D-mystifying the D-root Address Change

Matthew Lentz mlentz@cs.umd.edu
Dave Levin dml@cs.umd.edu
Jason Castonguay castongj@umd.edu
Neil Spring nspring@cs.umd.edu
Bobby Bhattacharjee bobby@cs.umd.edu

University of Maryland

ABSTRACT

On January 3, 2013, the D-root DNS server hosted at the University of Maryland changed IP address. To avoid service disruption, the old address continues to answer queries. In this paper, we perform an initial investigation of the traffic at both the new and old addresses before, during, and since the flag day. The data we collected show non-obvious behavior: the overall query volume to the D-roots increases by roughly 50%, the old address continues to receive a high volume of queries months after the changeover, and far more queries to the old address succeed than those to the new one. Our analysis provides a window into how compliant resolvers change over and how non-standard and seemingly malicious resolvers react (or not) to the IP address change. We provide evidence that a relatively small number of implementation errors account for nearly all discrepancies that are not misconfigurations or attacks.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations; C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*Internet*

Keywords

Domain Name Service; Measurement; Root Server

1. INTRODUCTION

DNS is a foundational protocol of the Internet, and fundamental to DNS are its 13 root name servers, responsible for last-resort queries for top level domains (TLDs). To bootstrap DNS, root servers' IP addresses are widely disseminated, both out-of-band (via hints files) and in-band (via so-called priming queries, where one root server provides the others' IP addresses).

The University of Maryland hosts the D-root name server, originally referred to as terp.umd.edu (TERP). TERP has always been a root nameserver. 128.8.10.90 was an early IP address for TERP (D) appearing in BIND 4.2.1 in 1989. In order to support a robust anycasted service for "critical infrastructure," ICANN and ARIN have allocated several /24 IPv4 (and /30 IPv6) address blocks to different organizations. Root name server operators receive a new /24 IPv4 micro-allocation that is used to host anycasted root servers [8, 11]. Under this policy, the 199.7.91/24 block was allocated to the University of Maryland, and the D-root server address was changed from 128.8.10.90 to 199.7.91.13. The new /24 IPv4 block will be anycasted globally.

Though root servers very rarely change their IP addresses (historically, at most once each, to permit anycast addressing), there are some generally well-known phenomena that occur when they do. In particular, overall query volume increases, and queries persist at the old IP address, seemingly indefinitely (we review these prior observations in §2). Yet, surprisingly, there remains no explanation as to *why* these occur. This is almost our last chance to find out; D-root is the penultimate root server to change its IP address.

This paper presents and analyzes data we collected before, during, and for several months after D-root's address change. With these data, we observe the same phenomena of prior measurements at other root servers, and we identify a new one: widely differing query success rates between the old and new addresses (§3). We identify potential root causes to increases in query volume: we believe it to be largely due to a particular resolver, PowerDNS (§4). To explain persistent queries to the old address, we present a classification of name servers—those that only visit the new address after an extended period of time, those that only visit the old, and, surprisingly, those that rapidly swap between the two. We show that these different classes exhibit very different behavior, and can be used to help identify root causes (§5). We conclude that, while it can have wide-reaching effects on hosts throughout the Internet, changing a root name server's IP address can be helpful in identifying bugs, misconfigurations, and attacks, and that perhaps changing root servers' IP addresses should not be a historical event but rather a periodic, crude means of garbage collection (§6).

2. BACKGROUND AND RELATED WORK

In DNS, root name servers serve the "root zone," which lies at the top of the namespace hierarchy. Each root name server provides authoritative answers to queries regarding root zone records. For all other queries, they provide pointers to the authoritative name servers for the top level domain (TLD) (e.g., `com` or `edu`) as specified in the query.

As per its original design constraint that limited UDP responses to 512 bytes, the DNS protocol supports a maximum of 13 root name server addresses. Due to the expansive growth of the Internet, these servers quickly became under-provisioned to handle the large number of queries. In order to fix this limitation without patching the existing protocol, anycasting [8, 11] was suggested as a means

IMC'13, October 23–25, 2013, Barcelona, Spain.

http://dx.doi.org/10.1145/2504730.2504772.

Queries ($\times 10^3$)	01/02 00:00 - 01/03 09:53 Old	New	01/03 09:53 - 01/09 17:11 Old	New
Total	1,690,311	897	5,229,920	6,547,199
Invalid TLD	655,732	199	1,613,107	4,918,226
Malformed	3,934	0.6	6,802	9,104
Underscore	2,529	1.0	9,107	1,755
RFC1918 PTR	272	0.2	985	168
Non-Print Char	117	0	94	28
A-for-A	6	0	39	143
Invalid Class	4	0	19	29

Table 1: Scaled overall query volume (in thousands) broken down into reasons for invalidity for both the old and new server. The time interval is divided according to before and after the old server started advertising the new IP address (on Jan 3rd at 09:53).

to distribute the root name server instances across a large number of global replicas.

DNS Root Server Traffic Analysis.

Prior analyses of traffic at DNS root name servers identified significant levels of malformed or invalid queries [3, 4, 7, 13]. A 2008 study by Castro et al. [4] found that only 1.8% of the traffic that arrives at 8 of the 13 root name servers was valid. Based on the traffic breakdown, the following four categories comprise 94.9% of the total: identical and repeated queries, failure to cache referrals, and invalid TLDs (e.g., `local`). These results validate and expand upon a 2003 study by Wessels and Fomenkov [13], which discovered only 2.15% legitimate traffic at F-root. These prior studies focus on behavior at steady state, whereas we study a rare event; however, if DNS resolvers transition to the new address as intended, we expect the steady state behavior at the new address to mimic the old.

D-root is not the first root server to be measured during an IP address change. Barber et al. measured the J-root address change [2], and Manning presented an overview of B-root's [10].

Both initiatives reported significant, prolonged levels of traffic at the old IP address—even two years after the change, despite the fact that root name server records have limited TTLs (currently 41 days for `A` records and 6 days for `NS` records). To determine the various software versions for the resolvers that appeared to be acting incorrectly, both the B- and J-root studies used the `fpdns` [1] fingerprinting tool. Newer resolver software versions (e.g., BIND9 [9]) appeared incorrect despite having implemented the mechanisms to (ostensibly) handle these IP address changes.

Nonetheless, the root cause behind this behavior has remained undetermined. While generally well-known that an increase of volume is imminent, there is surprisingly little insight into why. In this paper, we initiate a thorough analysis of D-root's address change and identify what we believe to be the major root causes behind this and other anomalies.

3. DATA COLLECTION AND OVERVIEW

In this section, we describe the data we collected, and identify three anomalies that drive our initial measurement analysis.

3.1 Data collection

The new address is hosted on a new interface on the same physical host as the old address. We port mirrored both interfaces, and collected two datasets. The first consists of full-payload packet captures using `tcpdump` on the aggregate traffic to both the new and old IP addresses. Unfortunately, the traffic volume (~20,000 queries per second on average) is too high to allow us to capture all

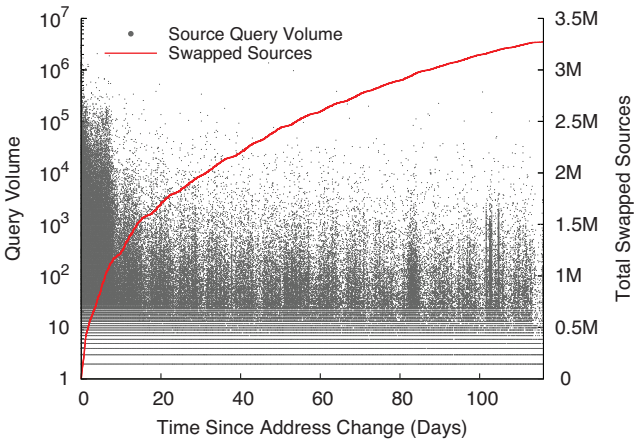


Figure 1: In the first two weeks, approximately 100,000 sources switched to the new address every day. In the following three months, this rate decreased to approximately 20,000 sources per day.

packets for an extended period. Instead, once a minute, we capture 200,000 packets, which, after the changeover, results in approximately five seconds of packet capture every minute (i.e., we collect slightly more than 8.3% of all aggregate traffic).

We sampled complete packets for the week of the trace (one day prior and approximately six days after). In order to quantify steady state behavior, we sampled complete packets at irregular intervals (200,000 packets every 5–30 minutes, rather than once a minute) a few weeks prior to the address switch. We verified that our more complete data from the day before is representative of these earlier data. Finally, we sampled complete packets a few months after the address switch (April 29th to May 2nd) to measure traffic at both addresses.

Infrequent `tcpdump` packet captures may miss precisely when any given host first switched over. To measure this, we wrote a tool `switch` that captures (1) the precise time the source first contacts the new IP address, as well as (2) approximately how many queries it had issued to the old IP address since the changeover before switching over. This second dataset sacrifices full payloads in order to get more complete information regarding changeover times and behavior.

3.2 Overview of D-root's changeover

Table 1 summarizes the traffic at both addresses the week of the address change. Recall that our trace captures only the first 200,000 packets every minute; the query volumes in Table 1 are scaled up assuming the rate at which the first 200,000 packets are received is maintained for the entire minute. The new address was announced on DNS and operator mailing lists prior to the changeover and received a negligible amount of probe/test traffic the day before.

The line in Figure 1 plots the delay in resolvers discovering the new address. The dots represent their corresponding query volume; the highest volume resolvers find the new address relatively quickly. Figure 2 shows the query volume to the old and new addresses during the week of the changeover. Query volume to both the old and new addresses show the expected time-of-day variation. Further, the thin spikes are (usually DNSSEC amplification) attacks on the root servers, which have been previously documented [6].

The adoption of the new address is rapid, and the total traffic volume to the new address exceeds that to the old within 24 hours.

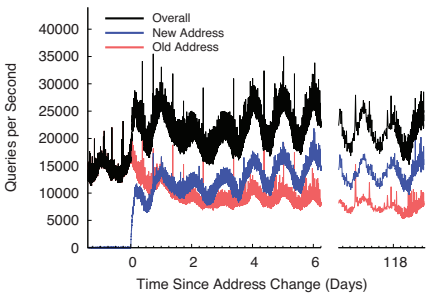


Figure 2: Number of queries per second sent to both the old and new address.

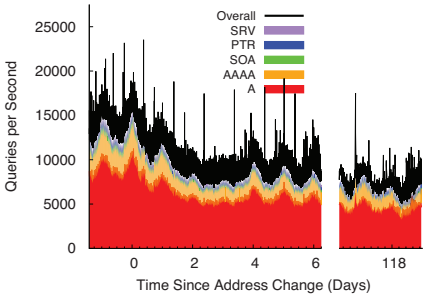


Figure 3: Query types at the *old address*, around the changeover time and several months later.

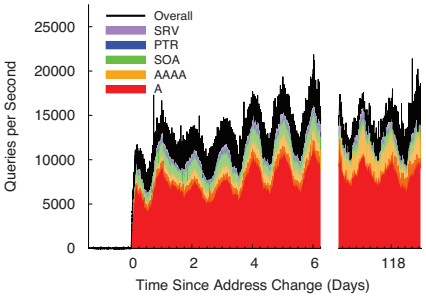


Figure 4: Query types at the *new address*, around the changeover time and several months later.

In contrast, it took nearly five days before the query volume to the new J-root address exceeded the old [2]. Note also that the new address is not initially attacked (the sharpest load spikes correspond only to queries at the old address).

Figure 2 shows that there is an immediate surge in traffic to the new address, and the rate at which it receives traffic far exceeds expected (and previously documented [2]) behavior. Further, the *total* query volume to the D-root increases dramatically, and that increase is sustained through the end of April (three months later). The old address continues to receive queries months after the changeover.

Figures 3 and 4 show the query volume to the old- and new server broken down by query type. The new address initially receives more `SOA` and `SRV` queries, but those queries are not sufficient to account for the increase in overall query volume. From Table 1, we also note that the new address receives far more invalid queries than the old address did. In contrast, the fraction of invalid queries to the old address decreases after the changeover.

These observations raise several questions that drive the analysis in the remainder of this paper:

(Q1) What causes the increase in overall traffic volume after the changeover day? Extra traffic after a root server changeover is to be expected, as resolvers will issue priming queries. But, with compliant name servers, these should constitute a small fraction of increased traffic and should dissipate once resolvers discover the root's new address and update their hints files. However, prior work has shown that there can be a prolonged increase in queries over time [2, 10]. Our data reflect this anomaly, as well (as seen in Figure 2). There are roughly 50% more queries shortly after the changeover than there were the day before, and this discrepancy continues for at least three months. While prior studies have observed this phenomenon, we are unaware of any investigation into the root cause, which is central to Q2.

(Q2) Why do servers continue to query the old address? It is not surprising that name servers continue to query the old address even months after the changeover date. Some very old BIND hints files contained exorbitantly long TTLs for root servers (99,999,999, or slightly more than three years), and given that root address changes are so uncommon, it is reasonable to assume that otherwise stable name server implementations may have faulty changeover logic. However, it is surprising that the overall volume to the old address has stabilized shortly—to approximately 50% of its original query volume—after the changeover.

(Q3) Why are queries to the old IP address on average more successful than those to the new address? Table 1 shows that the queries to the old address result in fewer `NXDOMAIN` responses than the queries to the new. To the best of our knowledge, we are the first to observe this phenomenon. Seemingly straightforward explanations for why a name server would remain at the old address—a misconfiguration or a faulty implementation—do not appear to explain these increased success rates.

4. WHY DOES QUERY VOLUME INCREASE?

Intuitively, there are two (not necessarily mutually exclusive) causes to an overall increase in volume: new resolvers could begin to query the D-root who did not before, or queries from some (possibly strict) subset of resolvers could increase. The number of servers contacting D-root did not change significantly in the 24 hours before and after the IP address change; in fact, they dropped slightly (1,336,167 sources before, 1,187,801 after).

4.1 Excitables issue many more queries

Overall query volume increases because a (relatively) small number of sources issue many more queries ($100\times$ or more) to the new address than they did to the old. We refer to these sources as *excitables*.

For each host querying the D-root, Figure 5 plots the ratio of queries per second one day before and one day after the address change. Along with the expected symmetric clustering around $y = x$ (the $1\times$ line), Figure 5 clearly identifies the excitables: high volume hosts that increase their ratio by two or more orders of magnitude. By themselves, the top-1000 or so excitables account for $\sim$58% of the increase in overall query volume. The top-1000 hosts that increase their query rate by at least $2\times$ account for the entire increase in volume. Note that if high volume servers simply added the new address to their set of 13 root addresses (and if they queried each of the root servers uniformly) then their query volume would increase by $\frac{2/14}{1/13}$, less than 86%. Therefore, a different process is responsible for the increase in query volume.

4.2 Likely cause: Non-uniform server selection algorithms

Yu et al. [14] found that various versions of DNS resolvers (notably BIND and PowerDNS) adjust the rate at which they query different root servers based on RTTs, sometimes in nonintuitive ways. BIND 9.7, for instance, preferentially queries root servers with *greater* RTTs. In general, different versions of BIND do not distribute their query loads uniformly over all addresses that can

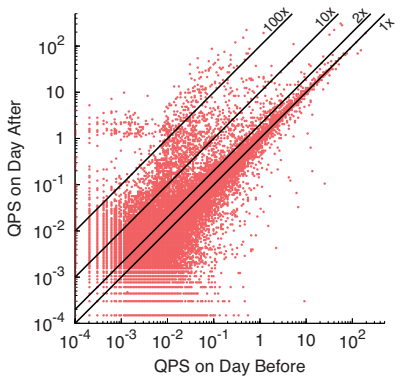


Figure 5: Comparison of queries per second (QPS) for each source in the 24 hours before and after the address change.

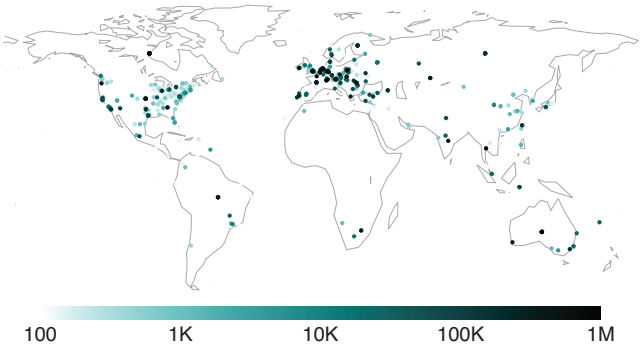


Figure 6: Locations and query volume of all excitables (resolvers whose query volume increased by at least 100× after the changeover). Most of the excitables' queries come from Europe.

serve a zone. Further, in our experiments, we see that relatively new versions of BIND (9.2, 9.5) do not implement priming correctly, and query both addresses, accounting for some of the increase in query volume, particularly in the 2×–10× range.

BIND implementations do not explain the sudden increase by 10×, 100× or more. PowerDNS, which is popular in Europe, on the other hand, exhibits a "spike distribution," causing it to almost exclusively contact the single lowest-RTT server [14]. Figure 6 shows the geographic distribution of servers whose query volumes increased by at least 100×; 62% of these servers are located in Europe, 20% in North America, 10% in Asia, 7% in Oceania, and the rest in South America and Africa. D-root, however, is not the lowest-RTT root server for European hosts; our measurements from both academic and residential hosts in Germany indicated that D-root (both old and new addresses) had an RTT one order of magnitude larger than the hosts' closest root server.

However, we have observed from an analysis of PowerDNS version 3.1 (the latest version on the D-root changeover date) that it will provide a greater weight to a *new* server, even though it does not have the lowest RTT. This can arise as certain corner cases may cause PowerDNS to stop updating the RTT-based server selection mechanism, causing PowerDNS to "stick" to a single server. It appears that the only way to exit this state is to restart the software. This bug, coupled with the fact that PowerDNS sends *all* its queries to the chosen server, could explain the increase in query volume. Further, PowerDNS has increased in popularity relatively recently, which would explain why prior address changes did not witness such extreme change in query volume. Finally, Figure 6 shows that most of the highest volume excitables are hosted in Europe, where PowerDNS is used extensively; PowerDNS reported in 2012 that they power around 30% of all domains in Europe [12]. **These observations point to resolvers using PowerDNS as being the primary source of increased traffic volume to D-root.**

Unfortunately, PowerDNS is not responsive to the latest version of the `fpdns` fingerprinting tool, and we have not yet developed a robust method for fingerprinting PowerDNS resolvers. (Reassuringly, the `fpdns` tool fails to identify the high volume hosts as BIND.) We are in the process of trying to both identify PowerDNS resolvers in the wild, and to reproduce the bug in older versions of PowerDNS which would enable definitive attribution of the increase in volume to PowerDNS.

5. WHO'S STILL USING THE OLD ADDRESS?

Technically, a DNS root IP address change does not require maintaining connectivity on the old IP address if all resolvers handle an individual failure by contacting any of the other 12 root servers. However, older resolvers that do not perform priming queries (and use outdated hints files) may completely break. A fundamental question facing a DNS root address change is: At what point can one responsibly shut down the old IP address? When all hosts switch over? When some fraction of traffic switches over? Or perhaps when it appears that the only hosts still contacting the old address are faulty?

The original expectations for D-root's changeover were that (1) the vast majority of traffic would migrate over to the new address relatively quickly, (2) the only remaining traffic would be due to faulty servers or botnets, and thus (3) a few months of running the old address would suffice.[1] However, the changeover has not been as rapid in practice: 63% of sources do not switch over after four months (by May 2, 2013), and they generate 4,721 queries per second on average.

In this section, we investigate why, after several months, the old address continues to see such a high volume of queries. We do not believe there to be a single reason for this. Instead, we seek to classify name servers among several groups, and to identify the likely causes among these groups.

5.1 Classifying resolvers

We begin by investigating the types of access behaviors resolvers exhibit towards the two addresses. So as not to confuse hosts who *never* change to the new address from those who have not *yet* changed, we focus on data collected in April and May 2013, three to four months after turning on the new IP address (and well beyond the 41 day TTL). Our hypothesis was that, after this much time, there would be very few hosts on the old address, most on the new address, and few to none who regularly contact both, as that would be incorrect behavior.

The results in Figure 7 tell another story. In this figure, we plot for each source viewed in our April/May data the difference between the fraction of queries it issued at the old address and the fraction it issued at the new. A resolver who contacted only the old address has a value of 1; a resolver who contacted only the new has value −1; a value of 0 corresponds to an equal amount of queries

[1]L-root had similar expectations, and in fact relinquished the old IP address after six months, which was "hijacked" soon after [5].

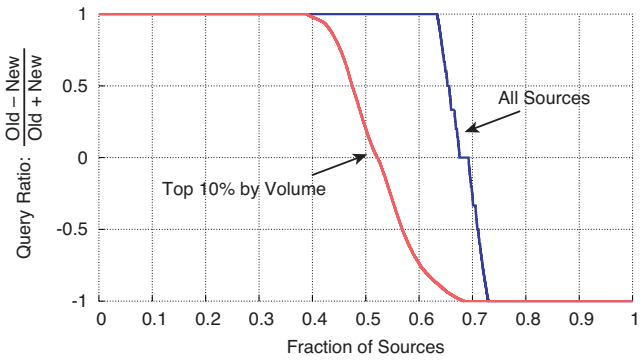


Figure 7: Fraction of sources with various query patterns to the old and new IP addresses. A query ratio of 1 or -1 corresponds to sources who only send to the old or new IP address, respectively.

to old and new, and so on. From this we see that of the top 10% of the hosts by volume (who constitute roughly 80% of the overall volume), a mere 32% contact the new address only. Approximately 40% of the resolvers we saw in April/May contact the old only. We refer to these hosts who latch onto the old address as *barnacles*.

This leaves a remainder of about 28% who swap between both old and new at varying rates—we refer to these as *swappers*. We investigated the rate at which these hosts swap (data not plotted). The majority do so infrequently (roughly once every 20 queries), though some (about 2%) swap at least as fast as every two queries they issue.

The rest of this section analyzes those who continue to query the old address after several months: the barnacles and the swappers. As a point of comparison, we also include a baseline of hosts which we refer to as *normals*: sources we expect to act in a reasonably correct manner. Specifically, a set of large ISP nameservers constitutes our "normals": Verizon DSL, Verizon HS, QWEST, COX, and Speakeasy/Megapath (business only).

5.2 Classifying root causes

For each host from each of the three classes of resolvers (barnacles, swappers, and normals), we measure it along two features. The first is the fraction of failed queries the host issues (including those that issue `NXDOMAIN` responses as well as malformed queries). The rationale behind this feature comes from a manual investigation of the data; we identified that some hosts never failed (because they issue the same small set of queries constantly), and some always failed (because they issued seemingly random strings, or incorrectly configured their internal root DNS servers and issued many queries for a TLD of `local` or `internal`). With a correctly implemented cache, one would expect high failure rates at the root (as the root knows of relatively few, long-TTL TLDs).

The second feature we measure a host against is its *query diversity*: the number of unique queries (in terms of domain name) it issues divided by the total number of queries it issues. Query diversity takes a value in (0,1], and with correctly implemented negative and positive caches, one would expect diversity to be high.

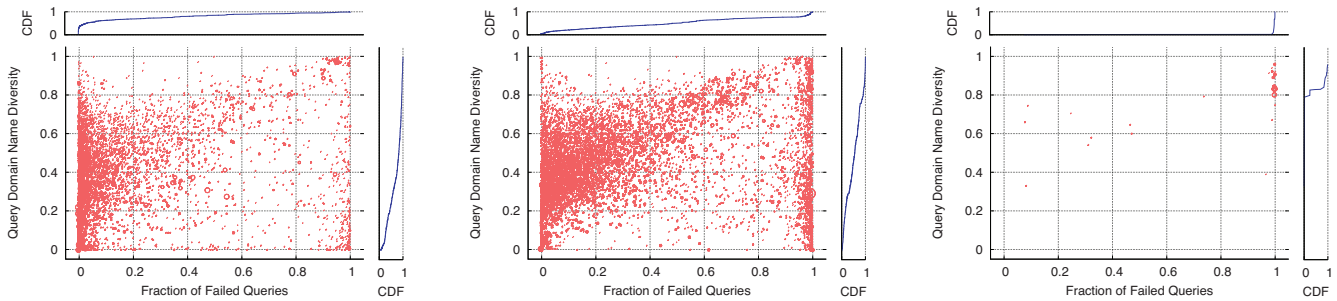
We present plots for all three types in Figure 8. We use the April–May dataset for classifying hosts as barnacles and swappers, and the more detailed January dataset for calculating their failure rates, query diversities, and query volumes.

Normals. As expected, the normals exhibit high failure rates and high diversity, forming a large cluster in the upper right corner of Figure 8(c). It is important to note, however, that a high failure rate and high query diversity does not necessarily imply correct behavior. For instance, 47% of the queries from a subset of the Google public DNS resolvers contain random strings (thus have high diversity) and include a TLD of `internal` (and thus have a high failure rate). This is likely caused by a misconfiguration of some of Google's name servers meant to serve a private namespace.

Barnacles. The vast majority of barnacles (Fig. 8(a)) exhibit low failure rates, suggesting that these are resolvers that are misconfigured, used for measurement, or used for attack. *Disproportionately low failure rates from hosts who never query the new address are a viable explanation for the anomaly wherein the old address experiences a lower failure rate than the new (Q3).*

Among the barnacles, we see resolvers that continually query for: (1) lists of known name servers (likely measurements), (2) DNSBLs (spambots or attacks), or (3) very small sets of names (embedded devices that implement their own resolver looking for updates and patches). We believe *all* of the barnacles with low error rates (<10%) are pieces of software issuing mechanized queries using incorrect DNS implementations. Moreover, we believe that barnacles with very low query diversity (wherein the resolver asks for the same name repeatedly) can be specifically attributed to misconfigurations or hard-coded attack software that does not have the facility for handling address changes.

There are a few very high volume barnacles that have approximately 30% query diversity and 50% failures. These are OpenDNS resolvers. Because OpenDNS resolvers are publicly available, we



(a) Barnacles (b) Swappers (c) Normals

Figure 8: Diversity of domain names in queries versus the fraction of failed queries for the three types of resolvers. Each circle represents a source /24, where the area is a function of the query volume.

conjecture that their behavior is a composite of normal queries, attacks, and misconfigurations.

Swappers. Finally, swappers (Fig. 8(b)) exhibit a nearly uniform distribution of failures (note the linear CDF at the top of the plot). We believe that the very low error rate queries in this set also represent mechanized bots, as do the very low diversity queries. Moreover, there is a positive correlation between failure rates and diversity. We believe the bulk of these (relatively low volume) resolvers simply use both addresses interchangeably. We are in the process of fingerprinting each of these resolvers and with the hope of mapping the ones that respond to known implementations that prime incorrectly and use all known addresses.

6. DISCUSSION AND OPEN QUESTIONS

Many signs of excitables point to PowerDNS: it is most popular in Europe (where the majority of excitables come from), we have identified that some versions include both of D-root's IP addresses in its list of root name servers, and its root selection algorithm can send all queries to a single name server. However, our runs of `fpdns` have turned up only a single PowerDNS resolver running an old version of the software. `fpdns` is known to be unable to identify PowerDNS; further, the vast majority of the European resolvers are configured to not answer external queries. Validating our hypothesis—either by improving `fpdns` or by simply running a PowerDNS resolver from multiple vantage points (with different RTTs to the new and old addresses)—is an area of ongoing work.

Our analysis of why name servers continue to visit the old address identifies many examples of what appear to be misconfigurations, buggy code, or scanners. For example, many queries from these barnacles are redundant yet they infrequently fail; among these, we have identified resolvers that scan DNSBLs. One question is: do these bugs and behaviors occur at smaller scales for the less-concentrated TLD servers or even others, perhaps creating similarly inexplicable fluctuations in traffic volume? Answering this question may provide greater confidence that barnacle's redundant queries account for the large difference in query success rates between the old and new IP addresses.

Answering these questions may require old-fashioned footwork: getting in touch with operators. One promising, potential outcome of this is that a DNS root server IP address change may be able to assist in identifying and raising awareness about bugs, common misconfiguration errors, and possibly attacks.

Indeed, perhaps changing root DNS IP addresses should be done every so often as a matter of regular practice! So doing would possibly encourage operators to run serviceably recent versions of BIND or PowerDNS and discourage hard-coding. In the long run—and if our hypothesis is correct that many of the barnacles are buggy or forgotten code—then the occasional address change could serve as a crude form of garbage collection.

Acknowledgments

We thank Xiehua Li from Hunan University, the anonymous reviewers, and our shepherd, Mark Allman, for their helpful comments on the paper. We also thank James Litton for his help in analyzing the PowerDNS code. This work was supported in part by NSF Awards CNS-0917098, IIS-0964541, and CNS-1255314.

7. REFERENCES

[1] R. Arends and J. Schlyter. `fpdns`. `https://github.com/kirei/fpdns`.

[2] P. Barber, M. Larson, M. Kosters, and P. Toscano. Life and Times of J-Root. In *NANOG32*, Oct 2004.

[3] N. Brownlee, K. Claffy, and E. Nemeth. DNS Measurements at a Root Server. In *IEEE Global Communications Conference (GLOBECOM)*, 2001.

[4] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy. A Day at the Root of the Internet. *ACM SIGCOMM Computer Communication Review (CCR)*, 38(5):41–46, 2008.

[5] D. Conrad. Ghosts of Root Servers Past. `http://blog.icann.org/2008/05/ghosts-of-root-servers-past/`.

[6] A. Cowperthwaite and A. Somayaji. The Futility of DNSSec. In *Annual Symposium Information Assurance (ASIA)*, 2010.

[7] P. Danzig, K. Obraczka, and A. Kumar. An Analysis of Wide-Area Name Server Traffic: A Study of the Internet Domain Name System. In *SIGCOMM Conference on Data Communication*, 1992.

[8] T. Hardie. Distributing Authoritative Nameservers via Shared Unicast Addresses. *RFC 3258*, Apr 2002.

[9] Internet Systems Consortium. BIND. `https://www.isc.org/software/bind`.

[10] B. Manning. Persistent Queries and Phantom Nameservers. In *CAIDA-WIDE Workshop*, 2006.

[11] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. *RFC 1546*, Nov 1993.

[12] PowerDNS Technologies. `https://www.powerdns.com/resources/PowerDNSTechnologies.pdf`.

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

[14] Y. Yu, D. Wessels, M. Larson, and L. Zhang. Authority Server Selection in DNS Caching Resolvers. *ACM SIGCOMM Computer Communication Review (CCR)*, 42(2):80–86, 2012.