Reexamining DNS From a Global Recursive Resolver Perspective

Hongyu Gao, Vinod Yegneswaran, Jian Jiang, Yan Chen, *Member, IEEE*, Phillip Porras, Shalini Ghosh, and Haixin Duan

***Abstract*—The performance and operational characteristics of the Domain Name System (DNS) protocol are of deep interest to the research and network operations community. In this paper, we present measurement results from a unique dataset containing more than 26 billion DNS query–response pairs collected from more than 600 globally distributed recursive DNS resolvers. We use this dataset to reaffirm findings in published work and notice some significant differences that could be attributed both to the evolving nature of DNS traffic and to our differing perspective. For example, we find that although characteristics of DNS traffic vary greatly across networks, the resolvers within an organization tend to exhibit similar behavior. We further find that more than 50% of DNS queries issued to root servers do not return successful answers, and that the primary cause of lookup failures at root servers is malformed queries with invalid top-level domains (TLDs). Furthermore, we propose a novel approach that detects malicious domain groups using temporal correlation in DNS queries. Our approach requires no comprehensive labeled training set, which can be difficult to build in practice. Instead, it uses a known malicious domain as anchor and identifies the set of previously unknown malicious domains that are related to the anchor domain. Experimental results illustrate the viability of this approach, i.e., we attain a true positive rate of more than 96%, and each malicious anchor domain results in a malware domain group with more than 53 previously unknown malicious domains on average.**

Manuscript received February 07, 2014; accepted August 15, 2014; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Sen. This material is based upon work supported in part by the National Science Foundation under Grants No. CNS-0831300 and CNS-1314956, the Army Research Office under Cyber-TA Grant No. W911NF-06-1-0316, the National Basic Research Program (973 Program) of China under Grant No. 2009CB320505, the National Natural Science Foundation of China under Grant No. 61472215, and the Open Foundation of the State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) SKLNST-2013-1-17. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies. *(Corresponding author: Yan Chen.)*

H. Gao is with Google, Mountain View, CA 94043 USA (e-mail: hygao@u.northwestern.edu).

V. Yegneswaran, P. Porras, and S. Ghosh are with the Computer Science Laboratory, SRI International, Menlo Park, CA 94025 USA (e-mail: vinod@csl.sri.com; porras@csl.sri.com; shalini@csl.sri.com).

J. Jiang is with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China, and also with the School of Information, University of California, Berkeley, Berkeley, CA 94720 USA (e-mail: jiangjian@berkeley.edu).

Y. Chen is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208 USA, and also with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: ychen@northwestern.edu).

H. Duan is with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China (e-mail: duanhx@tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2014.2358637

***Index Terms*—Domain Name System (DNS), measurement, malicious domain detection.**

I. INTRODUCTION

THE DOMAIN Name System (DNS) protocol plays a cardinal role in the operation of the Internet by enabling the bidirectional association of domain names with IP addresses. It is implemented as a hierarchical system with a few trusted root servers that distribute the responsibility of updating the name-to-IP-address mapping to hundreds of millions of authoritative nameservers that correspond to each domain. DNS as a protocol has steadily evolved since its initial specification [33]–[36], as has the mix of applications that find new and innovative ways of using it. Most applications today and future Internet architectures (such as Named Data Networks and Software-Defined Networks) depend on DNS. It is also increasingly abused by malware authors, both as an effective redirection mechanism for obfuscating location of their servers [22] and as a covert channel for command and control [19], [37].

Given its crucial importance for the Internet's functioning, DNS has been the subject of many measurement studies during the last decade. Prior measurement studies have scrutinized the behavior of DNS caches [25], characterized global DNS activity from the perspective of root servers [16], [17], and evaluated the effectiveness of DNS in the context of content-delivery networks [41]. The first study of global DNS activity was by Danzig *et al.*, which uncovered the prevalence of many bugs in popular DNS implementations [18]. More recently, this problem was revisited by Brownlee *et al.*, who measured the prevalence of bogus DNS traffic at the F-root nameserver, finding that some of the same problems persist: 60%–85% of observed queries were repeated queries from the same host, and more than 14% of requests involved queries that violated the DNS specification. Jung *et al.* measured that a significant portion of DNS lookups (more than 23%) receive no answer and that they account for more than half of all DNS packets in the wide area due to persistent retransmissions.

Several of these studies were conducted more than a decade ago and often from a small number of vantage points. Collaboration between the Internet research and operations community has evolved significantly since these foundational studies, and we now have access to a new and unique data source, the Internet Systems Consortium (ISC)'s Secure Information Exchange (SIE) [20], which enables researchers to monitor DNS activity from hundreds of operational networks in real time. One of the driving forces behind such data sharing has been its untapped potential for rapidly identifying malware domains. In particular, domain registrations and DNS access patterns could be an effective means for tracking cyber-criminal behavior, and several recent studies have explored the application of machine-learning techniques to automatically identify malicious domains [8], [15], [45].

In this paper, we report on findings from a global and multidimensional analysis of DNS activity, as observed from a large set of widely distributed and operational DNS resolvers. Specifically, we analyze 2 weeks of data from more than 600 resolvers comprising more than 26 billion queries and responses. First, we systematically dissect this data, present high-level characteristics of observed traffic behavior, and identify invariant characteristics across resolvers. Second, we use this dataset to critically reexamine the validity of certain prior measurement studies, in the context of this more global perspective and modern traffic characteristics. Finally, we evaluate the feasibility of using this dataset to automatically extract malicious domain groups. We make the following key findings.

- We find that resolvers from different /24 subnets have different profiles, including query/response counts, unanswered query rates, unsolicited response rates, query type distributions, and query success-to-failure ratios.
- In comparison to prior measurement results, A queries continue to dominate, AAAA queries have sharply increased, and other query types depict a decrease in popularity.
- We find that although root servers are always available (i.e., have no unanswered queries), more than 15.1% of the queries sent by recursive DNS resolvers are unanswered.
- We explored the cause of DNS query with negative answer (queries that do not return "NOERROR"). We identify DNSBL as having a much higher failure ratio than do other query types.
- We find that invalid top-level domain (TLD) is the primary cause of query with negative answer at root servers, and that the percentage of invalid TLD has increased in comparison to the results from prior measurements. However, A-*for*-A queries have decreased in popularity and almost disappeared in our data.
- We find that 12.0% of traffic to root servers and 8.0% to other servers are *truly* repeated queries. We further identify the possible causes including concurrent query, CNAME chain sanitization, premature retransmission, and implementation quirks.
- We find that the number of DNSSEC-enabled domains has increased sharply compared to prior reports.
- We find that 24.1% of second-level domains (SLDs) have IPv6 authoritative servers.
- We find that temporal correlation of domain queries is an effective means to detect correlated domain groups. Based on this finding, we develop a novel approach that detects previously unknown malicious domains related to known anchor malicious domains. The approach achieves 96.4% detection precision and detects 53 more malicious domains on average for each given anchor domain.

II. Background and Dataset

DNS Protocol: The DNS is a distributed, hierarchical naming system that translates between domain names and IP addresses. Client end hosts (also called stub resolvers) simply contact a recursive resolver that implements the hierarchical resolution process of iterating through nameservers to perform the translation. In the example shown in Fig. 1, the stub resolver queries the local recursive resolver for the IP address of `www.example.com`. The recursive resolver usually resides within the local network of the client's organization and is managed by the organization's administrator. However, clients can also choose to contact recursive resolvers located outside their local network (e.g., OpenDNS resolvers and Google public DNS resolvers). Assuming an empty cache, the recursive resolver starts by querying the root server for the IP address of `www.example.com`. The root server responds with a referral to the `.com` TLD server. The recursive resolver then queries the `.com` TLD server, and in response is provided with a referral to the authoritative server for `example.com`, which hosts the name-to-address mapping. Finally, the recursive resolver contacts the authoritative server of `example.com` to obtain the corresponding IP address.

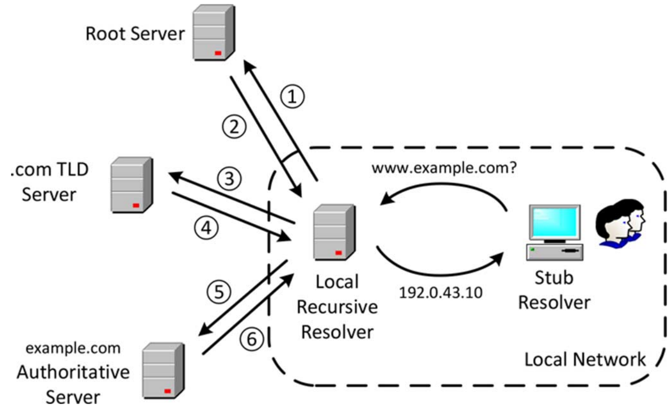


Fig. 1. Illustration of the DNS resolution process for www.example.com.

Fig. 2. Geolocation of the DNS resolvers that contribute to the data.

Data: Our data are collected from a high-volume passive DNS source at the SIE [20]. This provides a near-real-time data feed from multiple hundreds of DNS recursive resolvers distributed over the Internet. These resolvers represent large ISPs, universities, as well as public DNS service providers located in North America and Europe, suggesting a wide diversity in the user population behind these resolvers. We plot the geolocations of the DNS resolvers in Fig. 2 using the MaxMind geolocation database [32].

Due to privacy concerns, the data-collection sensor is deployed "above" the recursive resolvers and records all DNS queries and responses between the recursive resolvers and the remote DNS servers. The sensor does not collect traffic between client stub resolvers and recursive resolvers. As a result, the identities of client end-hosts that sit behind the recursive resolvers are not available.

Previous SIE data analysis has shown that 93% of the domain labels immediately under the `.edu` TLD have a resource record in the SIE data in a 2-week observation period [46]. The DNS servers that generate responses are dispersed in 70.7% of the /8 CIDR blocks and 69.2% routable autonomous systems (ASs) [46]. We collected all DNS traffic in the raw SIE channel for 2 weeks from December 9 to 22, 2012. In total, our dataset contains about 26 billion DNS queries and responses.

Note that our dataset, although the most diverse to date, still has a geographic bias because the monitored resolvers are exclusively located in the US and Europe (Fig. 2). Hence, we focus our study on macroscopic characteristics and temporal behaviors, which we believe do not have strong correlations with geographic location.

Local and Root Perspective: Since our data are collected from local recursive DNS resolvers, it naturally enables studying DNS behavior from the perspective of the local resolvers. On the other hand, 13 root servers of vital importance sit atop the DNS hierarchy. Due to their importance, multiple prior works have analyzed DNS protocol behavior from the perspective of the root servers [16], [17], [48].

We attempt to analyze our DNS data from the root perspective as well. As described in Section II, if a client-side nameserver restarts with empty cache, or the time-to-live (TTL) expires for a TLD nameserver entry, the recursive resolution process starts by querying the root servers and obtaining a referral to an authoritative TLD nameserver. Although our data are collected from local recursive DNS resolvers, the availability of the response nameserver's IP address enables us to isolate the DNS traffic to and from root servers. Given the volume and diversity of our dataset, we believe that the subset of DNS queries and responses is a representative sample of DNS traffic that root servers experience. In this paper, we analyze the DNS traffic characteristics from both the local perspective (i.e., using the full dataset) and the root perspective (i.e., using only traffic to and from root servers), whenever applicable.

III. DNS Traffic Characteristics

In this section, we analyze the characteristics of the collected DNS traffic from various perspectives.

A. High-Level Characteristics

Our data include traffic from 628 distinct DNS resolvers including 10 IPv6 resolvers. Not surprisingly, we find significant variance in the volume of DNS queries that they generate. The most active resolver generates more than 70 M queries per day, which translates to an average of more than 800 queries per second. In contrast, 407 resolvers generate fewer than 10 000 queries during the 2-week measurement period.

This observed range shows that the query volume of DNS resolvers has a heavily skewed distribution. A small fraction of deployed DNS resolvers are serving the majority of the DNS queries. This observation is consistent with that of prior measurement studies by Pang *et al.* [41] in 2004 and Osterweil *et al.* [38] in 2012. Interestingly, the vast majority of inactive resolvers belong to a European educational institution (354 resolvers) and a US educational institution (49 resolvers). We subsequently learned that DNS experiments are conducted at these institutions, and speculate that ongoing DNS experiments may be the reason behind the large number of inactive DNS resolvers. Nonetheless, the amount of traffic generated by the inactive resolvers is negligible and should not remarkably affect our measurement results.

TABLE I
Percentage of Traffic Generated From the Top 20 /24 Subnets With IPv4 Resolvers, and the Aggregate Traffic Generated by IPv6 Resolvers

Organization	Resolver #	Traffic %
US ISP A (subnet 1)	40	32.6%
US ISP A (subnet 2)	34	22.7%
US ISP A (subnet 3)	10	17.4%
Public DNS Service	4	11.7%
US ISP B	2	2.0%
US ISP C (subnet 1)	2	1.6%
US ISP C (subnet 2)	2	1.5%
US ISP D (subnet 1)	8	1.1%
US ISP D (subnet 2)	8	1.0%
US ISP E (subnet 1)	2	1.0%
EU ISP A	8	1.0%
US ISP C (subnet 3)	2	1.0%
US ISP F	4	0.8%
US ISP D (subnet 3)	8	0.7%
EU EDU (subnet 1)	11	0.6%
US ISP C (subnet 4)	1	0.5%
US EDU	50	0.4%
US ISP E (subnet 2)	1	0.4%
EU EDU (subnet 2)	2	0.2%
EU ISP B	2	0.2%
IPv6	10	0.6%

We further agglomerate IP addresses into /24, /16, and /8 subnets, respectively. We also put all resolvers with IPv6 addresses into one group. Our monitored DNS resolvers span 71 distinct /24 subnets, 33 distinct /16 subnets, and 22 distinct /8 subnets. This further validates that our data are collected from vantage points distributed widely across the IPv4 address space.

1) Organizations: We use /24 subnets to group DNS resolvers into organizations and bin all resolvers with IPv6 addresses into a special group. Although large organizations may have /16 or /8 subnets, we find /24 subnets to be a good way to group DNS resolvers as it provides sufficient abstraction and enables capturing the difference between different subnets within large organizations.

We identify the 20 top /24 subnets in our data with the highest traffic volume. By using `whois` lookups to determine the organization of the /24 subnets, we identified six commercial US ISPs, one US educational institute, two commercial European ISPs, one European educational institute, and a public DNS service provider. Many organizations deploy DNS resolvers in multiple /24 subnets as shown in Table I. Due to privacy concerns, we use the location (US or EU) and type (commercial, EDU, or public) to denote the organizations. The bulk of the data is collected from US ISP A, which serves a large population and contributes a large number of resolvers.

2) DNS Data Type: In normal operation, each DNS query is associated with a response. However, cases exist when a DNS query is not answered or a DNS response is received without a matching query, either due to misconfiguration, backscatter

TABLE II
SERVER-SIDE DISTRIBUTION OF DNS TRAFFIC, AGGREGATED BY ORGANIZATIONS

Organization	Traffic %
Akamai	11.6%
.com/.net TLD	6.1%
Amazon	4.7%
Google	4.4%
cox.net	3.2%
Apple	2.3%
Mcafee	2.1%
dynect.net	2.0%
Root	1.8%
iana.org	1.5%
others	60.3%

from attack traffic, or packet loss. Hence, we group DNS traffic in our data into three categories: query–response pairs, unanswered queries, and unsolicited responses. More than 83.3% of the entries in our data are query–response pairs, 14.9% are unanswered queries, and 1.8% are unsolicited responses. The percentage of abnormal cases, including both unanswered queries and unsolicited responses, is 16.7%, which seems anomalous and is worthy of deeper investigation. An obvious consideration is packet loss in the data collection infrastructure. We find that three subnets deviate significantly from others with drastically lower percentage of query–response pairs and higher percentage of unanswered queries. They belong to two organizations—the public DNS service and the European educational institute. In addition, the public DNS service is the only organization that suffers from a high percentage of unsolicited answers (15.2%).

Finally, we recompute the numbers for the percentage of query–response pairs, unanswered queries, and unsolicited responses after excluding the two anomalous organizations. We find these numbers to be 88.6%, 11.3%, and 0.03%, respectively. The low percentage of unsolicited responses also indicates that packet loss may not be a detrimental issue in the SIE data collection infrastructure outside of these two providers.

3) Server-Side Traffic Distribution: Besides the traffic distribution across monitored resolvers, we are also interested in where the traffic goes. By counting the destination IPs of DNS queries, we identify nearly 1.38 million distinct DNS authoritative servers, including 17 874 (1.3%) IPv6 servers. The authoritative servers span 30 129 distinct ASs, 229 distinct /8 subnets, 23 614 distinct /16 subnets, and 378 298 distinct /24 subnets.

The distribution of DNS traffic across the 1.38 million DNS servers follows a heavy-tailed distribution. While the 500 busiest servers attracted nearly half (49.9%) of all DNS queries; 94.1% of DNS servers received less than 10 000 queries during the 2-week measurement period.

We further group the DNS servers into organizations, for which we first correlate the addresses with DNS records in our dataset to identify their domain names, then empirically label the domain names with organizations (e.g., we label addresses with `*.apple.com` and `*.mac.com` domain names as *apple*). Table II presents the 10 organizations that received highest DNS traffic volume in our dataset, which together absorbed 39.7% of all DNS queries.

TABLE III
DISTRIBUTION OF DNS LOOKUPS BY POPULAR QUERY TYPES. THE TABLE OMITS THE PERCENTAGE OF OTHER QUERY TYPES. THE PERCENTAGES FOR YEARS 2001, 2002, AND 2008 ARE FROM [25], [48], AND [17], RESPECTIVELY

Perspective	Year	A	AAAA	PTR	MX
Local	2012	66.2%	13.4%	11.1%	2.3%
Local	2001	60.4% - 61.5%	N/A	24% - 31%	2.7% - 6.8%
Root	2012	57.5%	26.6%	4.8%	0.2%
Root	2008	60%	15%	8.4%	3.5
Root	2002	55.5%	4.7%	19.9%	4.6%

B. Query Type Breakdown

The DNS protocol supports a variety of query types for different purposes. To summarize the most popular types, an A query translates a domain name into IPv4 addresses, an AAAA query translates a domain name into IPv6 addresses, an MX query translates a domain name into mail exchange hostnames, and a PTR query translates an IP address back to domain names. We examine how popular each query type is in the real world and measure how this distribution has changed over time. Table III shows the distribution of the four popular types of DNS queries in real-world traffic. Because we do not have access to legacy DNS traffic, we quote the numbers reported by Jung *et al.* [25], Wessels *et al.* [48], and Castro *et al.* [17] in the row of year 2001, 2002, and 2008, respectively. Jung *et al.* collected their data from local resolvers at MIT and KAIST. On the other hand, Wessels *et al.* and Castro *et al.* reported the distribution observed from only root servers.

From the Perspective of Local DNS Resolvers: After more than 10 years, the A query remains the most dominant DNS query type in the US and Europe, accounting for about 66.2% of total queries. This percentage remains stable with a slight increase after 10 years. With wider deployment of IPv6 protocol, the volume of AAAA queries (13.4%) has risen sharply. This query type did not exist 10 years ago. Meanwhile, the percentage of PTR queries has decreased from 24%–31% to 11.1%, and MX queries have decreased from 2.7%–6.8% to 2.3%. While the absolute number of queries has also grown significantly in the past 10 years, the growth of other query types is not comparable to that of AAAA queries.

From the Perspective of Root DNS Servers: We observe a similar trend with local perspective. The percentage of A query remains steadily high at root servers. The percentage of AAAA query has increased with time, while the percentage of PTR and MX query has decreased. However, the change is more drastic from the root perspective than from the local perspective. At root, the percentage of AAAA queries has increased by 466% from 2002 to 2012. In contrast, the percentages of PTR query and MX queries have shrunk by 76% and 94%, respectively, in the same time period.

Query Types in Different Organizations: Figs. 3 and 4 plot the distribution of query types across different organizations. Fig. 3 analyzes all the traffic at local resolvers, whereas Fig. 4 only analyzes queries at root servers. We observe that different organizations have different characteristics. In addition, the organization profiles from local perspective are highly diverse. The organization profiles from root perspective are more consistent. We find several organizations that exhibit drastically different patterns in comparison to others.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

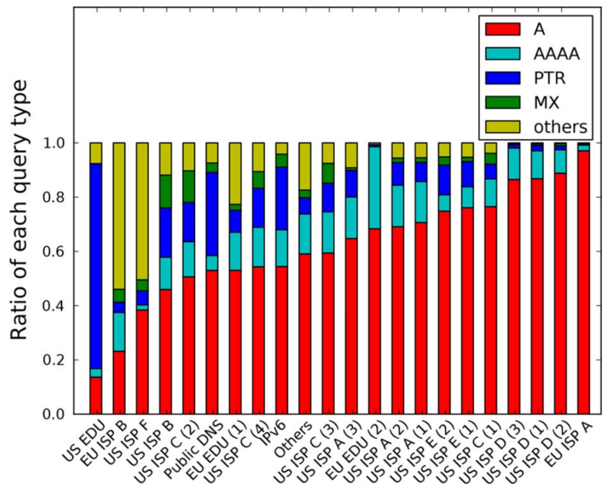


Fig. 3. Query type breakdown in different organizations from local perspective.

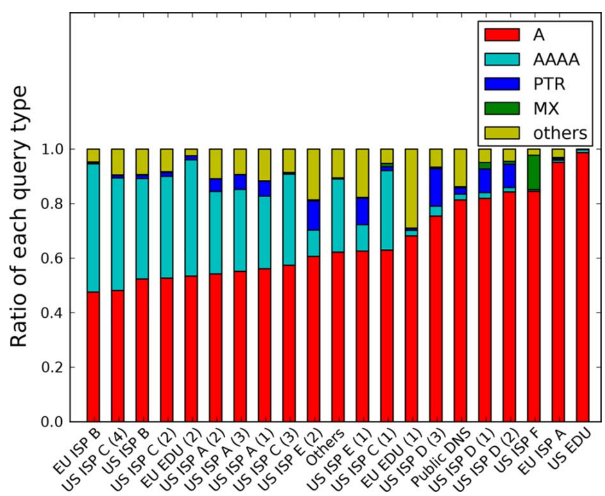


Fig. 4. Query type breakdown in different organizations from root perspective.

1) The US EDU subnet issues 75.5% of all PTR queries. However, the same US EDU subnet almost exclusively issues A queries (98.7%) to root servers.
2) The EU ISP A subnet issues almost exclusively A queries (97.1%). The same EU ISP A subnet issues almost only A queries (95.1%) to root servers as well.

C. DNS Query Success Rates

Next, we study the question of how many modern DNS queries return successful answers. We reuse the categorization method adopted by Jung *et al.* in [25]. In particular, DNS queries with successful answers are those having "NOERROR" as the return code in the response. We further divide the remaining queries into two categories: queries without response, and queries returning negative answers. Our definition of negative answer broadly includes all responses whose return code is *not* "NOERROR."

From the local perspective, the aggregated ratios of DNS queries with successful answers, negative answers, and no answers are 66.9%, 18.0%, and 15.1%, respectively. The overall ratios are similar to the result from 10 years ago, when Jung *et al.* reported that the percentages of answers with successful, negative answer, and unanswered queries were 64.3%, 11.1%, and 23.5%, respectively, in their MIT trace [25]. This suggests that many of the contributors to DNS queries with negative answers and no answers persist from a decade ago. From the root perspective, the ratio of unanswered query is 0, meaning that every query issued to the root servers is answered. It implies that root servers were always available during the measurement period. However, the percentage of successful answers returned by root servers, (i.e., referrals to nameservers that should know the queried hostnames) is significantly lower than that of other servers. Negative answers are returned by 54.0% of the queries issued to root servers. In comparison, in 2000 only about 2% of lookups to root return negative answers [25]. The sharply increased percentage of query with negative response at root servers may result from the high ratio of invalid traffic reaching them, as reported by multiple previous measurement studies at root servers [16], [17]. We further investigate the cause of failed query in Section III-D.

TABLE IV
FOUR QUERY TYPES CAUSING THE LARGEST NUMBER OF NEGATIVE ANSWERS FROM THE ROOT AND LOCAL PERSPECTIVE, RESPECTIVELY

Qtype	A	PTR	DNSBL	AAAA
Root perspective	66.0%	9.1%	0.2%	5.8%
Local perspective	50.9%	28.2%	7.2%	4.5%

D. Causes of Queries With Negative Answers

We first identify which query types cause the most negative answers. Table IV shows the top four types with their respective percentages. We find that A queries cause the vast majority of negative answers, in viewing from both the root and the local perspective. In comparison, in 2000 the dominant query type resulting in negative answers was the PTR query type [25]. Due to the shrinking percentage of PTR queries in our traffic, A queries have now become the dominant contributor to negative query responses. At the root servers, negative answers caused by PTR queries and DNSBL are much less common when compared to the local perspective.

Different query types also have differing ratios of negative answers. The ratio of DNSBL query with negative answers to the total number of DNSBL queries is 73.9%, which is significantly higher than any other query types due to the nature of blacklist lookup: Most lookups do not hit the blacklist, in which case an "NXDomain" response is returned. We further analyze DNSBL in Section III-D.5. Among the other three types, the ratio of PTR query with negative answers to the total number of PTR queries is 46.5%, which is higher than corresponding ratios for the A (14.8%) and AAAA (6.5%) query types.

Independent from query types, prior research has identified problematic query names that evoke negative answers [16], [17], [25], [48], including invalid TLDs, A-for-A, nonprintable characters, and private IP address in "PTR" query. We investigate them in detail in Sections III-D.1–III-D.4 and present their respective percentage in Table V. Note that the columns are *not* mutually exclusive.

1) Invalid TLD: Invalid TLD denotes the case when the queried hostname does not have a valid TLD. This may be

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE V
PERCENTAGE OF *INVALID TLD*, A-*FOR*-A, PRIVATE IP IN PTR QUERY, AND NONPRINTABLE CHARACTERS FROM THE ROOT AND LOCAL PERSPECTIVE, RESPECTIVELY

Perspective	Invalid TLD	A-*for*-A	Private IP	Non-printable char
Root	53.5%	0.4%	0.1%	3.2%
Local	1.2%	<0.1%	0.8%	0.2%

caused by either user typos or client-side application implementation bugs. Because the queried names do not exist, such queries will result in NXDomain as the response. Table V presents that 1.2% of the traffic from the local perspective contain an invalid TLD. However, 53.5% of the queries seen by root servers contain invalid TLDs. This observation, although highly skewed, seems reasonable because queries with invalid TLDs terminate the recursive resolution process at root servers, in the absence of valid TLD servers. Recall that from the root perspective, the total percentage of queries with negative answers is 54.0% (Section III-C). It means that *invalid TLD* has become the primary contributor to negative answers at root servers.

Multiple prior studies have investigated the prevalence of *invalid TLD* domains at root servers [16], [17], [48]. The percentages of *invalid TLD* domains reported in 2001, 2003, and 2008 were 20%, 19.53%, and 22.0%, respectively, which is stable. Surprisingly, it has sharply increased to 53.5% in 2012, from the perspective of our dataset. In addition, the resolvers issuing *invalid TLD* queries were widespread in all the major organizations that we monitor. Note that the above comparison only applies to root servers. The percentage of *invalid TLD* is low from the local perspective.

We summarize the most common invalid TLDs in Table VI. For each TLD, the table shows its count in million as well as its percentage among all invalid TLDs. We observe that a large number of invalid domains do not contain any dot. We put such domains in a special "no_dot" group, which is the second most popular form of invalid TLDs. Together with "local" and "belkin," these three invalid TLDs are far more popular than the other ones. ".local" is a pseudo-TLD that a computer running Mac OS X uses to identify itself if it is not assigned a domain name. Similarly, queries with ".lan," ".home," ".localdomain," ".loc," and ".internal" are likely used by other programs under certain circumstances. Nevertheless, these queries are meant to stay local and should not leak out to the Internet. "Belkin" is a famous brand that manufactures electronic devices. We suspect that queries with ".belkin" are generated by the device under the same brand due to misconfiguration. These are likely good candidates to be suppressed by local implementations. Although we have identified several likely causes of frequently appearing invalid TLDs, user typos can also result in invalid TLDs. In our data, the count of invalid TLDs exhibit a long-tailed distribution. More than 500 000 other invalid TLDs are used much less frequently.

2) A-for-A Query: A-*for*-A query denotes the case that the queried "hostname" is already an IP address. Because an IP address is also represented as a dot-separated string, the IP address A.B.C.D will be interpreted as having the TLD "D." Thus, A-*for*-A queries are a special case of *invalid TLD* queries.

TABLE VI
LIST OF 10 MOST FREQUENTLY QUERIED INVALID TLDS WITH COUNTS IN MILLIONS AND PERCENTAGE

TLD	Count (M)	%
local	68.4	21.9%
no_dot	52.2	16.7%
belkin	51.2	16.4%
corp	9.6	3.1%
lan	2.9	0.96%
home	2.3	0.74%
localdomain	1.7	0.54%
loc	1.5	0.48%
internal	1.4	0.45%
pvt	1.2	0.39%

In comparison to multiple prior works [16], [17], [48], we observe an interesting trend. The percentages of A-*for*-A seen by root servers reported in 2001, 2003, and 2008 was 12%–18%, 7.03%, and 2.7%, respectively. The decreasing trend continues in our data collected in 2012, where A-*for*-A only contributes 0.4% of the traffic. It indicates that most buggy implementations that caused this problem have been fixed. From the local perspective, the percentage of A-*for*-A is also negligible ($< 0.1\%$).

3) Private IP Address in PTR: RFC 1918 defines several networks that can be used internally but cannot be routed on the Internet: 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16. In theory, PTR queries to these IP addresses should be handled by the DNS administrators locally without leaking to the global Internet.

However, this is not the case in the real-world deployment. Previous studies revealed that at root servers, 7% and 1.61% of the queries are PTR queries with private IP addresses in a 2001 trace [16] and a 2002 trace [48], respectively. In our 2012 trace, only 0.1% of the queries issued to root servers have private IP addresses. This decreasing trend shows that more DNS administrators handle PTR queries with private IP addresses properly now. Viewing from the local perspective, 0.8% of the queries are PTR queries with private IP address.

4) Nonprintable Characters in Query Name: According to RFC 1035, domain names should only contain alphanumeric characters and hyphens, separated by dots. We follow the name in [17], [48] and refer to characters outside this scope as nonprintable characters. From the root perspective, 3.2% of the query names contain nonprintable characters. In comparison, this number in 2002 [48] and 2008 [17] is 1.94% and 0.1%, respectively.

Viewing from the local perspective, 0.2% of the queries contain nonprintable characters, and 0.9% of the failed queries contain nonprintable characters.

5) DNS Blacklists: DNS blacklist (DNSBL) is a popular method used by site administrators to vet domains for spam, malware, etc. Although DNSBL utilizes the DNS protocol, it does not translate between hostnames and IP addresses. Rather, site administrators use it to determine whether the target hostname is blacklisted by crafting the target hostname into a special URL under the blacklist provider's domain and issuing an A query. When the query reaches the blacklist provider's authoritative nameserver, the nameserver will send a response according to its own format. In popular DNSBL designs, the return

TABLE VII
FOUR TYPES WITH LARGEST NUMBER OF UNANSWERED QUERIES. THE FIRST ROW SHOWS THE PERCENTAGE IN UNANSWERED QUERIES. THE SECOND ROW SHOWS THE PERCENTAGE OF UNANSWERED QUERY WITHIN THE CORRESPONDING TYPE

Qtype	A	PTR	SOA	AAAA
Ratio across types	42.3%	17.8%	14.5%	14.0%
Ratio within type	9.5%	23.9%	87.6%	15.5%

code will be NXDomain (domain not exist) if the target hostname does not hit the blacklist. In particular, 73.9% of DNSBL queries return NXDomains, which gives DNSBL queries significantly higher failure odds than other query types.

The usage of DNS blacklists has been reported in [24]. DNS blacklists lookups accounted for 0.4% and 14% of lookups in their December 2000 trace and February 2004 trace, respectively. In 2012, DNSBL queries account for 1.7% of the lookups. The percentage is lower than year 2004, but higher than year 2000.

E. Unanswered Queries

As described in Section III-C, root servers do not incur any unanswered queries from our networks. Every query issued to the root servers is answered. All the unanswered queries are caused by other servers. In addition, the ratio of unanswered queries differs drastically across different organizations.

We further measure the correlation between unanswered query with different query types and show the results in Table VII. Due to the dominance of the A query, it also represents the largest portion of unanswered queries (42.3%). However, the probability of an A query without an answer is the lowest among the four listed types (9.5%). Interestingly, 87.6% of SOA queries do not get an answer.

F. TTL Distribution

The TTL field in the DNS responses informs the resolver how long it should cache the results. Fig. 5 shows the cumulative distribution of TTL values of three distinct record types in our DNS data: A, AAAA, and NS. Root servers very rarely reply with A or AAAA records in answer section, so we only plot NS record TTLs returned by root servers. In particular, A and AAAA records provide a direct mapping from a hostname to an IPv4 address and an IPv6 address, respectively, and the NS record provides a reference to the authoritative nameserver that should know the queried hostname when the nameserver being queried does not know the IP address of the queried hostname. We observe that NS records have much larger TTL values than A and AAAA records. This result is consistent with the result reported by Jung *et al.* from 10 years ago [25], except that AAAA record did not exist back then. Given that AAAA and A records play a similar role, which is to translate domain names to IP addresses, it is reasonable to observe that AAAA records and A records share similar TTL distributions. On the other hand, the longer TTL value of NS records is the key reason that keeps the load of DNS servers residing higher in the hierarchy manageable. Only 1.8% of the queries are issued to root servers in our trace because, in most cases, the client-side nameserver knows the authoritative nameserver using the cached NS records. If NS records have a much shorter TTL, the client-side nameserver

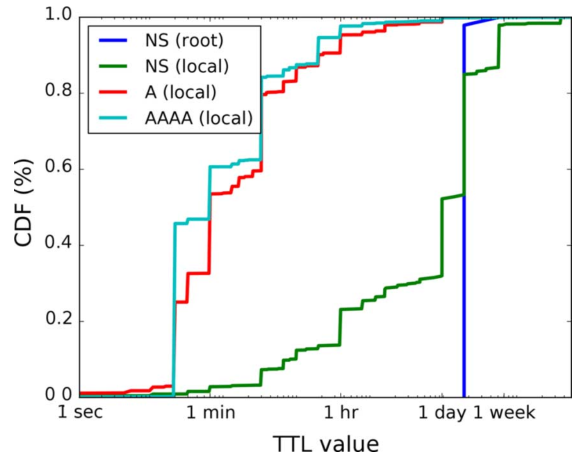


Fig. 5. Cumulative distribution of TTLs of NS record returned by root servers and three record types—A, AAAA, and NS—returned by other servers.

will need to query the root servers much more frequently. We also observe that the TTL of NS records returned by root servers is extremely regular: Almost all records have TTL of 2 days.

We further compare the TTL of A and NS records in 2012 and that in 2000 as reported in [25]. The TTL of NS records roughly remains stable. However, the TTL of A records in 2012 is much smaller. In 2000, only about 20% of A records have TTL less than 1 h. About 20% of A records have TTL larger than 1 day. In 2012, about 90% of A records have TTL less than 1 h, and almost 0% of A records have TTL larger than 1 day. This difference shows the wide deployment of CDN and other services that leverage short TTLs, which inevitably poses more pressure on the DNS infrastructure.

G. DNS Hosting

In DNS, a domain has one or more NS records indicating its delegation names, i.e., the names of its authoritative servers. A common practice for domain owners is to outsource name resolution to DNS hosting service providers such as Godaddy by pointing the NS records to names under corresponding hosting domains (e.g., Godaddy uses `domaincontrol.com` as its hosting domain).

In our dataset, we observe nearly 35.3 million distinct SLDs. Only a small fraction (2.1%) of these domains has in-domain NS names. The rest either host on different yet self-operated hosting domains, or most likely outsource name resolution to DNS hosting service providers. Most domains have NS names under 1 (85.0%) or 2 (10.2%) hosting domains. The market share of hosting domains also follows a heavy-tailed distribution; the top 100 hosting domains serve name resolution for 50.0% of the domains. Table VIII lists 10 of the most popular hosting domains observed in our dataset.

H. DNSSEC Deployment

The original design of DNS did not include data integrity protection. Attackers can fool end-users by forging DNS responses in a number of ways [13], [26]. The Domain Name System Security Extensions (DNSSEC) attempts to integrate Public Key Infrastructure (PKI) into DNS by assigning hierarchically delegated public keys to DNS domains and using

TABLE VIII
LIST OF 10 MOST POPULAR DNS HOSTING DOMAINS

Hosting Domain	# of Hosted Domains	%
domaincontrol.com	4,481,303	12.7%
worldnic.com	762,973	2.2%
hostgator.com	718,416	2.0%
bluehost.com	545,446	1.6%
rzone.de	471,652	1.3%
1and1.com	456,083	1.3%
ovh.net	450,216	1.3%
yahoo.com	449,760	1.3%
websitewelcome.com	415,534	1.2%
dreamhost.com	411,659	1.2%

digital signatures to provide data integrity and origin authentication. DNSSEC is backwards-compatible with the existing DNS. It is implemented by adding more DNS data types such as DNSKEY, DS, RRSIG, NSEC, and NSEC3 and a few flag bits in DNS packets [10]–[12]. The Internet coordination organization ICANN is now promoting the DNSSEC deployment. As of the last day of our dataset, 107 out of 317 TLDs have deployed DNSSEC [23].

In our dataset, we observe 151 179 domains with DNSSEC-related records, including 107 TLDs (all signed TLDs at that time), 135 924 SLDs, and 15 148 lower-level domains. In comparing with measurements by Osterweil *et al.* from 2005 to 2008 [39], who observed 871 signed "production" domains, the number of signed domains has increased by nearly 170 times.[1] The enormous increase demonstrates great progress of DNSSEC deployment in recent years.

Fig. 6 breaks down the number of signed domains in TLDs, which again shows a highly skewed distribution. The top 10 TLDs in Fig. 6 contribute 92.1% of all signed domains in our dataset. Some country code TLDs (notably `.nl`, `.cz`, `.se`, and `.br`) contribute significantly more signed domains than others, demonstrating their extraordinary efforts in deploying DNSSEC.

I. IPv6 Deployment

Besides DNSSEC, IPv6 is another major change currently happening in DNS. We have seen in Section III-A that the volume of IPv6 DNS traffic and the number of IPv6 resolvers are quite small. However, the deployment of IPv6 could still be considered progressful from the server-side perspective. From our dataset, we observe nearly a quarter of (24.1%, 8.5 million out of 35.3 million) of SLDs having IPv6 authoritative servers, an astonishing percentage compared to the numbers reported in Section III-A. This is mainly because popular DNS hosting providers are positive with adopting IPv6. Twenty-eight out of the top 100 hosting domains in our dataset have IPv6 authoritative servers, which make their customers resolvable from IPv6. The high percentage of IPv6 enabled domains suggests that ISPs and other DNS resolution service providers such as GoogleDNS and OpenDNS should support IPv6 in the iterative process of DNS resolution.

[1]Osterweil *et al.* empirically excluded signed domains that were likely to be deployed only for testing. We do not apply such division as only few domains are seemingly testing domains from our empirical inspection.

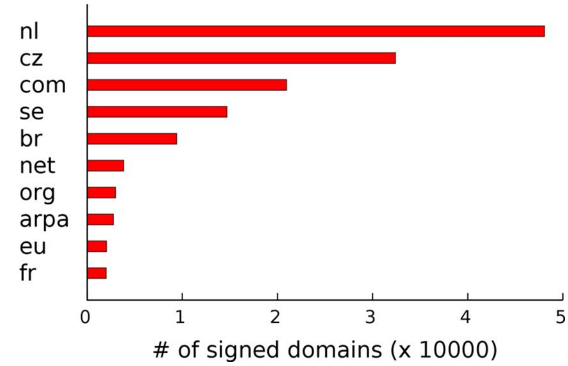


Fig. 6. Number of signed domains breakdown in TLDs.

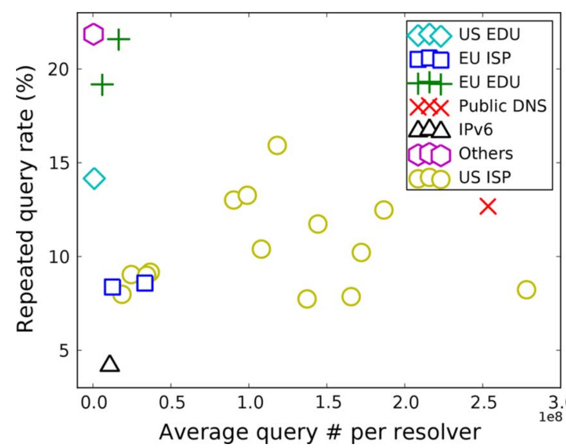


Fig. 7. Repeated query ratio and average number of queries per resolver for major organizations.

J. Repeated DNS Queries

Multiple previous studies of root DNS servers have revealed that over 56%–85% of queries observed at root servers are repeated [17], [48]. These studies further identified that misconfigured or abusive clients mainly caused these astonishingly high numbers. Ideally, a "normal" resolver should not issue many, if any, repeated queries to authoritative servers because of the effect of caching. However, our dataset shows that this is not the case—a considerable portion of DNS queries from "normal" resolvers could still be considered repeated. In this section, we analyze the prevalence and explore potential reasons behind the repeated query behavior of resolvers in more detail.

1) Simulation Methodology: For our analysis, we simulate an infinite resolver cache while replaying the captured DNS traffic. If the query returns an A, AAAA, or PTR record, the resolver knows the IP address of the queried domain or the domain for the queried IP address. It should not issue a query for the same domain or IP address before the TTL expires. If it issues such a query, we count it as a repeated query.

We find that from the perspective of our resolvers, the percentage of repeated queries that is issued to the root servers and other authoritative servers is 12.0% and 8.8%, respectively. The ratio of repeated query varies significantly across different organizations. We plot the hourly repeated query ratio in addition to the overall query volume for major organizations in Fig. 7. Although some organizational subnets have a repeated query rate

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

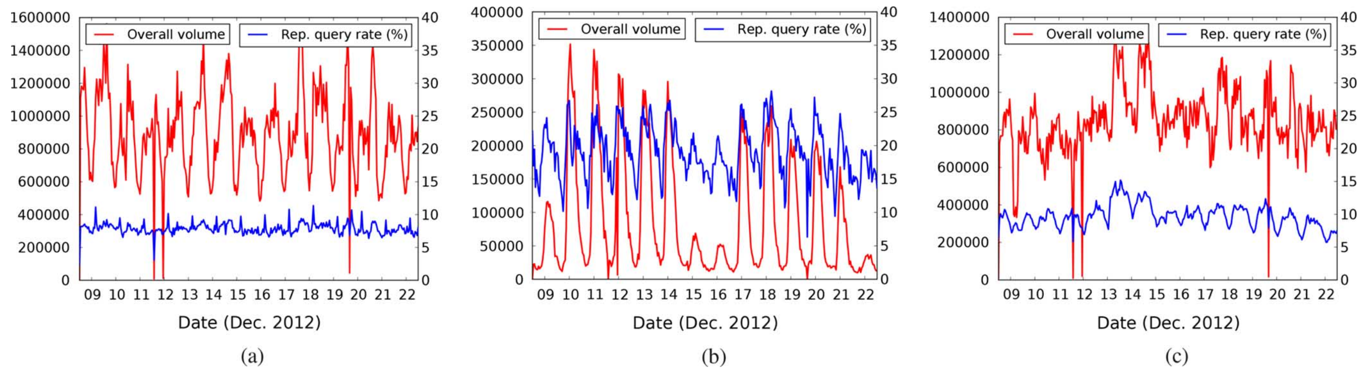


Fig. 8. Hourly repeated query ratio and overall DNS query volume for typical resolvers. (a) US ISP A (subnet 3). (b) EU EDU (subnet 1). (c) Public DNS Service.

of 20% or higher, their traffic volume is low. The repeated query rates for the largest subnets lie between 10% and 15%.

2) Hourly Plot of Repeated Query Ratio: The three resolvers shown in Fig. 8 exhibit very different characteristics. The university resolver [Fig. 8(b)] has the highest repeated query rate. Meanwhile, it also exhibits a strong positive correlation (p-value <0.001) between the repeated query rate and the query volume (i.e., the repeated query rate rises when the query volume rises). In addition, its overall query volume shows a clear diurnal pattern and weekly pattern. The traffic peaks appear during business hours of each day. Much more DNS traffic occurs during weekdays and less traffic during weekends.

The commercial ISP resolver [Fig. 8(a)] has a repeated query rate that varies between 5% and 10% during most of the days. The overall query traffic also exhibits a strong diurnal pattern, i.e., the traffic volume rises during nighttime and falls during daytime. It reflects the typical network usage of a residential network. However, we do not observe strong weekly pattern. In addition, a strong positive correlation (p-value <0.001) between the overall query volume and the repeated query rate exists.

The public DNS resolver [Fig. 8(c)] has fluctuating repeated query rate ranging from 5% to 15%. Because its users span different time zones, we naturally observe neither diurnal patterns nor weekly patterns from its overall query volume. Although hard to observe from the plot, statistical tests also indicate a strong positive correlation (p-value <0.001) between its overall query volume and its repeated query rate.

While many resolvers exhibit strong positive correlation between the query volume and the repeated query rate, it is not always the case. The former suggests that cache eviction plays an important role in the volume of repeated queries. The higher the query volume is, the higher the repeated query rate will be. We further observe that resolvers within a /24 subnet show high homogeneity (i.e., either all of them or none of them exhibit strong correlation, with very few exceptions). We omit these graphs due to space considerations. This reflects on the administrative policies within /24 subnets, the choice and configuration of network and DNS software.

3) Possible Causes: To further understand the cause of these repeated queries, we perform additional analysis to separate repeated queries that were issued in close proximity (the remainder could be attributed to cache eviction). We find that over 75% of repeated queries (across all resolvers) are due to related queries issued in close temporal proximity, and the remainder are likely due to cache evictions at the resolver. We investigate two popular resolver implementations BIND (9.9.2-P1) and Unbound (1.4.16), as well as the behaviors of OpenDNS and GoogleDNS, from which we distill a few possible implementation-related factors that cause repeated queries in close temporal proximity.

- *CNAME chain sanitization*: When a response includes multiple records forming a CNAME chain, both BIND and Unbound issue extra queries to verify the trustworthiness of the chain. This is an intentional security enhancement to counter the Kaminsky attack [26], which could cause repeated queries and increased response times. Nearly 20% of A and AAAA queries in our dataset were eventually responded to with CNAME answers, which makes CNAME chain sanitization contribute to about 40% of all repeated queries in our simulation.
- *Concurrent overlapping queries*: A resolver could issue repeated queries if it receives two *overlapping* queries in close proximity. Two queries are considered overlapping if they belong to *either* of the two cases: 1) they request identical name; or 2) some parts of their delegation chain or CNAME chain are identical. If the identical segment is missing in the cache, the resolver will send two identical requests, which will be counted as repeated query. Implementing birthday attack protection [47] can help mitigate this effect. We observe that both BIND and Unbound have implemented birthday attack protection, but interestingly GoogleDNS and OpenDNS do not strictly suppress identical queries. When probing GoogleDNS and OpenDNS with identical queries, we observed repeated queries from same resolver instances (IP addresses), although the numbers of repeated queries are less than our identical probes. One possible explanation is that Google and OpenDNS implement birthday attack protection per thread, therefore identical queries processed by mutiple threads on same instance could still trigger repeated queries.
- *Premature retransmissions*: We found that Unbound takes an arguably aggressive retransmission strategy, waiting for only one round-trip time before it retransmits the request. BIND is more conservative and has a minimum retransmission timeout of 800 ms. In our local experiments, we observed that Unbound issued several times more repeated queries than did BIND due to its premature retransmission timer.

- *Resolver quirks*: Resolvers might also have some implementation quirks (or bugs) that could trigger repeated queries in some cases. We have found that, in certain cases, BIND will resolve expired NS names twice before replying to client queries, resulting in repeated queries and increased response times. Given the complexity of the name resolution process, we suspect similar vagaries could lurk in resolver implementations.

IV. Malware Domain Group Detection

In this section, we present our approach to detect previously unknown malicious domains by simply using temporal correlation in DNS queries. The key intuition is that DNS queries are not isolated from each other. For any DNS query, the underlying process that generates it is likely to generate other related queries. For example, when a browser loads a Web page, it starts by querying the page's domain name, assuming it is not cached already. After the browser starts to render the page, it will generate additional DNS queries for other domain names whose content is embedded or linked in this page. This applies to malware as well. For example, drive-by exploits typically involve a long redirection chain before the occurrence of an exploit. Malware frequently uses domain generation algorithms (DGAs) as a rendezvous technique to search for command and control updates. Hence, we propose to detect malicious domain groups by using the temporal correlation among DNS queries, given some well-known seed malicious domains (also known as anchor points).

One of the key differentiators between our work and recent malicious domain detection work using DNS traffic [7], [8], [15] is the ability to detect malicious domains groups. We also only need only a small number of malware domains as seeds instead of a large training set. In addition, our intuition of DNS query correlation is general, so that our approach can detect different types of correlated domain groups, including but not limited to phishing, spam and scam campaigns, DGA-generated domains, redirection links, and so on. The ability to detect malicious domains in general also differentiates our work from existing work targeting specific types of correlated domains [9], [28], [43], [51].

Detecting correlated malicious domain groups using the DNS traffic collected from recursive resolvers is a challenging task. The difficulty rises from two major factors. *First*, the DNS queries are quite noisy in the sense that we observe a mixture of queries belonging to many different groups. We will also frequently fail to observe some queries that should have been in the group because of DNS caching. *Second*, the traffic volume is high. With about 80 million DNS queries per hour, conventional approaches that are able to discover correlated groups like clustering will not scale.

In order to make the problem tractable, we introduce the notion of "*anchor malicious domains*." Instead of searching in the entire DNS corpus, we only target domains that are correlated with the anchor domains. Given one anchor domain, we discover a group of additional malicious domains that are related with it. The processing of different anchor domains is mutually independent. This design benefits our detection approach with high applicability. It can work as long as at least one anchor domain is available. Thus, the bar to apply our approach is much lower than those systems that require a comprehensive labeled training set. In addition, parallelizing our approach for large-scale computation is straightforward.

In particular, we devise a multistep approach to discover correlated domain groups for each anchor domain. We describe the steps in detail in Sections IV-A, IV-B.1, and IV-B.2, respectively.

A. Coarse Identification of Related Domains

We represent the notion of correlation with co-appearance (i.e., a domain is considered to be correlated with the anchor domain if it is frequently queried together with the anchor from the same recursive resolver). We set a time window threshold T_{w} to restrict the search scope. Given an anchor domain, we extract the domain segment with the anchor domain in the middle according to the window size. All domains in the segment are considered as related domain candidates.

We quantify how closely the candidate domain is related with the anchor domain using two metrics derived from the idea of TF-IDF [31], a metric used widely in information retrieval to measure the importance of a term in a document, given a collection of documents. Let us consider the set of anchor domains to be A. Given a query domain d, a segment s corresponding to an anchor domain a (note that there can be multiple segments corresponding to an anchor domain), and a total set of n segments S, the TF-IDF-based metric has two components: 1) the term frequency $m_{\mathrm{tf}} = n(d,s)$, where n is a function indicating how many times the domain d occurs in the segment s; and 2) the inverse document frequency $m_{\mathrm{idf}} = |S|/|\{s \in S : d \in s\}|$, which measures how rare the domain d is across the set of segments S by computing the ratio of the total number of segments to the number of segments in which the domain occurs. The final TF-IDF score is the product of m_{tf} and m_{idf}.

Note that if the candidate domain is popularly queried in the DNS traffic, its m_{tf} value is expected to be large no matter whether it is related with the anchor domain or not—this will be counteracted by m_{idf} in the score, which down-weights popular domains. For the domains truly correlated with the anchor domain, we expect both its m_{tf} and m_{idf} values to be large. We set two thresholds—T_{tf} is the minimum value of m_{tf}, and T_{idf} is the minimum value of m_{idf}. Given an anchor domain, we extract all domain segments containing it, and then compute the m_{tf} and m_{idf} values for all domains that appear in the segments—we keep the domains with *both* values passing the corresponding thresholds to get a coarse identification of the group of domains related to the anchor domain.

B. Finer Identification of Related Domains

To get a more precise identification of domains related to the anchor domains, we first cluster domains according to their *pattern of co-occurrence* with the anchor domain.

1) Domain Clustering: Let us consider the set S_{a} of domain segments that have anchor domain a at their center point (note that there can be multiple such domain segments for any anchor domain). Let $f(a, S_{\mathrm{a}})$ denote the number of times a occurs in S_{a}. Each domain d in S_{a} is represented as a Boolean vector

v of dimension $f(a, S_{\mathrm{a}})$, where v_i is set to 1 if d co-occurs close (within a small window) to the ith occurrence of the anchor domain a in S_{a}, and 0 otherwise. We then cluster the vectors corresponding to each domain in S_{a} using XMeans [42] clustering, with squared Euclidean distance as the clustering distance metric. Note that XMeans is a partitional clustering algorithm like KMeans, which additionally selects the number of clusters automatically. In clustering models, it is possible to increase the likelihood by adding parameters, but that may overfit the data. We use XMeans with Bayesian Information Criteria (BIC) as the model complexity cost, which gives a penalty term proportional to the number of parameters in the model—XMeans finds the number of clusters that trade off the increased data likelihood with the increased penalty term. Each cluster in the output of XMeans groups together domains that share a common pattern of co-occurrence with a (e.g., a cluster may have domains that co-occur with only the first and second occurrence of the anchor point a, but not with other occurrences of a in S_{a}).

2) Domain Group Extraction: After clustering the domains related to an anchor domain, we further process the domain segments surrounding the anchor domain. We break each domain segment into multiple subsegments according to the cluster result, where each subsegment is created from the domains in a particular cluster. Note that the subsegment size is smaller than or equal to the cluster size because part of the cluster may not appear in that particular segment.

We use two filters to further refine subsegments—the first filter T_{freq} is based on domain frequency, while the second filter T_{size} is based on the size of the subsegment. Small subsegments with infrequent domains are more likely to have benign domains that pass the co-occurrence-based relatedness checks but actually share little commonality with the anchor domain.

The subsegments corresponding to the anchor domains form the refined domain groups (i.e., related domains) for the anchor domains—they are considered to be potentially malicious, and hence prime candidates for further analysis.

C. Evaluation

We use one day's worth of the DNS traffic to evaluate the malicious domain group detection technique. The data were collected on Dececember 16, 2012, and contain 1.82 billion DNS query–response pairs.

1) Evaluation Methodology: The module needs known malicious domains as anchors as input. We visit three blacklists: Malware Domain Block List [1], Malware Domain List [30], and Phishtank [3]. We choose these three blacklists instead of other popular ones because they provide their blacklisted domain database including timestamp for download. We select all domains that are blacklisted on the same days of the data used for the experiment as anchor domains. We obtain 129 anchor domains using this method. Note that although we use these three blacklists to obtain the anchor malicious domains, our approach is not limited to these three blacklists. Rather, this method can be used as long as some initial anchor domains are available.

Next, we need to label the detected domains as either malicious or legitimate to measure the detection accuracy. Labeling all domains in our dataset is impractical due to the huge volume. Hence, we only label the domains that are identified in the *coarse identification of related domains* (according to Section IV-A).

TABLE IX
NUMBER OF IDENTIFIED DOMAINS AFTER THE COARSE RELATED DOMAIN IDENTIFICATION STEP, AND THE NUMBER OF LABELED MALICIOUS AND BENIGN DOMAINS

Anchor Domain #	Coarse Related Domain #	Malicious Domain #	Benign Domain #
129	25373	16601	8772

We conduct a two-step process to label the domains as follows.

1) *Blacklist matching*: We match the detected domains against five popular external blacklists, including Malware Domain Block List [1], Malware Domain List [30], Phishtank [3], WOT (Web of Trust) [4], and McAfee SiteAdvisor [2]. If a domain is listed as malicious by any one blacklist, we will confirm it as malicious.
2) *IP address comparison*: If a domain resolved to the same IP address with a known malicious domain confirmed in the first two steps, we also confirm it as malicious. Because DNS name resolutions may contain multiple steps (e.g., a CNAME record that reveals the canonical name followed by an A record that translates into IP addresses), we build a directed graph to represent the name resolution results for all the detected domains. Next, we use standard graph traversal to find all detected domains that resolve to the same IP address with known malicious domains.

We label a domain as malicious if *any* of the above steps confirms it. Any domain that cannot be confirmed is conservatively labeled as benign, although some of them look very suspicious. Our data labeling approach is strict. Hence, our evaluation may overestimate the false positive rate. (We make this design choice because the damage of false alarms on legitimate domains is greater than missing malicious domains.)

Table IX presents the result of Step 1 (coarse related domain identification), as well as the number of malicious domains that we can label based on the identified domains. We observe that, first, domain co-appearance is an effective way to discover more malicious domains given anchor domains. On average, each anchor domain is expanded to 128 malicious domains. Second, the coarse identification includes a large number of benign domains as well. This is expected because the DNS traffic is noisy in nature. However, this does *not* mean that our approach incurs 8772 false positive domains. This is only the intermediate result after the first step described in Section IV-A. Our approach contains two more steps to further refine the detection result.

2) Detection Accuracy: In order to understand how the values of different thresholds affect the detection accuracy, we apply Step 2 (fine-grained identification of related domains), systematically tune the thresholds, measure the system performance with different values, and plot the result in Fig. 9. In our experiment, we find that setting threshold $T_{\mathrm{tf}} = 2$, and $T_{\mathrm{idf}} = 0.05$ produces significantly better results than higher values, so we only show the varying detection accuracy when these two thresholds are fixed at such values. Due to space constraint, we do not show the other cases. We vary T_{freq} from

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

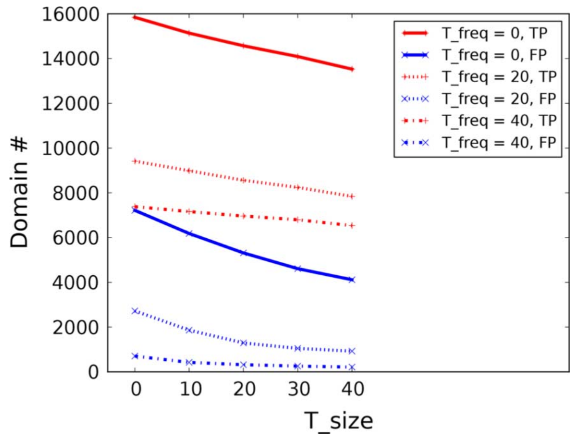


Fig. 9. Number of TPs and FPs in the detection result, varying the minimum domain frequency and minimum segment size thresholds.

0 to 40, and $T_{\rm size}$ from 0 to 40. We observe a steep drop in true positive number when $T_{\rm freq}$ increases from 0 to 40. In the mean time, the number of false positive domains also decreases quickly. We observe a similar trend when we vary the $T_{\rm size}$ threshold value. A larger threshold causes both the number of true positives and the number of false positives to decrease. As detection modules are typically tuned toward a low false positive rate, we find $T_{\rm freq} = 40$ and $T_{\rm size} = 20$ to be a good threshold choice. With this setting, this module detects 6890 previously unknown malicious domains (true positives), with 258 false positive domains. The detection precision achieves 96.4%. On average, each anchor domain is expanded to 53 previously unknown malicious domains. During real-world deployment, the operator who runs the malware domain group generation system will determine whether he prefers a tighter or a looser threshold. A tight threshold produces fewer false positives, but also discovers fewer malicious domains. A loose threshold does the opposite.

3) Detection Speed: The most time-consuming step in the malicious domain group detection process is to isolate the domain queries that are related with known anchor domains from the raw DNS data, so that we do not need to process other unrelated domain queries. In our experiment on the December 16, 2012, data, this preprocessing takes 40 689 s. After the preprocessing, the dataset that needs to be processed becomes much smaller than the raw DNS data, so subsequent steps run much faster. In particular, the coarse related group identification step takes 224.5 s. The domain clustering step takes 345.7 s. The domain group extraction step takes 266.9 s.

4) Comparison to Blacklists: We compare our approach to the five blacklists that we used in Section IV-C.1 to evaluate how much faster our approach can detect malicious domains. Since Web of Trust (WOT) [4] and McAfee SiteAdvisor [2] do not list the time when a domain is blacklisted, this experiment requires to repeatedly check detected domains with the blacklists to obtain the approximate blacklisted time. Hence, we repeat the detection process on January 20, 2013, data and compare the detection result against blacklists. The result is that 63.6% of the detected malicious domains are blacklisted on or before January 22, 2013. We keep monitoring the blacklists for an additional week, and find that 7.0% more detected domains are blacklisted. The remaining 29.4% detected domains are never blacklisted, but are confirmed as malicious by IP address comparison. The results demonstrate that our approach can detect malicious domains significantly faster than existing blacklists.

V. Related Work

DNS Measurement Studies: Many prior studies have measured the performance of the DNS infrastructure. Multiple measurement studies conducted at root servers reported that a large percentage of traffic at root servers is invalid [5], [16], [17], [48]. In particular, Brownlee *et al.* discovered that 60%–85% of queries to the F-root server are repeated [16]. Castro *et al.* analyzed traffic collected from multiple root servers and reported that 98% of the traffic is invalid [17]. Castro *et al.* confirmed in a later study that a low fraction of busy clients (0.55%) generate the most invalid traffic at root servers [5]. We cross-compare some of these same results from the perspective of a globally distributed resolver set to assess the persistence of such problems. Our vantage point provides a different perspective and greater opportunity for understanding the root cause of certain phenomena.

Jung *et al.* analyzed SMTP traffic with DNS blacklist lookups [24]. In this work, we compare the DNS blacklist usage in 2012 to their reported findings. Ager *et al.* used active probing techniques to compare local DNS resolvers with public DNS services like GoogleDNS and OpenDNS in terms of latency, returned address, and so on, by actively issuing DNS queries from more than 50 commercial ISPs [6]. Otto *et al.* studied the impact of using public DNS resolvers instead of local resolvers on the network latency of CDN content fetching [40]. Liang *et al.* measured and compared the latency of root and TLD servers from various vantage points [29]. Our measurement study has a different goal from theirs. In particular, we study the performance of recursive DNS resolvers. We do not cover the client-perceived DNS performance in our study.

DNS Performance Studies: Jung *et al.* characterized the DNS traffic obtained from two university sniffers and evaluated the effect of different TTL values with trace driven simulations [25]. Pang *et al.* measured DNS server responsiveness from the vantage points inside a large content distribution network [41], finding that a significant fraction of LDNS resolvers do not honor TTLs. Wessels *et al.* measured how the cache policy of different DNS software affects the number of DNS queries by trace driven simulations [49]. Bhatti *et al.* conducted experiments to reduce the TTL of A records on university DNS resolvers and found a low increase in DNS traffic [14]. While the focus of this paper is on the broad high-level characteristics of DNS data such as the overall distribution of query types and failures, prevalence of repeated queries, etc., revisiting the implications of caching and DNS performance in greater depth in the context of the SIE dataset is future work.

DNS Malware Studies: Researchers have recently proposed using DNS traffic statistics to identify malicious domains [7], [8], [15]. Notos [7] and EXPOSURE [15] build models of known legitimate domains and malicious domains and use these models to compute a reputation score for a new domain that indicates whether the domain is malicious or

TABLE X
SUMMARY OF COMPARISONS TO PRIOR MEASUREMENT STUDIES

Compared Feature	**Prior Work**	**Result Summary**
Root perspective		
Private IP address in PTR Query	[16, 48]	The ratio in 2001, 2002 and 2012 is 7%, 1.61% and 0.1%, respectively.
Invalid TLDs	[16, 17, 48]	The ratio is steady 20% in 2001, 2002 and 2008, but rises to 50% in 2012.
A-*for*-A	[16, 17, 48]	The ratio is 12%, 7.03%, 2.7% and <0.1% in 2001, 2002, 2008 and 2012.
Non-printable character in query name	[17, 48]	The ratio drops from 1.94% (year 2002) to 0.1% (year 2008), but rises to 3.2% in 2012
DNS query type breakdown	[17, 48]	The ratio of AAAA queries increases from 4.7% (year 2002), to 15% (year 2008) to 26.6% (year 2012)
Queries with negative answer	[25]	14.7%, 27.3% and 54% in Jan 2000, Dec 2000 and 2012, respectively.
Local perspective		
DNS query type breakdown	[25]	There are no AAAA queries in 2000, but 13.4% of all queries are AAAA queries in 2012.
Queries with negative answer	[25]	The ratio rises from 11.1% (year 2000) to 18.0%.
Queries with no answer	[25]	The ratio drops from 23.5% (year 2000) to 15.1% (year 2012).
Cause of negative answer	[25]	PTR and A queries cause the most negative answer in 2000 and 2012, respectively.
TTL distribution	[25]	TTLs of A records in 2012 are much smaller than those in 2000.
DNSBL query ratio	[24]	0.4%, 14.0% and 1.7% in year 2000, 2004 and 2012, respectively.

legitimate. Their objective is to compute reputation scores for domains by using a large set of features, whereas we try to extract malware domain groups by just using the temporal features. Kopis [8] aims to detect malware domains by monitoring network traffic at the "upper-levels" of the DNS hierarchy. Our approach is fundamentally different from Kopis in terms of the vantage point (monitoring at the TLD as opposed to the RDNS servers), features in use, and operational requirements. Konte *et al.* use active techniques to identify malicious fast-flux DNS domains from spam data [27]. Rajab *et al.* actively probed open DNS caches to test the prevalence of known malicious domains [44]. In contrast, we employ passive analysis on domain queries at resolvers. Hao *et al.* examine TLD servers to cluster newly registered domains based on registration information and lookups [21]. In [50], Yadav *et al.* proposed several statistical metrics to identify randomly generated domain names of botnets, and subsequently they improved on their techniques by examining failed DNS queries [51], [52]. Perdisci *et al.* [43] proposed a technique to detect malicious flux service networks through passive analysis of recursive DNS traces. Unlike our approach that looks at the co-occurrence and sequence in domain names, their approach is focused on fast-flux features, where multiple IP addresses are multiplexed to a single domain name using DNS responses with short TTLs. Sato *et al.* [45] extended blacklists using the co-occurence relation between DNS queries. We operate on a much more global and larger dataset and our analysis is complicated by the fact that our data stream occludes the client IP addresses, as we observe aggregated data streams emanating from the resolver, which necessitates more sophisticated analysis.

VI. CONCLUSION

In this paper, we conduct a comprehensive measurement study with more than 26 billion DNS query–response pairs collected from 600+ global DNS resolvers. Besides reaffirming some findings in published work, our results reveal some significant differences. We witness the demise of A-for-A queries and a significant rise in AAAA queries. We also find that queries for invalid TLDs are responsible for more than 99% of queries with negative answer observed at root servers and that TTLs of A records become much smaller than a decade ago. In Table X, we summarize comparisons made in this paper with five prior studies and highlight our results. Note that this table only includes a subset of our measurement results that are directly comparable to results in prior work.

Our findings can help implementation, deployment, and configuration of DNS software, Web sites, and other applications. First, because of the increase of AAAA queries for IPv6 addresses, Web sites should take IPv6 support into account. DNS providers and ISPs should also consider to support IPv6 for DNS resolution because of the high percentage of IPv6 resolvable domains. The high failure ratio of PTR queries implies that some DNS administrators pay less attention to configuring reverse mappings from IP addresses to domain names. The high rate of invalid TLD queries to root servers suggests that client-side implementations should differentiate local names (used only in Intranets) with global domain names. Our analysis of repeated queries reveals that complementary security enhancements of resolvers could have nonnegligible effects on DNS resolution, suggesting that more evaluations should be conducted before the wide adoption of such features. Our analysis also reveals several possible optimization to suppress unnecessary queries.

Furthermore, we propose a novel approach that isolates malicious domain groups from temporal correlation in DNS queries, using a few known malicious domains as anchors. On average, this approach achieves more than 96% detection accuracy while producing more than 50 previously unknown malicious domains for every known malicious anchor domain.

ACKNOWLEDGMENT

The authors are grateful to P. Vixie and the SIE contributors for providing access to this extraordinary data source. They would also like to thank Y. Cao for his help with some of their early analysis scripts.

REFERENCES

[1] "Malware domain blocklist," [Online]. Available: http://www.malwaredomains.com/
[2] McAfee, Santa Clara, CA, USA, "McAfee SiteAdvisor," [Online]. Available: http://www.siteadvisor.com/
[3] "PhishTank," [Online]. Available: http://www.phishtank.com/
[4] Web of Trust, "Safe Browsing Tool—WOT," [Online]. Available: http://www.mywot.com/
[5] "Understanding and preparing for DNS evolution," in *Proc. Traffic Monitoring Anal.*, 2010, vol. 6003, Lecture Notes in Computer Science.
[6] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing DNS resolvers in the wild," in *Proc. 10th ACM SIGCOMM Internet Meas. Conf.*, 2010, pp. 15–21.

[7] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for DNS," in *Proc. 19th USENIX Security Symp.*, 2010, p. 18.
[8] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon, "Detecting malware domains at the upper DNS hierarchy," in *Proc. USENIX Security Symp.*, 2011, p. 27.
[9] M. Antonakakis *et al.*, "From throw-away traffic to bots: Detecting the rise of DGA-based malware," in *Proc. USENIX Security Symposium*, 2012, p. 24.
[10] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS security introduction and requirement," RFC 4033, 2005.
[11] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "Protocol modifications for the DNS security extensions," RFC 4035, 2005.
[12] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "Resource records for the DNS security extensions," RFC 4034, 2005.
[13] S. M. Bellovin, "Using the domain name system for system break-ins," in *Proc. 5th USENIX UNIX Security Symp.*, Berkeley, CA, USA, 1995, vol. 5, pp. 18–18.
[14] S. Bhatti and R. Atkinson, "Reducing DNS caching," in *Proc. Comput. Commun. Workshops*, Apr. 2011, pp. 792–797.
[15] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "EXPOSURE : Finding malicious domains using passive DNS analysis," in *Proc. 18th Annu. Netw. Distrib. Syst. Security Symp.*, San Diego, CA, USA, 2011.
[16] N. Brownlee, K. Claffy, and E. Nemeth, "DNS measurements at a root server," in *Proc. IEEE GLOBECOM*, Nov. 2001, vol. 3, pp. 1672–1676.
[17] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy, "A day at the root of the internet," *Comput. Commun. Rev.*, vol. 38, no. 5, pp. 41–46, Sep. 2008.
[18] P. B. Danzig, K. Obraczka, and A. Kumar, "An analysis of wide-area name server traffic: A study of the internet domain name system," in *Proc. ACM SIGCOMM*, 1992, pp. 281–292.
[19] C. J. Dietrich, "Feederbot—A bot using DNS as carrier for its C&C," 2011 [Online]. Available: http://blog.cj2s.de/archives/28-Feederbot-a-bot-using-DNS-as-carrier-for-its-CC.html
[20] Farsight Security, Inc., "Farsight Security archive," 2013 [Online]. Available: https://archive.farsightsecurity.com/
[21] S. Hao, N. Feamster, and R. Pandrangi, "Monitoring the initial DNS behavior of malicious domains," in *Proc. ACM SIGCOMM Internet Meas. Conf.*, 2011, pp. 269–278.
[22] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, "Measuring and detecting fast-flux service networks," in *Proc. Netw. Distrib. Security Symp.*, 2008.
[23] ICANN, "TLD DNSSEC report," [Online]. Available: http://stats.research.icann.org/dns/tld_report/
[24] J. Jung and E. Sit, "An empirical study of spam traffic and the use of DNS blacklists," in *Proc. 4th ACM SIGCOMM Internet Meas. Conf.*, 2004, pp. 370–375.
[25] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, "DNS performance and the effectiveness of caching," *IEEE/ACM Trans. Netw.*, vol. 10, no. 5, pp. 589–603, Oct. 2002.
[26] D. Kaminsky, "It is the end of the cache as we know it," BlackHat USA, 2008.
[27] M. Konte, N. Feamster, and J. Jung, "Dynamics of online scam hosting infrastructure," in *Proc. Passive Active Meas. Conf.*, 2009, pp. 219–228.
[28] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang, "Knowing your enemy: Understanding and detecting malicious web advertising," in *Proc. ACM Conf. Comput. Commun. Security*, 2012, pp. 674–686.
[29] J. Liang, J. Jiang, H. Duan, K. Li, and J. Wu, "Measuring query latency of top level DNS servers," in *Proc. Passive Active Meas. Conf.*, 2013, pp. 145–154.
[30] Malware Domain List, "Malware domain list," [Online]. Available: http://www.malwaredomainlist.com
[31] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge Univ. Press, 2008.
[32] "MaxMind, Inc.," Waltham, MA, USA [Online]. Available: http://www.maxmind.com/
[33] P. Mockapetris, "Domain names—Concepts and facilities," RFC 1034, 1987 [Online]. Available: http://www.ietf.org/rfc/rfc1034.txt
[34] P. Mockapetris, "Domain names—Concepts and facilities," RFC 882, 1983 [Online]. Available: http://www.ietf.org/rfc/rfc882.txt
[35] P. Mockapetris, "Domain names—Implementation and specification," RFC 1035, 1987 [Online]. Available: http://www.ietf.org/rfc/rfc1035.txt
[36] P. Mockapetris, "Domain names—Implementation and specification," RFC 883, 1983 [Online]. Available: http://www.ietf.org/rfc/rfc883.txt
[37] C. Mullaney, "Morto worm sets a (DNS) record," 2011 [Online]. Available: http://www.symantec.com/connect/blogs/morto-worm-sets-dns-record
[38] E. Osterweil, D. McPherson, S. DiBenedetto, C. Papadopoulos, and D. Massey, "Behavior of DNS' top talkers, a .com/.net view," in *Proc. Passive Active Meas. Conf.*, 2012, pp. 211–220.
[39] E. Osterweil, M. Ryan, D. Massey, and L. Zhang, "Quantifying the operational status of the DNSSEC deployment," in *Proc. 8th ACM SIGCOMM Internet Meas. Conf.*, 2008, pp. 231–242.
[40] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante, "Content delivery and the natural evolution of DNS: Remote DNS trends, performance issues and alternative solutions," in *Proc. ACM SIGCOMM Internet Meas. Conf.*, 2012, pp. 523–536.
[41] J. Pang *et al.*, "Availability, usage, and deployment characteristics of the domain name system," in *Proc. 4th ACM SIGCOMM Internet Meas. Conf.*, 2004, pp. 1–14.
[42] D. Pelleg *et al.*, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Mach. Learning*, 2000, vol. 1, pp. 727–734.
[43] R. Perdisci, I. Corona, D. Dagon, and W. Lee, "Detecting malicious flux service networks through passive analysis of recursive DNS traces," in *Proc. Annu. Comput. Security Appl. Conf.*, 2009, pp. 311–320.
[44] M. A. Rajab, F. Monrose, and N. Provos, "Peeking through the cloud: Client density estimation via DNS cache probing," *Trans. Internet Technol.*, vol. 10, no. 3, Oct. 2010, Art. no. 9.
[45] K. Sato, K. Ishibashi, T. Toyono, and N. Miyake, "Extending black domain name list by using co-occurrence relation between DNS queries," in *Proc. LEET*, 2010, p. 8.
[46] J. Spring, L. Metcalf, and E. Stoner, "Correlating domain registrations and DNS first activity in general and for malware," in *Proc. Securing Trusting Internet Names*, 2011.
[47] J. Stewart, "DNS cache poisoning—The next generation," 2003.
[48] D. Wessels and M. Fomenkov, "Wow, that's a lot of packets," in *Proc. PAM*, San Diego, CA, USA, Apr. 2003.
[49] D. Wessels, M. Fomenkov, N. Brownlee, and K. Claffy, "Measurements and laboratory simulations of the upper DNS hierarchy," in *Proc. Passive Active Netw. Meas. Workshop*, 2004, pp. 147–157.
[50] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proc. 10th ACM SIGCOMM Internet Meas. Conf.*, 2010, pp. 48–61.
[51] S. Yadav and A. N. Reddy, "Winning with DNS failures: Strategies for faster botnet detection," in *Proc. SecureComm.*, 2011, pp. 446–459.
[52] Z. Zhu, V. Yegneswaran, and Y. Chen, "Using failure information analysis to detect enterprise zombies," in *Proc. Securecomm*, 2009, pp. 185–206.

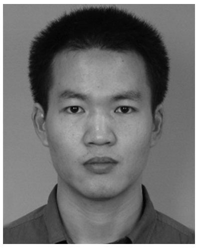
Hongyu Gao received the B.S. degree from Peking University, Beijing, China, in 2008, and the Ph.D. degree from Northwestern University, Evanston, IL, USA, in 2013, both in computer science.

He is now a Software Engineer with Google, Mountain View, CA, USA. His research interests span the areas of online social network security and spam detection.

Vinod Yegneswaran received the A.B. degree from the University of California, Berkeley, CA, USA, in 2002, and the Ph.D. degree from the University of Wisconsin, Madison, WI, USA, in 2006, both in computer science.

He is a Senior Computer Scientist with SRI International, Menlo Park, CA, USA, pursuing advanced research in network and systems security. His current research interests include DNS traffic analysis, SDN security, and anti-censorship technologies.

Dr. Yegneswaran has served on several NSF panels and program committees of security and networking conferences, including the IEEE Security and Privacy Symposium.



Jian Jiang received the B.E. and M.E. degrees from Xi'an Jiaotong University, Xi'an, China, in 2002 and 2005, respectively, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2014, all in computer science.

He is now a Research Specialist with the University of California, Berkeley, CA, USA. His research interests include security analysis of network protocols and systems and network measurement.

Yan Chen received the Ph.D. degree in computer science from the University of California, Berkeley, CA, USA, in 2003.

He is a Professor with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA. Based on Google Scholar, his papers have been cited over 7000 times and his h-index is 34. His research interests include network security, measurement, and diagnosis for large-scale networks and distributed systems.

Prof. Chen won the Department of Energy (DoE) Early CAREER Award in 2005, the Department of Defense (DoD) Young Investigator Award in 2007, and the Best Paper nomination in ACM SIGCOMM 2010.

Phillip Porras received the M.S. degree in computer science from the University of California, Santa Barbara, CA, USA, in 1992.

He is an SRI Fellow and a Program Director of the Internet Security Group in SRI's Computer Science Laboratory, Menlo Park, CA, USA. He has participated on numerous program committees and editorial boards, participates on multiple commercial company technical advisory boards, and holds 12 US patents. He continues to publish and conduct technology development on numerous topics including intrusion detection and alarm correlation, privacy, malware analytics, active and software defined networks, and wireless security.

Shalini Ghosh received the Ph.D. degree in computer engineering from the University of Texas at Austin, Austin, TX, USA, in 2005.

She is currently a Senior Computer Scientist with the Computer Science Laboratory, SRI International, Menlo Park, CA, USA. At present, she is a Visiting Scientist with Google Research, Mountain View, CA, USA. She has served on the program committees of many conferences (e.g., KDD, NAACL), review boards of journals, NSF panels, and was invited to be a guest lecturer in an electrical engineering and computer science course at the University of California, Berkeley, CA, USA. Her main area of research interest is applications of machine learning and data mining for natural language understanding and dependable computing. She is also interested in probabilistic relational models, e.g., Markov logic networks.

Dr. Ghosh and her coauthors received the Best Paper Award from the 19th IEEE PRDC in 2013.

Haixin Duan received the B.E. and M.E. degrees from Harbin Institute of Technology, Harbin, China, in 1994 and 1996, respectively, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2000, all in computer science.

He is a Professor with Tsinghua University and a Visiting Scholar with the International Computer Science Institute (ICSI), Berkeley, CA, USA. His research interests include network security and network measurement.