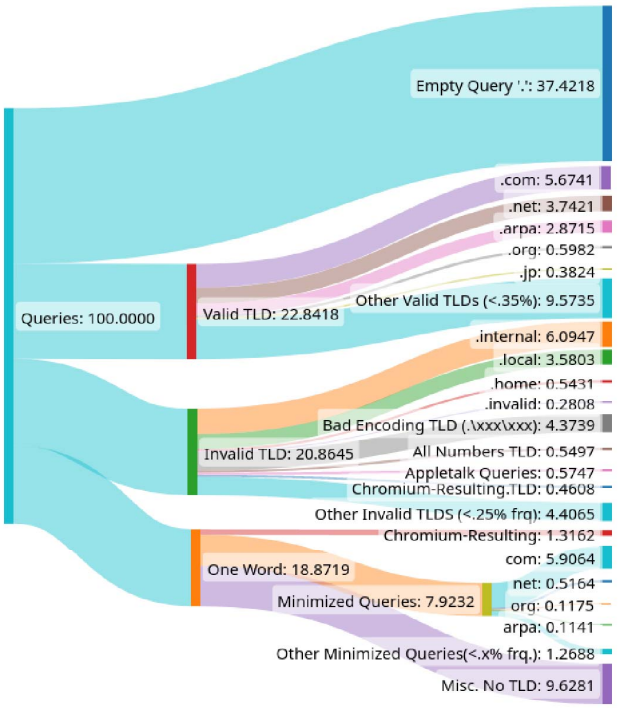

Fig. 2: Stratification of 4.11 billion DNS traces at B-Root in 2022

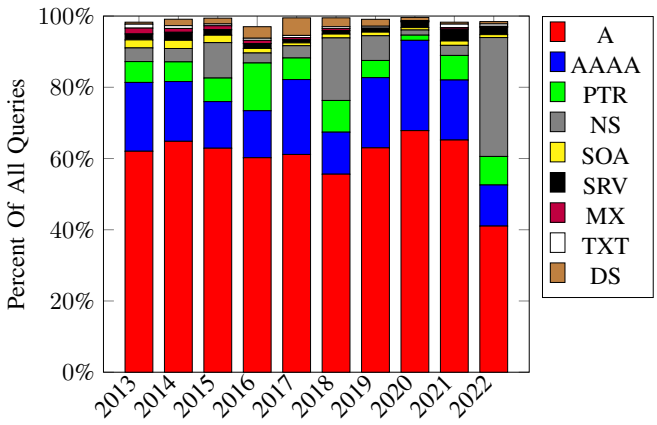

Fig. 3: Breakdown of DNS query types at B-Root from 2013 through 2022

queries were introduced in 2016, we characterized them only in our 2022 dataset.

Between 2013 and 2022, we see a 34% increase in empty queries, a 36% reduction in valid-TLD queries, a 10% reduction in invalid-TLD queries, and a 10% increase in one-word queries. The large increase in empty queries is significant and could be due to priming queries, as discussed in Section V-G.

Within valid-TLD, we see a sharp reduction in the percentage of `.com` queries (21.43% to 5.67%), `.net` queries (13.83% to 3.74%), and .org queries (2.77% to 0.60%). Surprisingly, `.arpa` queries stay at approximately the same percentage across the 10 year gap (2.66% to 2.87%).

Within invalid-TLD, we see a small increase in `.internal` queries despite an overall reduction in the category—this is potentially indicative of a persistent, growing leak. Appletalk queries decrease from 1.13% to .57%, which is expected given Appletalk is long defunct [8].

B. Query Types

Figure 3 shows the distribution of queries by type at B-Root from years 2013 through 2022. For all years, A-Type queries, used to request an IPv4 address for a given query name, are the most common (60% of the total previous to 2022). AAAA-type queries, used to request IPv6 an address, are generally the second most common query type (15% of the total previous to 2022). In 2022, we measure a large reduction in A and AAAA-type queries and a large increase in NS-type queries. The increase in NS-type queries is associated with the implementation of resolver priming [13]—priming queries are further discussed in Section V-F.

C. Longitudinal Trends

We applied our classification method to each of our collections of B-Root DNS traces from 2013 through 2022 with the intent of discovering longitudinal trends. Figure 4 shows the breakdown of empty, one-word, invalid-TLD, and valid-TLD queries for each year 2013–2022. Valid-TLD queries consistently decline from 57.82% in 2013 to 22.84% in 2022. invalid-TLD queries stay approximately constant through the 10 year sample, hovering between 20% and 30% of all queries.

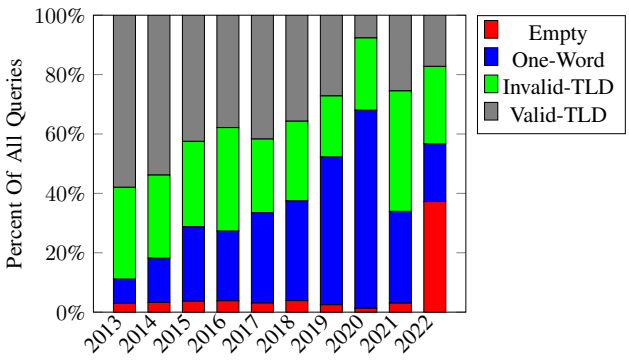

Fig. 4: Breakdown of longitudinal trends from 2013 through 2022 in DITL datasets at B-Root

One-word queries see a steady increase from 8.26% in 2013 to 68.45% in 2020, followed by a sharp decline to 18.87% in 2020. This rise and fall is largely due to Chromium-resulting queries, as further discussed in section V-E. Empty queries hovered around 3% until jumping to 37.42% in 2022. This sudden increase is thought to be a result of excessive priming queries, as discussed in section V-G.

D. Top Senders

We identified resolvers that are top senders in DITL dataset from 2022, and show them and their query composition in Figure 5. Amazon Web Services (AWS) accounts for the 1st, 2nd, 3rd, 8th, and 10th highest IP host groups and account for approximately 14% of all queries to B-Root. AWS sends almost entirely invalid-TLD and one-word queries to B-Root. Microsoft Azure, another cloud computing platform, has a similar query classification breakdown to AWS. This is potentially indicative of rented cloud machines being misconfigured or used for malicious purposes. Charter and Compudyn, both internet service providers, account for 3.43% of all traffic to B-Root. Both providers primarily send invalid-TLD queries, potentially indicating a misconfiguration. Additionally, empty and valid-TLD queries aren't present in significant quantities from these large senders.

E. Case Study: Chromium-Resulting Queries

Chromium is an open-source web browser project primarily maintained by Google. In addition to Google Chrome, several other major web browsers including Microsoft Edge, Opera, Brave, Samsung Internet, and Amazon Silk are based on the Chromium codebase. In total, approximately 75% of the web-browser market share is Chromium-based [7].

Chromium includes a feature titled Omnibox, which allows users to enter website names, URLs, or search terms. Chromium then decides if the entered term is a URL or a search term by performing a DNS query. A URL will result in a valid response, while a search term will not — Chromium can then supply search results from Google. However, a user's machine may be behind a captive portal (e.g., in a hotel), which intercepts each DNS query and responds with either the correct response (e.g., in the URL case) or with a redirect

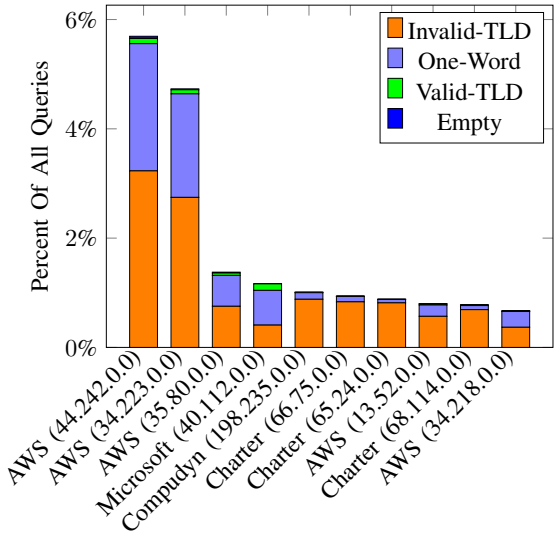

Fig. 5: Top query senders in 2022 DITL dataset at B-root

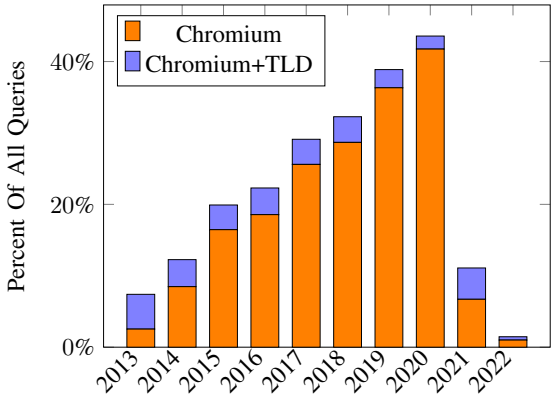

Fig. 6: Chromium-initiated queries from 2013 through 2022 in DITL datasets at B-Root

to an internal Web site (e.g., in the search term case). This situation would interfere with the Chromium's response to user input. For this reason each Chromium browser attempts to detect presence of captive portals by sending three randomly generated query names [6] [5]. These queries contain 7–15, lowercase alphabetic characters (e.g., "daozjwend.").

As a consequence of this feature combined with Chromium's high market share, root zone name servers have reported a very high quantity of Chromium-originating queries. Our findings at B-Root agree with the findings of previous work quantifying these queries [15]. We see a gradual increase in Chromium-originating queries from 2013 through 2020, followed by a sharp decline after 2020 following a change to Omnibox's probing process [18]. This trend is shown in Figure 6. Because Chromium-resulting queries have been known to appear both with and without a TLD [15], we quantify both types.

F. Case Study: DNS Query Name Minimization

We seek to specifically quantify the presence of minimized queries since the inception of query name minimization

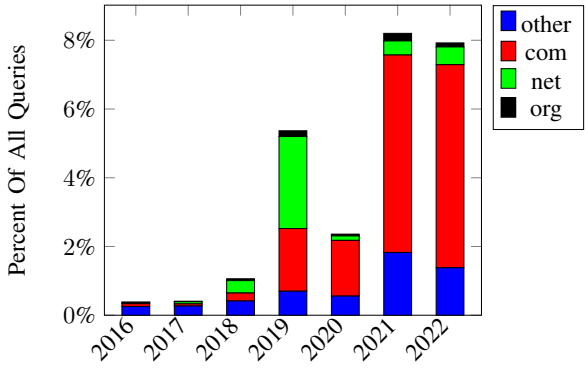

Fig. 7: Breakdown of minimized DNS queries at B-Root from 2016 through 2022

(QMIN) in March of 2016 [9]. Because the DNS is highly distributed and ultimately controlled by hundreds of different organizations, a newly implemented change in DNS protocol, practice, or implementation often takes years to be realized. Having insight into the speed at which QMIN was propagated could provide DNS researchers with a better understanding of the timeline between inception and mass adoption of new DNS improvements. Figure 7 shows a steady increase in minimized queries after 2016. In accordance with the highest frequency valid TLDs we discuss in Section V-A, the distribution of most popular minimized queries is expected.

G. Case Study: Empty Queries

One of the most notable outliers we discover in our data is the overabundance of empty queries in 2022—37.42% of queries in 2022 to B-Root are empty. To the best of our knowledge, we are the first to identify this pattern. We investigate the cause and the nature of empty queries in 2022. Figure 8 shows the top senders of empty queries to B-Root in 2022. We find empty query top senders take up a very small percentage of all empty queries as shown in Figure 8, indicating the decentralized nature of empty query senders. We find each sender, on average, sends 2.8 empty queries to B-Root. We also find 97% of empty queries sent to B-Root in 2022 are NS-type queries. A high degree of decentralization among senders and mostly NS-type queries are to be expected if the empty queries hitting B-Root are priming queries.

VI. Conclusion and Further Directions

This investigation into B-Root's DNS traces collected from the annual DITL experiment over ten years characterized longitudinal trends, as well as modern issues, such as a high volume of priming queries. Future work involves characterizing valid TLD traffic at B-root and identifying unexpected queries in that category. We would also like to analyze other root's TLD data and see if trends identified at B-root apply to other roots. We encourage other DNS operators to implement our classification method. We also hope to extend our classification approach with more categories in the future.

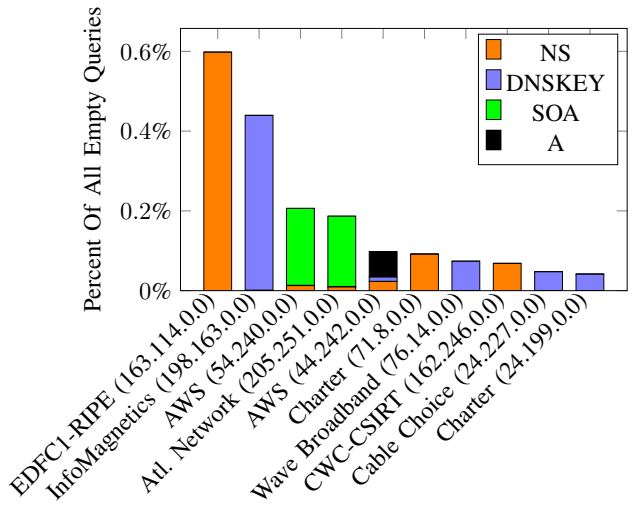

Fig. 8: Top senders of empty queries in 2022 DITL dataset at B-Root

VII. Acknowledgement

This work was performed as a part of a Research Experience for Undergraduates (REU) program, supported by National Science Foundation (NSF) grant #2051101.

References

[1] Domain names - concepts and facilities. RFC 1034, Nov. 1987.
[2] Domain names - implementation and specification. RFC 1035, Nov. 1987.
[3] List of Top-Level Domains - ICANN — icann.org. https://www.icann.org/resources/pages/tlds-2012-02-25-en, 2012.
[4] DITL Traces and Analysis — DNS-OARC — dns-oarc.net. https://www.dns-oarc.net/oarc/data/ditl, 2022.
[5] Network Stack Use in Chromium — chromium.org. https://www.chromium.org/developers/design-documents/network-stack/network-stack-use-in-chromium/#networkchangenotifier, 2022.
[6] Omnibox: History Provider — chromium.org. https://www.chromium.org/omnibox-history-provider/, 2022.
[7] W3Counter: Web Browser Market Share Trends — w3counter.com. https://www.w3counter.com/trends, 2022.
[8] Apple. Mac OS X v10.6: Mac 101 - Printing — web.archive.org. http://support.apple.co/kb/HT3771, 2010.
[9] S. Bortzmeyer. DNS Query Name Minimisation to Improve Privacy. RFC 7816, Mar. 2016.
[10] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy. A day at the root of the internet. *ACM SIGCOMM Computer Communication Review*, 38, 2008.
[11] P. B. Danzig, K. Obraczka, and A. Kumar. Analysis of wide-area name server traffic a study of the internet domain name system. 1992.
[12] IANA. Root Servers — iana.org. https://www.iana.org/domains/root/servers, 2022.
[13] P. Koch, M. Larson, and P. E. Hoffman. Initializing a DNS Resolver with Priming Queries. RFC 8109, Mar. 2017.
[14] G. C. Moura, S. Castro, W. Hardaker, M. Wullink, and C. Hesselman. Clouding up the internet: How centralized is dns traffic becoming? 2020.
[15] M. Thomas. Chromium's impact on root DNS traffic — APNIC Blog — blog.apnic.net. https://blog.apnic.net/2020/08/21/chromiums-impact-on-root-dns-traffic/, 2020.
[16] USC/ISI. ANT Datasets — ant.isi.edu. https://ant.isi.edu/datasets/all.html.
[17] G. Van Rossum. *The Python Library Reference, release 3.8.2*. Python Software Foundation, 2020.
[18] Verisign. Chromium's Reduction of Root DNS Traffic — blog.verisign.com. https://blog.verisign.com/domain-names/chromiums-reduction-of-root-dns-traffic/, 2021.
[19] D. Wessels and M. Fomenkov. Wow, that's a lot of packets. In *Passive and Active Network Measurement Workshop (PAM)*, 04 2003.