Detecting Malicious Activity with DNS Backscatter Over Time

Kensuke Fukuda John Heidemann Abdul Qadeer

***Abstract*—Network-wide activity is when one computer (the *originator*) touches many others (the *targets*). Motives for activity may be benign (mailing lists, CDNs, and research scanning), malicious (spammers and scanners for security vulnerabilities), or perhaps indeterminate (ad trackers). Knowledge of malicious activity may help anticipate attacks, and understanding benign activity may set a baseline or characterize growth. This paper identifies *DNS backscatter* as a new source of information about network-wide activity. Backscatter is the reverse DNS queries caused when targets or middleboxes automatically look up the domain name of the originator. Queries are visible to the authoritative DNS servers that handle reverse DNS. While the fraction of backscatter they see depends on the server's location in the DNS hierarchy, we show that activity that touches many targets appear even in sampled observations. We use information about the queriers to classify originator activity using machine-learning. Our algorithm has reasonable accuracy and precision (70–80%) as shown by data from three different organizations operating DNS servers at the root or country-level. Using this technique we examine nine months of activity from one authority to identify trends in scanning, identifying bursts corresponding to Heartbleed and broad and continuous scanning of ssh.**

I. Introduction

Network-wide activity is when one computer (the *originator*) touches many others (the *targets*). Malicious activity is a growing problem in the Internet. *Spammers* exploit compromised computers to send mail, one part of the gray-market ecosystem [31]. *Scanners* walk the IP address space, for research [26], [19], in academic [18] or commercial [36] vulnerability scanning, as cyber criminals searching for vulnerabilities [17] (or joyriding [6]), or as nation-states espionage [25], [29]. Knowledge of malicious activity may help anticipate attacks [12]. Non-malicious activity is of concern as well: ad trackers and content-delivery networks (CDNs) also interact with many computers. Studies of benign activity help set a baseline [56] or characterize growth (results like Calder et al. [10], but without active probing).

Unfortunately, it is difficult to understand the scope of these potential threats because of the Internet's decentralization. While firewalls observe activity directed at one network, and a few security providers aggregate data from a few hundreds [3], by design the Internet has no central vantage point to detect widespread activity. Prior work has used Domain Name System (DNS) traffic to assess misbehavior seen in specific networks or resolvers [58], [27], [41], [57], [2], but this work does not generalize to network-wide activity. Darknets [38], [35], [56], [13], [14], [17] and honeypots (for example, [42]) are effective at understanding network-wide activity, but they miss targeted scans (scanning only Alexa top websites [17]), and new large darknets are unlikely given IPv4 full allocation and the huge IPv6 space. Search engines gather information about activity that appears in the public web, but information is unstructured and may be delayed by indexing [50]. (§ VII has detailed related work.)

This paper identifies a new source of information on network-wide activity: *DNS backscatter*, the reverse DNS queries triggered by such activity (see Figure 1 and § II). Activities of interest are those that touch many Internet devices, including malicious or potentially malicious activity such as spamming and scanning, as well as widespread services such as CDNs, software updates, and web crawling. These activities trigger *reverse DNS queries* as firewalls, middleboxes, and servers (*queriers*) resolve mapping of the IP address of the originator to DNS name in the process of logging or host-based authentication. Authoritative DNS servers provide a point of concentration of these queries that allows detection of large activities. Since backscatter occurs mostly as automated processes, and we consider only originators with many queriers, our approach avoids traffic from individuals and so has minimal privacy concerns. Since backscatter is generated by the targets of network activity, not the originator, an adversarial originator cannot prevent its generation.

The contribution of this paper is to identify DNS backscatter as a new concentrator of information about network-wide activity (§ II). We show that this source is noisy and attenuated by caching, but careful interpretation of aggregate queriers allows construction of a new type of sensor (§ III) that detects large, automated activities that touch many targets. We use machine learning (ML) to classify the originators of each activity into broad groups (spammers, scanners, and several types of commercial activity) with reasonable precision (70–80%) and robustness (§ IV). We study long-term accuracy of our algorithm and show that malicious and benign originators exhibit different activity behaviors and evolution. We also study how different training strategies affect classification accuracy and show that it is necessary to aggressively adapt training according to changing world (§ III-E and § V; these sections are new beyond our prior work [22].) Finally, we use backscatter and classification to examine world-wide scanning activity from three sources and up to nine months (§ VI). We characterize the "natural footprint" of different activities for two days at two sensors at the DNS root and one national-level server. We also examine nine months of data, showing that there is a continuous background of scanning, and identifying

K. Fukuda is with National Institute of Informatics and Sokendai, Tokyo, Japan e-mail: kensuke@nii.ac.jp.

J. Heidemann and A. Qadeer are with Information Sciences Institute and Department of Computer Science at University of Southern California, Los Angeles, CA, USA email: johnh@isi.edu; aqadeer@isi.edu.

Manuscript received April 25, 2016; revised December 10, 2016.

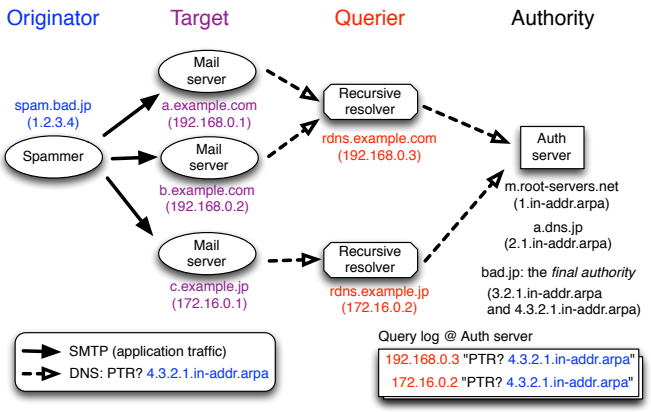

Fig. 1: The process behind a DNS backscatter sensor: an originator sends mail to targets, causing queries that are observed at the authority.

increased scanning following announcements of vulnerabilities such as Heartbleed [39]. Through this study, our work helps characterize scanning and other network-wide activity; our approach may also serve to support detection and response.

II. DNS Backscatter's Potential

DNS backscatter is the set of reverse DNS queries observed by a DNS *authority* as the result of a *network-wide activity*. In such activity, an *originator* interacts with many *targets*. Targets usually do not directly interact with the authority, but do so through a *querier* that acts on their behalf. Figure 1 shows these players. We use DNS backscatter observations to identify and classify the originators.

Origination of Network Activity: We look for network activity where an *originator* interacts with many *target* hosts. Examples can be legitimate (large mailing lists and web crawlers), malicious (spam), or perhaps in-between (scanning and peer-to-peer sharing). Our goal is to infer and classify the originator (§ III) and to understand the size of its *footprint* (from [49])—how many targets it touches. In Figure 1, the originator is spam.bad.jp from IP address 1.2.3.4, and it sends spam emails to (a.example.com and its neighbors; our goal is to identify an otherwise unknown originator.

An originator is a single IP address that touches many targets. In the application classes we study, originators interact with their targets. In principle the originator could be the victim of spoofed traffic (such as a DNS server as part of an amplification attack); we have not identified such originators in our data. (For a few application classes, such as software update service, ad-trackers, and CDNs, the activity is initiated by the target and some other service co-located with the target does a reverse-DNS query.)

At the Target: The activity prompts the target's interest in discovering the originator's domain name from its IP address: a reverse DNS mapping that causes a *querier* to make a reverse query if the result is not already cached. This query may be for logging (as by firewalls), to perform domain-name based access control, or to characterize the originator (for example, mail servers that consider the sender's hostname as part of anti-spam measures). The querier is defined as the computer that does resolution of the reverse name. That the target and the querier may be the same computer, or the querier may be a dedicated recursive resolver shared by several targets.

A reverse DNS query will have the query name (QNAME 4.3.2.1.in-addr.arpa in Figure 1) with the reversed IP address of the originator, and will ask for a reverse Internet name (QTYPE PTR, QCLASS IN). In this example, this query returns spam.bad.jp, but we do not use this response or even depend on the existence of a reverse name for the originator.

At the Authority: An authoritative DNS nameserver (the *authority*) will reply to this query. We analyze DNS traffic at the authority to infer information about the originator.

DNS is a hierarchical, distributed database where many authoritative servers cover different levels and parts of the namespace. DNS uses aggressive caching at nearly all levels of the system. These factors mean that there are multiple authorities that will see different reverse-DNS traffic with different amounts of sampling due to caching and coverage.

The ideal place to observe reverse queries is the *final authority* directly responsible for the originator. In typically the originator's company or ISP (serving 3.2.1.in-addr.arpa in our example), the final authority will see *all* queriers. However, authorities higher in the DNS hierarchy will see some attenuated fraction of queriers and can discover unknown originators. We consider several authorities: a national-level, top-level server (we examine Japan), and root nameservers (we consider B and M, two of the 13), as well as a final authority. Our national-level view sees traffic only for originators in address space delegated to that country (perhaps 2.1.in-addr.arpa in our example). Roots potentially see all originators, but caching of the top of the tree (in-addr.arpa and 1.in-addr.arpa) filters many queries, and visibility is affected by selection algorithms that favor nearby DNS servers. Some *recursive* servers, such as those run by large ISPs or large open DNS providers, will also have broad visibility.

Backscatter Evaluation: We observe reverse queries at the authority and use this backscatter to classify the originator's application. Queries directly provide the IP address of the originator, but we use that only to group queriers (we ignore the originator reverse DNS record, if any). Instead we use information about *queriers* to suggest the originator's application class. Each individual query has little information, but the *collective behavior* of all queriers generating backscatter indicates the nature of an originator's activity. For example, spammers send mail to many mail servers, generating backscatter from these servers and anti-spam middleboxes. Similarly, large-scale network scanning causes firewalls at targets log the originator's domain name.

The number of reverse queriers also provides a rough idea of the size of the originator's activity. Unfortunately, the DNS's caching policies make it extremely difficult to quantify originator traffic rates—we caution that querier counts only *approximate* activity sizes.

An adversarial originator may seek to hide its activity. Spreading activity over many originators will reduce detection,

but other aspects of our approach depend on targets and thus are outside the control of the originator.

Privacy: Analysis of DNS traffic raises potential privacy issues, since it often originates from activity by individuals. Our approach minimizes these concerns for several reasons. First, the data sources we use intrinsically mask the visibility and identity of individuals. Caching heavily attenuates all queries seen by the authority, and a shared cache obscures the identity of any individual. We see network-wide activity only because of its many targets, while activity of any given individual is extremely unlikely to appear. Second, authorities have little or no direct contact with individuals due to indirection from recursive resolvers. Finally, while raw data at an authority is a mix of individual and automated traffic, the reverse queries we consider is nearly all automated. Humans typically use DNS to map names to addresses; almost all reverse queries are from automated sources. (As evidence, we examined ten minutes of B-Root queries in B-post-ditl. Using not-found replies [NXDomain] as evidence of typos, only 8 of 126,820 reverse queries against in-addr.arpa are not-found, while about half of the forward queries for IP addresses are not-found.)

Our analysis and results also raise minimal concern. Input is effectively anonymized through our the abstraction and aggregation in our analysis, as DNS becomes feature vectors about collective queriers. Our results identify and classify originators in bulk, and we examine only the most prolific originators—a set that is necessarily automated to reach that scope. Finally, when examining specific originators to validate our results, we would anonymize any that present personally-identifiable information (PII). None so far show any PII.

Our work has been institutionally reviewed and approved as non-human-subjects research (USC IIR00001718). We see minimal privacy risk in this work, and substantial benefit in better understanding network-wide activity such as scanning.

III. Methodology: Backscatter to Activity Sensor

We next summarize our approach to make DNS backscatter a sensor for network-wide activity: collecting DNS data at the authority, classifying data for each originator with features based on the queries and queriers, and then clustering originators into activities. This approach is grounded in training based on expert-labeled data, possibly retrained periodically. Figure 2 shows how these steps fit together for classification practice, and shows the training that goes into classification.

III-A Data Collection: Queries at the Authority

We begin with data collection at the authority. We observe all DNS queries at arrival and retain only reverse DNS queries (PTR queries made against in-addr.arpa). Each query results in an (originator, querier, authority) tuple; we identify the originator from the QNAME, and the querier and authority are the source and destination of the DNS packet.

Queries may be obtained through packet capture on the network or through logging in DNS server itself. DNS packet capture techniques are widely used [53], [20]. DNS logging is supported in most servers, and tools such as dnstap define standard logging formats [52]. Centralized DNS information is available today through services such as SIE [47]. We describe the datasets used in this paper in § III-G.

III-B Interesting and Analyzable Originators

We focus our classification on only those originators that are *interesting*, carrying out activities that affect large parts of the network, and that are *analyzable*, providing enough information from queriers that we can reasonably classify them. We identify interesting originators with data over time intervals that are long enough to include a significant number of analyzable originators, and then identifying the most interesting among them based on how many targets they touch.

We generate a *feature vector* v^o for each originator o over some time interval lasting d days. The time interval under consideration must be long enough that the originator touches enough targets to allow inference of the originator's application class. We consider 20 or more unique queriers per originator to be sufficient for analysis. We select the time interval for each dataset to meet this goal (details of the datasets are shown in § III-G). DITL datasets [16] last 1–2 days, and for each we generate one feature vector for each originator across the whole dataset (d is the dataset duration, 36 or 50 hours). For the M-sampled dataset, d is 7 days, thus generating an array of feature vectors v_i^o where $i \in [0, 35]$, and for B-multi-year, d is 1 day.

Originators that touch many targets have larger, more interesting activities. To identify the most prolific originators, we record the number of unique (originator, querier) combinations in each time interval. We rank originators by number of unique queriers in the time interval and retain only the N originators with the most unique queriers. We count originators (§ VI-C) keeping only originators with 20 or more unique queriers.

III-C Queries to Originator Features

We define static and dynamic features for each originator. *Static* features are derived from the domain names of the queriers and infer the originator's intention based on the names of computers they touch. *Dynamic* features use spatial and temporal patterns in queries.

Querier domain names provide these static features, often determined by specific keywords:

home computers with automatically assigned names like home1-2-3-4.example.com. The name includes digits of the IP address and a keyword: ap, cable, cpe, customer, dsl, dynamic, fiber, flets, home, host, ip, net, pool, pop, retail, user.

mail servers like mail.example.com. Keywords: mail, mx, smtp, post, correo, poczta, send*, lists, newsletter, zimbra, mta, pop, imap.

ns nameservers like ns.example.com. Keywords: cns, dns, ns, cache, resolv, name.

fw firewalls like firewall.example.com. Keywords: firewall, wall, fw.

antispam anti-spam services like spam.example.com. Keywords: ironport, spam.

www web servers like www.example.com

ntp NTP servers like ntp.example.com

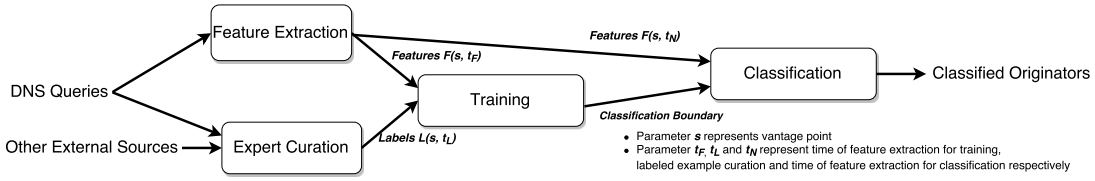

Fig. 2: Training and classification process.

cdn CDN infrastructure, including suffix of Akamai, Edgecast, CDNetworks , LLNW.

aws Amazon AWS, including suffix of amazonaws.

ms Microsoft Azure, checked with web page.

google IP addresses assigned to Google, as confirmed by SPF record in DNS.

other-unclassified a name not matching the categories above.

unreach cannot reach the authority servers.

nxdomain no reverse name exists.

Domain names may match several static features (for example, mail.google.com is both google and mail). We match by component, favoring matches by the left-most component, and taking first rule when there are multiple matches. (Thus both mail.ns.example.com and mail-ns.example.com are *mail*).

We list the fraction of queriers that match each feature into a vector of static features. For example, an originator where all queriers have "mail" in their domain names would have the mail feature as 1 and all others as zero. Another where a quarter have "mail" and the rest have "firewall" would have feature vector with (mail$=0.25$, firewall$=0.75$). We use the fraction of queriers rather than absolute counts so static features are independent of query rate.

We use these dynamic features to capture temporal and spatial aspects of the queries:

queries per querier (temporal) As a *rough* query rate, we compute the mean number of queries per querier.Differences in caching and TTLs prevent generate an exact rate, but this metric is roughly proportional to query rate.

query persistence (temporal) To show how often an originator is active, we count the number of 10-minute-long periods that include the originator.

local entropy (spatial) To see if an activity affects many people, we compute the Shannon entropy of /24 prefixes of all queriers.

global entropy (spatial) We compute the Shannon entropy of the /8 prefix of all querier IP addresses. Since /8 prefix are assigned geographically, wide variation here shows global activity.

unique ASes (spatial) the number of ASes across all queriers normalized by the total number of ASes that appear in the time interval. (ASes are from IP addresses via whois.)

unique countries (spatial) the number of countries across all queriers, normalized by the total number of countries that appear in the time interval. We determine country from the IP using MaxMind GeoLiteCity database [33].

queriers per country (spatial) unique queriers per country.

queriers per AS (spatial) unique queriers per AS.

To avoid excessive skew of querier rate estimates due to queriers that do not follow DNS timeout rules [54], [11], we eliminate duplicate queries from the same querier in a 30 s window. Some variation remains due to different caching times (TTLs) for different portions of the namespace. In some cases, such as fast flux domains, low TTLs will assist detection.

We selected the features and classes (in this section and the next) after iteration with our several datasets. Our choice of static features is based on our understanding of Internet naming conventions, and dynamic features to help distinguish target diversity. Different features and classes will produce different results (for example, although we omit details due to space, we see higher accuracy with fewer application classes), but our results show the general approach is sound. We hope to evaluate additional features in future work.

III-D Originator Features to Application Classes

Finally we classify each originator based on its feature vector using machine learning. Each feature vector has its constituent static and dynamic components. We identify a feature vector using the IP address of the associated originator. We use several standard machine-learning algorithms: a decision tree (Classification And Regression Tree; CART) [8], random forest (RF) [7], and kernel support vector machines (SVM) [45]. For non-deterministic algorithms (both RF and SVM use randomization), we run each 10 times and take the majority classification. These algorithms classify each originator into one of the following classes:

ad-tracker Servers implementing web-bugs (objects embedded in web pages) to track user for advertising.

cdn Computers that are part of public content delivery networks (Akamai, CDNetworks, etc.)

cloud Front-end hosts of cloud service such as Google map, Goggle drive, and Dropbox.

crawler Web crawlers by Google, Bing, Baidu, etc.

dns Large DNS servers

mail Mail servers that send mail to large mailing lists and webmail service

ntp Large Network Time Protocol (NTP) servers

p2p End-user computers that participate in peer-to-peer file sharing. (We added this late in our work; our analysis of root servers does not include it.)

push Push-based messaging services for mobile phones (Android and iPhone); typically TCP port 5223.

scan Internet scanners using ICMP, TCP, or UDP.

spam Computers (end-user or mail servers) that send spam to many destinations.

update Servers for software update distribution run by OS, computer, and printer vendors

Classification requires training. We label a subset of each dataset as ground truth, manually identifying the application class of 200–300 originators in each dataset. Our automated algorithm extracts feature vectors for these labeled examples. We input these feature vectors of labeled examples to our ML algorithm. Conceptually a ML algorithm puts each feature vector in an n-dimensional space and finds out an optimal classification boundary that separates out feature vectors of one application class from another. Our trained classifier uses classification boundary for finding application class of an originator. We present feature vectors of experimental data as input to our trained classifier. Classifier's output is a label (i.e. an application class) for each of the input feature vectors.

III-E Training Over Time

Classification (§ III-D) is based on training, yet training accuracy is affected by the evolution of activity—specific examples come and go, and the behavior in each class evolves. Change happens for all classes, but the problem is particularly acute for malicious classes (such as spam) where the adversarial nature of the action forces rapid evolution (see § V).

When the labeled dataset and observed features are both current, training provides the most accurate classification boundaries, (see Figure 2). In principle, one would continuously adapt both the labeled dataset and features. However, both evolve gradually over few weeks, so one might hold one or both fixed for some period of time. Training is affected by its relevance of the labeled dataset to the current vantage point, its size and timeliness of the labeled dataset, and the timeliness of the feature vectors. We next examine several combinations of updating inputs: fixing all three, fixing the labeled dataset and updating feature vectors, and growing the labeled dataset both automatically and manually.

Currently we always customize the labeled dataset for each vantage point. We find that each vantage point has distinct perspective and traffic mix. We start from a common labeled dataset of all originators, but add and remove examples to get a sufficiently large dataset for training purposes (typically we require about 20 examples in each class, and about 200 or more total examples).

Selecting labeled examples and training once: As a first step, one might identify a labeled dataset and then use it to train against current data once. We use this approach for short-term datasets, such as the DITL datasets of about two days (JP-ditl, B-post-ditl, M-ditl and M-ditl-2015). We generate the labeled dataset by iterating against what is detected and external sources (§ III-B), then we train against all features seen over the short dataset. With this training, the t_L and t_F in Figure 2 are constant.

Since a labeled dataset is built by an expert for a specific vantage point and time, short datasets use one-time training. However, datasets that last weeks or more require labeled data from multiple times, as we show in § V-B).

New feature values for fixed labeled examples and retraining often: As a first step to supporting long-duration measurements, we can retrain against new feature values, without changing the labeled dataset. We will later show that all classes change over time, but some classes are quite stable (87% for 8 months) and all are relatively stable over a few weeks (13% for 4 weeks). While the IPs that originate activity change slowly, exactly what they do tends to change more rapidly. We can take advantage of this difference by retraining with fresh observations (new feature vectors) even though we do not change or grow the labeled dataset. In practice, we typically retrain daily.

The advantage in retraining is that it updates the classification boundaries and improves accuracy. We show that this approach improves over never retraining (§ V-C), however, it cannot address the challenge of activity that migrates to new locations over time, and such migration can be rapid for malicious activity. One would need a mechanism to know when an activity stops using a specific labeled example originator and a new curation for diminishing application class might be needed.

Evolving labeled examples and retraining: The only way to accommodate the shift of activity to new originators at new IP addresses is to evolve the labeled dataset, adding new originators and pruning those that are no longer active. Such curation can be done automated or manually.

Automatic curation is attractive since manual curation is labor-intensive; we evaluate evolution of the labeled dataset in § V-A. If classification is correct, then we can use today's classification to grow the labeled dataset for tomorrow. This approach is very sensitive to classifier accuracy (see § V-C), so we can also check proposed new labels against external sources (for example, verifying newly identified spammers appear in Spamhaus' reputation system).

The gold standard for changing the labeled dataset is to employ a human expert. A human expert can review proposed labels, validate them against external sources, and employ judgment as to trends. We use this approach for dataset M-sampled, building a single labeled dataset with candidates taken from three dates, each about a month apart, and then retraining every day on current features.

III-F Constraints in Backscatter as a Data Source

While DNS backscatter provides a new tool to identify network-wide activity, it is an incomplete source about originators and their activity.

First, querier domain-names only provide limited information about targets. Domain names may be overly general, unavailable, or never assigned. Currently we see 14–19% of queriers without reverse names, and the fraction serves as a classification feature, but if *all* queriers lacked reverse names our approach would not work. Other network services use reverse names, such as anti-spam and network reputation services, finding as we do that they are imperfect but still useful. Our results use only backscatter, but we show that applications will benefit from combining it with other sources of information (such as small darknets) to overcome the limitations of querier names alone.

Second, the signal provided in DNS backscatter is spread over many authorities by DNS anycast, attenuated by caching in recursive DNS resolvers, and complicated by caching

durations (TTLs) that differ across the DNS hierarchy.These challenges prevent us from getting strong estimates on the rates of originator activity, but we show that they do not prevent their identification or classification.

Finally, an adversarial originator may try to defeat our analysis. Spreading traffic from an activity across many separate originating IP addresses or a longer duration reduces the signal. We cannot prevent this countermeasure, but it greatly increases the effort required by an adversarial originator. Fortunately our classification depends on queriers (not originators), and they cannot be manipulated by the originator.

In this paper we show that querier provides reasonable classification accuracy and backscatter's signal identifies hundreds of originators. We are working to refine our estimates of accuracy and sensitivity, and of course additional data sources, larger labeled ground-truth, and more refined classes may improve on our work.

III-G Datasets

This paper uses DNS datasets from three authorities: one national-level top-level domain and operators of two root servers as shown in Table I.

JP-DNS operates the .jp country code domain for Japan; we have data from their complete service (all anycast sites of their seven IP addresses providing service). We obtained data from two root server operators: B is a single site in the US west coast, and M operates 7 anycast sites in Asia, North America, and Europe. We use root datasets from the short-term, complete (unsampled) observations taken as part of the 2014 DITL collection [16] (for B-Root, shortly after 2014 DITL). We also use data for M-Root's 2015 DITL collection (§ IV-D). These root datasets are available to researchers through DNS-OARC.

For longitudinal analysis we draw on 9 months of data taken at the M-Root server. This dataset (M-sampled) is sampled with only 1 out of every 10 queries in deterministic manner; this sampling decreases our sensitivity but is part of their policy for long-term collection. We also use an 5-month unsampled dataset (B-long) taken from B-Root to evaluate for controlled experiments (§ IV-D), and more than 4-years unsampled dataset (B-multi-year) to evaluate long-term accuracy of our algorithm (§ V).

IV. Validation

Classification with machine learning requires useful features and accurate training data. Here we illustrate our features with several different type of originators, then identify external sources we use to label the ground truth for training and validation. We validate our approach by training on a random portion of the ground truth and testing the remainder, identifying specific datasets in each figure or table.

IV-A Distinguishing Different Classes of Originators

We first illustrate how the DNS backscatter allows us to differentiate the activity of six different originators: scan-icmp, scan-ssh, ad-tracker, cdn, mail, and spam. Figure 3 and Table II show their static and dynamic features.

Static features: Static features (Figure 3) distinguish several cases. We examine two scanners: *scan-icmp* is the

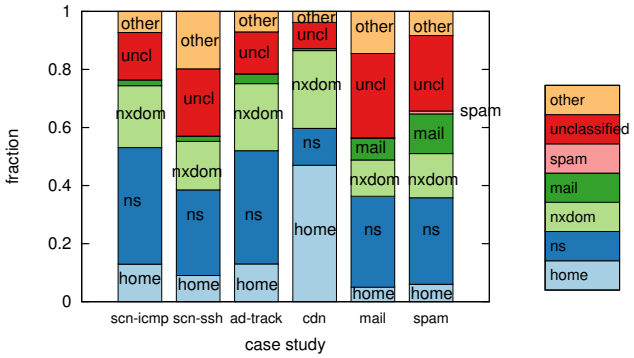

Fig. 3: Static features for case studies, derived from querier domain names. (Dataset: JP-ditl.)

Japanese site for a research scanner doing outage detection [43], while *scan-ssh* is an unknown party probing ssh (TCP port 22) on all computers. Scanners trigger many queries from shared nameservers (NS), as well as nxdomain, home, and static addresses. We believe this mix results from scanners walking all IP addresses and triggering firewall and logging reverse queries; the large number of NS queriers suggest these services often use an ISP's shared resolver, but not always.

Ad-trackers are servers triggered by user's web behavior. Similar to scanners, but class sees more shared nameservers, perhaps because it is queried by end-users's web browsers.

Our *cdn* case is an Akamai server based in Japan. It shows a *much* higher fraction of home queriers than others. We believe this shift results from heavy CDN use at homes.

Finally, *mail* and *spam* both show a much higher fraction of the mail feature, consistent with application. Although hard to see on the graph, spam shows more than twice the fraction of spam-related queriers (0.1% compared to 0.03%), indicating heavier involvement of spam-filtering and logging systems.

Dynamic features: Table II lists the distribution of some dynamic features for six originators. As with § III-C, query counts only approximate true rates because of caching, but we believe relative comparisons are appropriate for classification.

We see that the queries:querier ratio helps distinguish mail from spam. Global entropy demonstrates geographic dispersion of the queriers. Low global entropy for *cdn* reflects CDN selection algorithms that associate our Japan-based CDN server with Asian clients. Similarly, the global entropy *mail* is lower because the language of this specific list is Japanese. Queries per country also reflects geographic coverage, with high values of *ad-tracker*, *cdn*, and *mail* suggesting geographically constrained targets.

The examples in § III-C illustrate how these features help identify our application classes, and their overlap suggests the potential for machine learning as a framework to interpret them optimally. We quantify our approach in § IV-C.

IV-B Confirming Labeled Ground-Truth

We require ground truth with originators labeled into our twelve application classes (§ III-D) both to train our classifier and to evaluate its accuracy (§ IV-C).

In labeling ground truth we strive for *accuracy* over quantity because a mix of inaccurate originators will mis-train our

type	dataset	operator	start (UTC)	duration	sampling	queries ($\times 10^9$) (all)	queries ($\times 10^9$) (reverse)	qps ($\times 10^3$) (all)	qps ($\times 10^3$) (reverse)
ccTLD	JP-ditl	JP-DNS	2014-04-15 11:00	50 hours	no	4.0	0.3	22	1.8
root	B-post-ditl	B-Root	2014-04-28 19:56	36 hours	no	2.9	0.04	22	0.2
root	B-long	B-Root	2015-01-01	5 months	no	290*	5.14	22*	0.39
root	B-multi-year	B-Root	2011-07-08	4.16 years	no	2900*	51.4*	22*	0.39*
root	M-ditl	M-Root	2014-04-15 11:00	50 hours	no	8.3	0.06	46	0.3
root	M-ditl-2015	M-Root	2015-04-13 11:00	50 hours	no	9.9	0.07	55	0.4
root	M-sampled	M-Root	2014-02-16	9 months	1:10	36.2	1.5	1.6	0.07

TABLE I: DNS datasets used in this paper. (* indicates estimates.)

case	queries/ querier	global entropy	local entropy	queriers/ country
scan-icmp	3.3	0.83	0.92	0.006
scan-ssh	4.7	0.84	0.96	0.006
ad-track	2.3	0.85	0.94	0.017
cdn	4.4	0.48	0.97	0.018
mail	1.7	0.71	0.94	0.009
spam	3.4	0.85	0.95	0.005

TABLE II: Dynamic features for case studies. (Dataset: JP-ditl)

dataset	algorithm	accuracy	precision	recall	F1-score
JP ditl	CART	0.66 (0.05)	0.63 (0.08)	0.60 (0.06)	0.61 (0.06)
	RF	**0.78** (0.03)	**0.82** (0.05)	**0.76** (0.06)	**0.79** (0.05)
	SVM	0.73 (0.04)	0.74 (0.05)	0.71 (0.06)	0.73 (0.05)
B post-ditl	CART	0.48 (0.05)	0.48 (0.07)	0.45 (0.05)	0.46 (0.05)
	RF	**0.62** (0.05)	**0.66** (0.07)	**0.60** (0.07)	**0.63** (0.07)
	SVM	0.38 (0.11)	0.50 (0.14)	0.32 (0.13)	0.39 (0.13)
M ditl	CART	0.53 (0.06)	0.52 (0.07)	0.49 (0.06)	0.51 (0.06)
	RF	**0.68** (0.04)	**0.74** (0.06)	**0.63** (0.05)	**0.68** (0.05)
	SVM	0.60 (0.08)	0.68 (0.10)	0.52 (0.08)	0.59 (0.09)
M sampled	CART	0.61 (0.03)	0.65 (0.04)	0.58 (0.04)	0.61 (0.04)
	RF	**0.79** (0.02)	**0.82** (0.02)	**0.77** (0.03)	**0.79** (0.02)
	SVM	0.72 (0.02)	0.76 (0.03)	0.70 (0.03)	0.73 (0.02)

TABLE III: Validating classification against labeled ground-truth.

classifier. We draw on multiple sources of external data (blacklists, darknets, and manual investigation) to get reasonable coverage of most application classes, with 180 to 700 samples depending on the dataset (Table VI).

We use external sources to generate moderate to large lists of potential IP addresses for each application class, then intersect it with the top-10000 originators in dataset by the number of queriers. We then verify the intersection manually. We determine the verifying application class of each originator manually using the methods described in Appendix A. We added two categories (push and update) after examining all of the top 100 and 1000 largest originators, respectively.

IV-C Classification Accuracy and Algorithm Choice

We next carry out cross-validation, using random subsets of labeled ground-truth to test the classification accuracy and to compare three classification algorithms: Classification And Regression Tree (CART), Random Forest (RF), and Kernel Support-Vector Machines (SVM).

Classification accuracy: To evaluate a particular machine-learning algorithm, for each dataset we pick a random 60% of the labeled ground-truth for training, then test on the remaining 40% of data. We repeat this process 50 times, testing each subset with all three algorithms. For each run we compute accuracy ($(tp+tn)/all$), precision ($tp/(tp+fp)$), recall ($tp/(tp+fn)$), and F1-score ($2tp/(2tp+fp+fn)$), where tp: true positive, tn: true negative, fp: false positive, fn: false negative. Table III shows the mean of each metric over the 50 iterations, with standard deviations in smaller type.

Overall, we find that RF outperforms SVM and CART in most metrics. The accuracy of the best algorithm over each dataset is 0.7 to 0.8, suggesting classification from our limited data is not easy, but fairly strong given 12 categories (thus an expected 0.08 accuracy for guessing). We see mislabeling of application classes where the training data is sparse: *ntp*, *update*, *ad-tracker*, and *cdn* for JP-ditl. In particular, the classification ratio of *update* events is very low. Reducing the number of classes would improve accuracy, at the cost of less useful results. Improving the accuracy of classification across multiple classes with unbalanced training data is an active area of research in machine learning. Our current results show promise and will improve as the field progresses.

Similarly, for JP-ditl, *p2p* is sometimes misclassified as scan. Manual inspection shows that misclassified *p2p* actually sent traffic to dynamically-assigned ports in our darknets as well. We conclude that they are mis-behaving P2P-clients [32], [9] and these random probes are triggered by software bugs or by injection of random IP addresses by anti-P2P services.

Choice of authority: While still good, accuracy for B post-ditl and M-ditl, is slightly poorer (60–75%) because roots are attenuated and these datasets are short. M-sampled does better because it uses longer observations (7 days, not 2). JP accuracy is highest because it is unsampled and lower in hierarchy. We conclude that approach can be strong, but one must be careful to match the quality of the signal and training data.

Feature choice: In evaluating classification, we naturally would like to know what features are most discriminative. Random Forest provides this information, and Table IV lists the top discriminative features for two datasets, as determined by Gini coefficient [55]. Larger Gini values indicate features with greater discriminative power. We see that *mail*, *home*, *nxdomain*, and *unreach* are important static features in both datasets. *Query rate* and *global entropy* are important in different datasets, perhaps reflecting M-ditl's weaker global signal and JP-ditl's more regional signal. These results are consistent with intuitive case studies (§ IV-A).

Algorithm choice: Since Random Forest provides the best accuracy and is consistent (small standard deviations), we use

rank	JP-ditl feature	Gini	M-ditl feature	Gini
1	mail (S)	8.4	mail (S)	12.5
2	home (S)	7.9	ns (S)	8.3
3	spam (S)	6.3	unreach (S)	7.0
4	nxdomain (S)	6.2	query rate (D)	6.2
5	unreach (S)	5.2	home (S)	6.0
6	global entropy (D)	5.0	nxdomain (S)	5.8

TABLE IV: Top discriminative features. Classifier: RF.

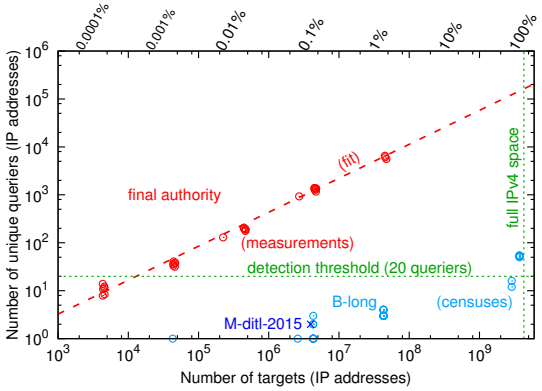

Fig. 4: Size of querier footprint due to controlled random network scans as observed at the final authority, in B-long and M-ditl data.

that algorithm to characterize originators in the next section, after retraining with the full labeled ground-truth data.

IV-D Controlled Experiments to Evaluate DNS Caching

Attenuation from DNS caching makes interpreting backscatter challenging. Such caching is difficult to model: first, the DNS resolver infrastructure can be quite complex [46], and is influenced by the target's choice of recursive resolver and authority. Second, DNS caches are kept for varying durations and caching of the resolution tree (in-addr.apra) will be caused by many competing users.

Attenuation: To estimate the number of queriers that respond to a large network event we conducted a controlled experiment where we probe a fraction of the IPv4 Internet from a host where we can monitor queries sent to the final reverse DNS server for the prober. We set the TTL of the reverse DNS record (PTR) to zero to disable or minimize caching (some resolvers force a short minimum caching period), thus we should observe all queriers triggered in response to our scan. We scan several protocols (ICMP, TCP port 22, 23, 80, and UDP port 53, 123) using ZMap [19], varying the fraction of the address space we scan from 0.0001% (4k addresses) to 0.1% (4M addresses) of the whole IPv4 space. The time required for the biggest scan (0.1%) was 13 hours. We run each scan up to 5 times. We also examine 8 full-internet scans (nearly 100% of unicast IP) taken with Trinocular [43], starting at two different months (January and April 2015) from four different sites [51].

Figure 4 shows the number of queries we see as we grow the size of the scan. Circles represent experimental trials measured at the final authority (slightly jittered). The diagonal line

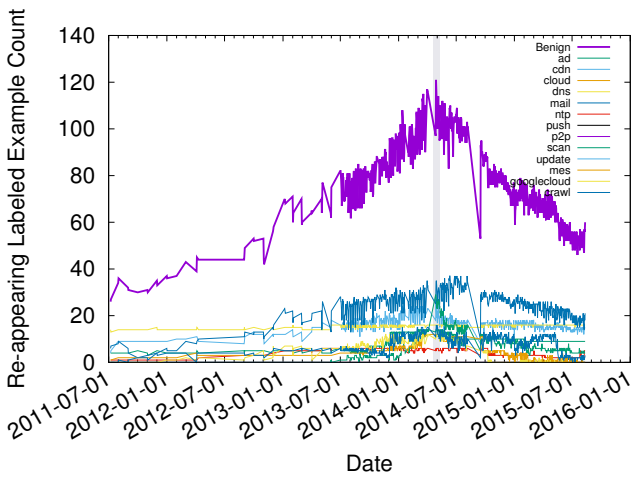

Fig. 5: Benign originator activity is relatively stable, before and after the labeled data on (28-30 April 2014, the gray bar). (Dataset: B-multi-year)

represents a best fit: roughly 1 querier per 1000 targets, but actually a power-law fit with power of 0.71. We also examine M-sampled and B-long, reporting what they see for 0.1% and 1% ZMap scans and the 100% Trinocular scans. This large reduction in responses at root servers is due to both DNS caching and because not all targets are actually interested in the scanner.

False negatives: Our controlled experiments can evaluate false negatives (missed network-wide events). The horizontal line at 20 queriers is our detection threshold, so we see that the final authority will detect all events scanning 0.001% of the Internet or more.

We expect greater caching at higher levels of the DNS hierarchy. We consider M-ditl-2015 data to show such attenuation. Only two trials overlap with this datasets: one for 0.01% and the other for 0.1%. Of these, we find two queriers for the 0.1% trial (the blue X) in Figure 4. Greater attenuation at higher levels of DNS means that it will detect only much larger (space) or longer (in time) activity. (We are currently extending this study to trials at larger percentages.)

This experiment shows that backscatter is highly attenuated due to disinterested targets and DNS caching, but responses follow the number of targets.

V. Long-term Accuracy of Algorithm

Our results are dependent on that how well training matches current conditions — as we described in § III-E, who carries out activities and how they act changes over time. These changes affect who is in the labeled examples, and what features they exhibit in observations of backscatter. To explore how these behaviors interact, we next study how rapidly sources come and go, and how each of the training strategies in § III-E affect accuracy: train once, retrain often with new feature values, and evolving the labeled dataset automatically and with expert guidance.

V-A Changing Activities Over Time

We know that network behavior changes over time, both naturally, as companies and applications come and go, and ar-

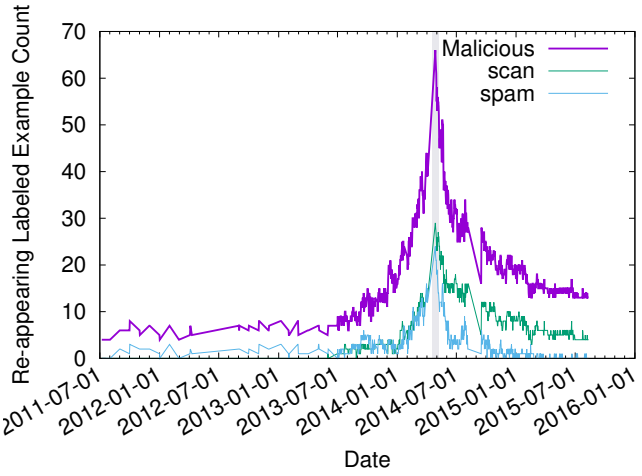

Fig. 6: Malicious originator activity changes quickly before and after the labeled data (28-30 April 2014,the gray bar). (Dataset: B-multi-year.)

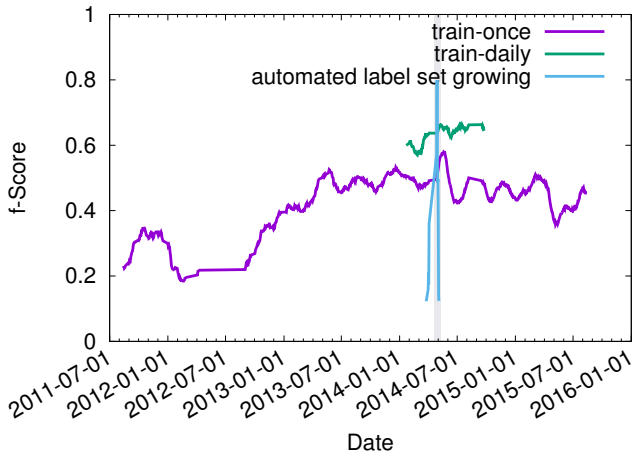

Fig. 7: Backscatter algorithm evaluation over time using different training methods. Train-daily strategy performs best among the three. The gray bar indicates curation days. (Dataset: B-multi-year)

tificially, as malicious activities are suppressed and re-appear. We next study how quickly activities change evolve over time, to understand how often we must retrain our classifier to maintain its accuracy.

To observe how quickly activity changes, we count daily active campaigns of each application class within curated labeled examples, which is a representative sample of network-wide activities within our data. Network-wide activity of a labeled example on a specific day indicates an active campaign of specific application class. In Figure 5 and Figure 6 we show count of active campaigns of curation day labeled examples (both benign and malicious) over time for B-multi-year data.

When we compare the number of labeled examples in the training period to times before and after, the largest number of active campaigns in the labeled dataset are between April 28th and 30th for both benign and malicious activity—a peak that corresponds with our curation period (the light gray vertical bar on Figure 5 and Figure 6). There is noise in all trends, showing day-to-day variation in how many of each category of campaigns are detected. But the dominant trend is that benign activity falls off slowly over time (e.g. 10% decay in 1 month and 20% decay in 6 months), both before and after curation. This drop occurs for all benign classes (Figure 5). Some services are very stable (for example, google and cloud), while others change more quickly (for example, cdn).

By contrast, malicious activity falls very sharply, to only 50% one moth before or after curation. We believe this very rapid turnover in malicious activity results from the adversarial nature of spam and scanning, with anti-spam companies seeking to suppress it and spam proponents changing which hosts they use and what and how they send to avoid detection.

V-B Trained-once Classifier Accuracy Degradation Over Time

We next consider how much classification accuracy degrades over time from shifting activity. To test classification accuracy with one-time training, we train using B-post-ditl data. We validate our classifier's performance on a day using re-appearing labeled examples by freshly calculating their feature vectors from that day's backscatter data. Figure 7 shows results of this experiment, comparing algorithm performance (using f-score metric from § IV-C that range from 0 to 1. All other metrics of algorithm performance follow the similar trends and not drawn.) against date (the x-axis).

This experiment shows that, with one-time training, our algorithm accuracy begins to drop off immediately after the curation days (2014-04-28 to -30). Even though there are a fair number of examples, the feature vectors those examples exhibit change quickly—we must retrain on new feature values to capture this shift.

In the next section we evaluate how retraining often on new feature values can improve accuracy, even if we do not change the set of labeled examples.

V-C Trained-often Classifier Improvements Over Time

We next show that we can improve accuracy over time if we retrain with new feature vectors, even if the labeled dataset is fixed. This retraining updates classification boundary with fresh information. Retraining against observations for time t is possible as long as there are enough examples in the labeled dataset that are active at time t, and § V-A showed that benign examples remain active for about 8 month while malicious activities for just one month.

Figure 7 shows classification performance with daily retraining. This graph shows that we sustain good classification performance (within 90% of our best performance) for about a month after curation for malicious activities and about 8 months for benign activities. After these periods there are insufficient activity in the labeled examples to train our classifier. Our training process fails due to lack of required examples before Feb 2014 and after Oct 2014.

While retraining gains one month of reasonable accuracy for malicious activity and eight months for benign activity, further improvements require more labeled examples. We next consider automatic and then manual labeling.

V-D Automatically Growing Labeled Examples

We saw that retraining often on the same labeled dataset but with fresh features, helps success, but it helps benign more than malicious, because the set of hosts doing malicious activities shift quickly. To address change, we next evaluate if we can automatically grow the labeled dataset, while in the next section (§ V-E) we consider expert guidance.

New labeled examples help by replacing lables that are no longer active. Fresh representatives of each application class improve training and result in better training and classification, counteracting drift (Figure 6).

In Figure 7, we show that simple strategy of utilizing classification output from a day as input to next day's training input fails to work. Accuracy drops 88% within a month after (or before) curation. In this experiment, about 30% of training input (after curation days) of the classifier is not correct on next day after curation (our classifier is about 70% accurate in classification when we used our curated labeled examples). Due to this classification error, we can not use our usual cross-validation method (§ IV-C) and rather use all the re-appearing labeled examples from B-post-ditl for validation on a specific day. As a consequence of this mechanism, training and validation labeled examples are the same on curation days, that results in a deceptively high accuracy on curation days. Though classification error quickly accumulates over days and proves such a mechanism ineffective for automatic generation of new examples because our ML algorithm fails to generate classification boundary before Mar 2014 and after May 2014. We conclude that automatic training is not viable with the level of classification error we see. Possible future work is to explore other automatic methods to verify candidate additions to labeled examples to improve their accuracy.

V-E Manually Growing Labeled Examples

Finally, we use M-sampled to show that growing labeled examples via recurring human expert based curation gives a high performing classifier. We conduct three curations on M-sampled on 2014-04, 2014-06 and 2014-09. As we reported earlier in Table III, we consistently get f-score of 0.79 for the duration of data. This is not surprising provided expert guidance is the gold standard for supervised algorithms. We rather see sensitivity of classification here to show that recurring curation provides a robust classifier.

We use M-sampled to evaluate the sensitivity of our conclusions by looking at when or if classifications change over time. We current vote on classifications of each originator over all weeks. To estimate degree of consensus in a vote, we define r as the fraction of weeks when the preferred class (the most common response) occurred of all weeks that originator appeared. When r is near one, we see consistent activity, but when $r < 0.5$ it seems likely that either the originator is changing activity over time (perhaps different machines behind a NAT, or a botnet node being repurposed), or doing two things concurrently, or we classify variations in behavior differently, suggesting an incomplete training set or indistinguishable classes.

Figure 8 shows the cumulative distribution of r, looking at subsets of data with at least q queriers per originator. To avoid overly quantized distributions we show only originators that appear in four or more samples (weeks). Requiring more queriers per originator (larger q) reduces the number of eligible originators, but the number of samples (shown in parenthesis in the legend) are all large enough to be robust.

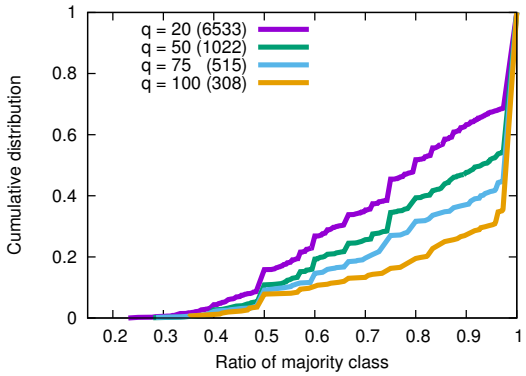

Fig. 8: CDF of r, the fraction of the most common class over all weeks with q or more queriers per originator. (Dataset: M-sampled.)

More queriers (thus more data for classification) provide more consistent results, since with $q = 100$, about 60% of originators are strongly consistent. Even with a threshold of 20 querier per originator, 30% of originators are consistent. In addition, almost all originators (85–90%, depending on q) have a class that has strict majority ($r > 0.5$). Thus our approach almost always (85–90%) provides a consistent result.

For the few where the strongest class is only a plurality ($r \leq 0.5$), we wanted to see if there are two nearly equally strong classes. We examined the entropy for originators in this case: we find that usually there is a single dominant class and multiple others, not two nearly equally common classes.

These initial results suggest our approach is consistent for observers with at least 20 queriers, although noise grows as queriers approach that threshold.

V-F Summary, Recommendations, and Future Work

We compare our training strategies in Figure 7. It is clear that, updating the classification boundary pays off in terms of better performance. Due to these results we recommend, human expert based curation from short-term data and adapting the classification boundary using fresh feature vector observations and re-training daily. Meanwhile labeled examples re-appearance count informs about next expert curation.

Study of long-term accuracy in the face of evolving behavior by originators is potential ongoing work. We also plan to compare our originator behavior change results to non-root entities (for example, .jp and final authority) in the future.

VI. Results

We next study network-wide activities with our method, identifying large network events and trends by applications and over time. Since our approach is based on feedback from targets, our results complement prior studies (such as darknets) and will observe targeted events will not appear in darknets.

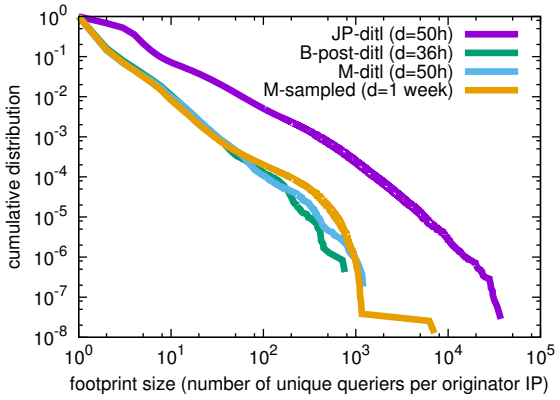

Fig. 9: Distribution of originator footprint size.

VI-A Sizes of Originator Footprints

We estimate the footprint of each originator as the number of unique queriers per originator. Figure 9 shows the fraction of originators with each footprint size a log-log scale for each of our three datasets.

Our data suggests *there are hundreds of originators that touch large parts of the Internet*. Our controlled trials (§ IV-D) show high attenuation at root servers, yet many originators show evidence of scanning most or all of the Internet (590 in M-ditl and 298 in B-post-ditl have footprints larger than 10^2).

The *distributions* of footprints are consistent across our datasets. (We cannot directly compare footprint sizes due to variation in duration and sampling.) As typical, they are a heavy-tailed, with some originators triggering queries from 10k queriers. The remainder of our analysis considers originators with the largest footprints: the top-10000 (about 0.5% of each dataset), or the top-1000 or -100. Considering only large originators will miss those that are intentionally trying to be stealthy, but many scanners make no such attempt [17], and we expect commercial large services to also be open.

The largest footprints here are larger than those we observe in controlled trials at M-Root (Figure 4). Those scans were quite short (a few to a dozen hours), while here we aggregate data over one or two days. In addition, our trials used random targets, most of which are unoccupied (only 6–8% respond, as seen before [26]); many real-world scans are targeted, resulting in higher responses rates and thus greater backscatter.

VI-B Observability and Size of Application Classes

We next classify the top originators. Our goal is to understand what activity is taking place and approximately how aggressive they are. Our key observations are: there are *thousands* of originators causing network-wide activity, different *authorities see different applications*, and *we see evidence of team of coordinated scanners* even with *no direct information from originators*.

Size of application classes: There are *thousands* of unique originators that touch large parts of the Internet. Table V shows how many originators we see in each originator class for each dataset, with classes with counts within 10% of the largest count in bold. We use our preferred classifier (RF) with per-dataset training over the entire ground-truth. Classes that lack ground truth for some dataset have no matches (a "-").

Applications vary by authority: The classes of applications seen at *different authorities vary considerably*. For JP-ditl, *spam* is the most common class of originator. Although Japan hosts computers for most major CDNs, the size of the *cdn* class seen from backscatter is small because CDNs often use address space assigned by other registrars (we verify this statement for Akamai and Google with geolocation, whois and prior work [21]).

The *update* class is exactly those in labeled ground-truth. We identified this class in examining the data (not from an external source), and lack of additional examples suggests either class has insufficient training data to avoid over-fitting, or update servers are rare.

Both unsampled root servers (B-post-ditl and M-ditl) show similar distributions of activity, with *mail* the most common and *spam* and *cdn* both close. M-Root shows many CDNs because due to 300 *cdn* originators located in two Chinese ISPs and interacting with queriers in China. Classification appears correct (they do not send traffic to darknets, nor appear in spam blacklists), but the originators lack domain names and we cannot identify them. Such originators appear only in M-ditl, suggesting that their queriers may be using DNS resolvers that prefer nearby authorities, since M-Root is well provisioned in Asia while B-Root is only based in North America.

Long-term, sampled root data (M-sampled) has some important differences from short term (M-ditl). Consider relative sizes of classes (since absolute counts vary due to dataset duration), we see many more scanner and spammers in long-term data. We believe the size of these categories reflect churn in the population carrying out the activity. We expect churn in spamming where computers known for spamming are less effective. We measure churn directly for scanners in § VI-C.

Big footprints can be unsavory: The mix of applications varies as we examine originators with smaller footprints, but we see that *big footprints are often unsavory activity*. Figure 10 shows how originator classes change as we look at more originators with smaller footprints (from Figure 10a to Figure 10c).

The largest footprints are often spammers (in JP-ditl) or scanners (for B and M). By contrast, we see that *mail* appears only in the top-1000 and top-10000, suggesting that legitimate mail servers may service large mailing lists (to many targets), but spammers touch many more targets. For B and M, more spammers rise in Figure 10c, suggesting spreading of traffic over many smaller originators to evade filtering.

By contrast, large but not top originators are often infrastructure: *cloud*, *mail*, *ad-tracker*, and *crawler*. In general, we find that application classes have a "natural" size, with some favoring origins with large footprints (prominent in Figure 10a), while others favor smaller footprints and so are more common in Figure 10c.

The *ad-tracker* class is most common in the prominent in top-most originators (a larger red *ad-tracker* fraction in Figure 10a compared to to Figure 10c). There are relatively a few originators (we see 5 companies as 22 unique originating

data	ad-track	cdn	cloud	crawl	dns	mail	ntp	p2p	push	scan	spam	update
JP-ditl	210	49	-	-	414	1412	237	2235	-	355	**5083**	6
B-post-ditl	72	1782	168	361	76	**3137**	8	-	318	1228	2849	-
M-ditl	76	**2692**	135	557	258	**2750**	67	-	119	983	2353	-
M-sampled	1329	17,708	2035	885	1202	14,752	-	-	3652	**47,201**	34,110	-

TABLE V: Number of originators in each class for all datasets. (Classifier: RF.)

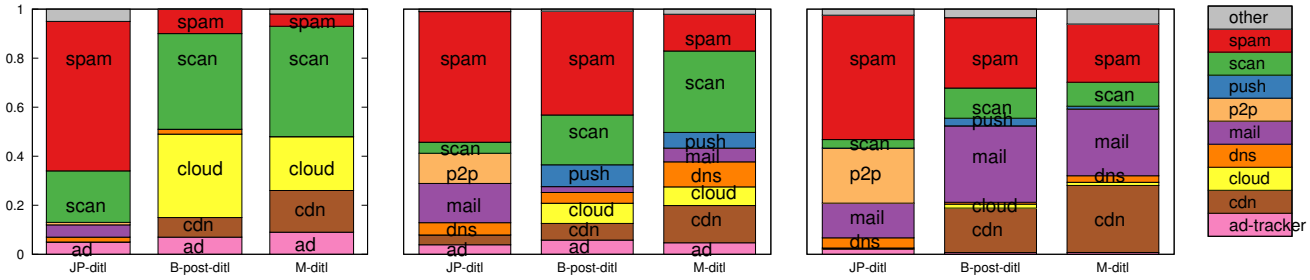

Fig. 10: Fraction of originator classes of top-N originators. (Dataset: JP-ditl, B-post-ditl, M-ditl; classifier: RF.)

addresses for top-100/JP-ditl). Unlike spam, they need not hide, and are likely prominent because tracking needs little traffic (a few originators can support a network-wide service), and because they use DNS records with short cache lifetimes (small TTLs). *Cloud* follows this pattern as well; 1 company across 21 distinct originating IPs for top-100 in M-ditl.

The *crawler* class shows the opposite behavior: most crawlers appear only in the top-10000, with few in top-1000 (554 vs. 3). This shift is consistent with web crawlers being data intensive, operating across many distributed IP addresses in parallel.

We also see that the physical location of the authority influences what they see. We earlier observed how differences in *cdn* for M-Root and B-Root are explained by their physical location to CDNs in China. B-Root's U.S.-only location may place it closer to more services in *cloud* (see Figure 10a) compared to M-Root's locations mainly in Asia and Europe.

New and old observations: A new observation in our data is *potential teams of scanners*. We manually identified several /24 address blocks where many addresses are engaged in scanning, suggesting possible parallelized scanning. Without direct scan traffic, we cannot confirm coordination, but backscatter suggests networks for closer examination. To understand with scope of potential collaborative teams, we start with a a very simple model where a team is multiple originators in the same /24 IP address block. In M-sampled we see 5606 unique scan originators (by IP address), across 2227 unique originating /24 address blocks. Of these, 167 blocks have 4 or more originators, suggesting a potential team of collaborators. While 128 of these blocks have multiple application classes, suggesting against collaboration (or possibly mis-classification), we see 39 blocks with 4 or more originators all with the same application class. Such blocks warrant closer examination.

We also confirmed prior observations that clients linger on retired services. Originators we find include four retired root DNS servers (B, D, J, L), two prior cloud-based mail servers,

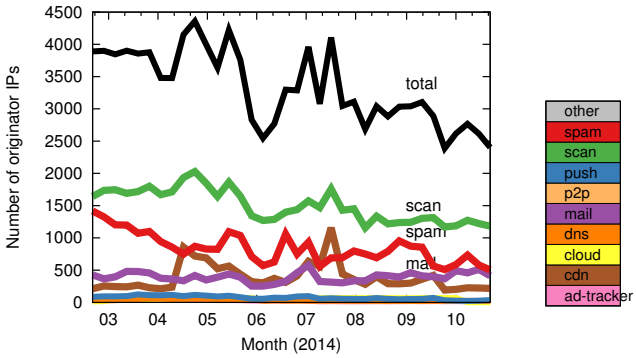

Fig. 11: Number of originators over time. (Dataset: M-sampled; classifier: RF.)

and one prior NTP server. These cases show our methods can be used to systematically identify overly-sticky, outdated clients across many services, automating prior reports of clients that stick to retired servers in DNS [30] and NTP [40].

Classification on originator actions: An important benefit of our approach is that we classify on *indirect actions caused by the originator, with no direct information* from the originator. In fact, about a quarter of the originators in JP-ditl and half of those in the root datasets have no reverse domain names, but originator omissions have no affect on our approach because we do not observe *any* traffic or reverse names of originators. This separation makes it more difficult for adversarial originators to conceal their activities.

VI-C Trends in Network-Wide Activity

We next look for long-term trends in our data. We believe this is the first longitudinal study of network-wide activities such as scanning (prior work focused on specific events [17]). Our goal is to understand the ebb and flow of network-wide activity, so rather than examine the N largest originators, we

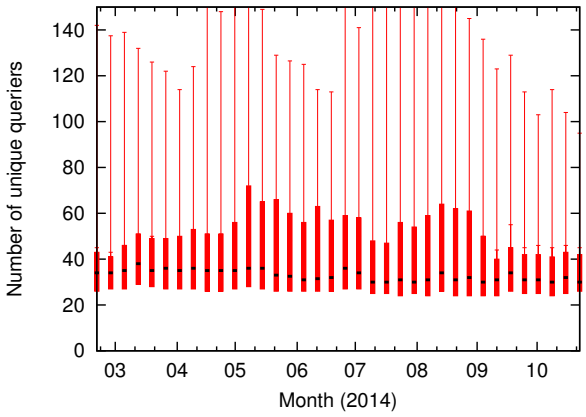

Fig. 12: Box plot of originator footprint (queriers per scanner) over time; whiskers: 10%ile/90%ile. (Dataset: M-sampled.)

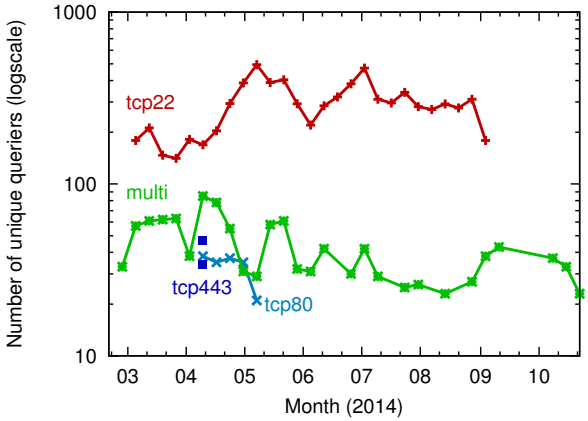

Fig. 13: Five example originators with application class *scan*. (Dataset: M-sampled with darknet.)

count all originators with footprints of at least 20 queriers (see also Figure 9). While we see no growth in network-wide events, we see *peaks that respond to security events* and *a core of slow-and-steady scanners*.

Peaks in numbers of originators: The top *all* line in Figure 11 shows the absolute number of originators over time, each class (the lower, colored lines) and total (the top, black line). There are fairly large week-by-week changes, showing churn in the number of active originator activities, and peaks that can be explained by reactions to network security events.

To understand how network activity results from real-world events we next look the *scanner* application class. Our observation period includes public announcement of the Heartbleed vulnerability on 2014-04-07 [39], and we know that there were multiple research [1], [18], commercial, and presumably government scanning activities triggered by that announcement. The green scanner line in Figure 11 shows more than a 25% increase in scanning by mid-April, from 1400 originator IPs per week to 1800 at its peak. While this change is noticeable, it is smaller than we might expect. Instead, it shows that reaction to Heartbleed is small compared to *the large amount of scanning that happens at all times*—the 1200–1400 scanners we saw in March, and the 1000–1200 scanners that are present from June to October.

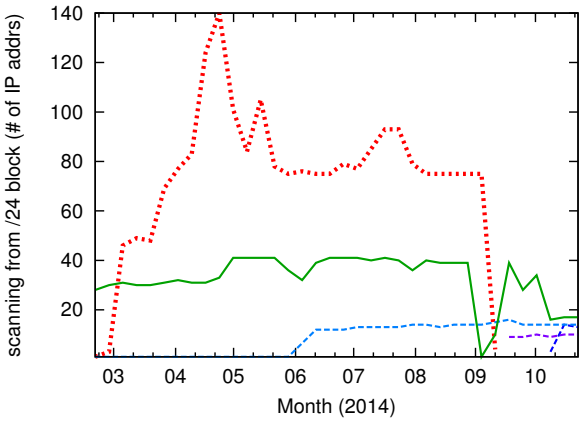

Fig. 14: Five example blocks originating scanning activity. (Dataset: M-sampled.)

Churn: To understand long-term scanning, Figure 12 shows the distribution of footprint sizes over time for class *scan*. While the median and quartiles are both stable over these 36 weeks, but the 90th percentile varies considerably. This variation suggests a few very large scanners that come and go, while a core of slower scanners are always present.

We illustrate this observation with five different scanners that appear in both M-sampled and our darknet data (Figure 13). Two are long-lived (the top "tcp22" line scanning ssh, and the middle line scanning multiple ports), while the tcp80 scanner occurs in April and May. Furthermore, two tcp443 scans only appear in one week in April (shown as dark squares), suggesting they are Heartbleed-related. We also see that tcp22 has a bigger footprint than the others, as part of a campaign with 140 other IP addresses in the same /24 block. (We show scanning for one of these addresses.) Using our darknets, we confirm 164 scanners for TCP ports 22, 80, or 443, and while there is no "typical" scanner, these variations are common.

Our approach also *identifies networks supporting scanners*. For each /24 block, we count the number of IP addresses in class *scan* over time; Figure 14 shows five of these blocks. The top dotted line is a block with significant scanning concurrent with Heartbleed and Shellshock, ending in September. The solid line shows a block that scans continuously, while the three dotted lines are scanners that appear during observation.

To understand if *who* scans changes over time, Figure 15 measures week-by-week change in scanner IP addresses. The bar above the origin shows the number of scanners each week, showing both new originators (top, dark) and continuing originators (middle, light) The red bar below the origin shows scanners that were lost from the prior week. While there are always scanners coming and going (about 20% turnover per week), this data confirms that there is a stable core of scanners that are consistently probing, week-after-week.

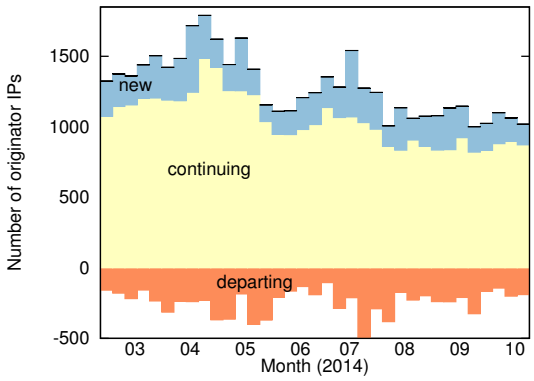

Fig. 15: Week-by-week churn for originators of class *scan*. (Dataset: M-sampled.)

VII. Related work

We next review prior work in both DNS- and non-DNS-based sensors and analysis. Overall, our contribution is to show that reverse DNS queries can identify network-wide behavior. Prior work instead considers forward DNS traffic and typically applies it to specific problems, or uses non-DNS sensors such as darknets and search engines.

DNS-specific sensors: Several groups use forward DNS queries to identify spam [58], [28], fast-flux [58], [27], automatically generated domain names [57], and cache poisoning [2]. Like our work, several of these approaches use machine learning to classify activity. However, this prior work focuses on *forward* DNS queries, while we consider *reverse* queries. Moreover, many use algorithms optimized to detect specific malicious activities, while we detect a range of network-wide behavior.

Recent work has used DNS to infer the structure of CDN networks [4] or internal to DNS resolvers [46]. They infer specific services from DNS traffic, we search for network-wide events from reverse queries.

An earlier work uses targeted scan and DNS backscatter for detecting Tor exit routers peeking POP3 authentication information [34], an earlier use of DNS backscatter de-anonymization; we generalize this use to detect scanners.

Plonka and Barford use custom clustering to identify undesirable activity from local DNS traffic [41]. They use DNS traffic from an organization's recursive resolver to infer activity about that organization. Overall, our approach provides larger coverage, both by using data from authoritative DNS servers that aggregate queries from many organizations, unlike their single organization, and by examining trends in ten months of data, unlike their week-long analysis.

Antispam software has long used reverse DNS lookups to directly classify sources of mail. We use the domain names of queriers to indirectly classify originators.

Non-DNS passive sensors: Darknets (or network telescopes) are a commonly used passive technique to characterize large-scale network activity [38], [35], [56], [13], [14], [17]. By monitoring a large, unoccupied blocks of addresses, darknets see active probes from viruses and scanners, queries from misconfiguration, and backscatter from spoofed traffic; traffic that can predict global malware, and its absence, network outages. Our analysis of DNS backscatter shares the goal of understanding network-wide activity from a simple, passive observer, but we observe at DNS authorities rather than large ranges of addresses. Like Durumeric et al. [17], we seek to enumerate scanners, but our use of DNS backscatter will see targeted scans that miss their darknet, and our study considers eight months of activity, not just one.

Some security services use middleboxes with deep-packet inspection to passively monitor large ISPs [3]. They observe all traffic from multiple points, while we monitor DNS backscatter from a single provider only.

Staniford monitored network traffic for scanners [49], and Gates emphasized rapid detection with scanner modeling [24]. Rather than protecting a single network, we look for network-wide activity with a simple observer.

Honeypots (for example, [42]) are a form of application-level darknet. By interacting with originators they see attacks darknets miss, but they miss attacks that probe specific targets (such as Alexa top sites). Interactivity also makes them fewer because of deployment expense. DNS backscatter uses information from existing servers.

Unconstrained endpoint profiling [50] uses search engines to understand addresses that leak into the public web, possibly complementing network flow data. We both seek to understand network-wide activity, but we use different data sources and methods. They use largely unstructured information from the web, while we infer features from semi-structured domain names and also traffic patterns. Their work depends on the speed of search engine indexing, while our work can provide rapid feedback given data from a DNS authority.

General DNS traffic analysis and privacy: Finally, a wide body of work has explored DNS traffic in general (examples include [15], [54], [11], [23]). Their work seeks to understand DNS, while we instead study what reverse DNS tells us about network-wide activities.

Work in DNS privacy focuses on client-to-recursive resolvers for end-users (examples include [37], [59], and proposals in the IETF DPRIVE working group). Our use of reverse queries from automated systems triggered by originators should see almost no human-triggered, end-user queries (§ II). Use of query minimization [5] at the queriers will constrain the signal to only the local authority (that immediately serving the originator's reverse address).

VIII. Conclusion

We identified DNS backscatter as a new source of information about benign and malicious network-wide activity, including originators of mailings list traffic, CDN infrastructure, spammers and scanners. Their activity triggers reverse DNS queries by or near their targets, and we show that classification of these queriers allows us to identify classes of activity with reasonable precision. We use our approach to identify trends in scanning across nine months of data from one data source, and we characterize several kinds of activity for two days over three data sources. Our work provides a new approach to evaluate classes of network-wide activity.

REFERENCES

[1] Mustafa Al-Bassam. Top Alexa 10,000 Heartbleed scan. https://github.com/musalbas/heartbleed-masstest/blob/94cd9b6426311f0d20539e696496ed3d7bdd2a94/top1000.txt, April 14 2014.

[2] Manos Antonakakis, David Dagon, Xiapu Luo, Roberto Perdisci, and Wenke Lee. A centralized monitoring infrastructure for improving DNS security. In *Proc. of the 13th International Symposium on Recent Advances in Intrusion Detection*, pages 18–37, Ottawa, Ontario, Canada, September 2010. Springer.

[3] Arbor Networks. Worldwide infrastructure security report. Technical Report Volume IX, Arbor Networks, January 2014.

[4] Ignacio Bermudez, Marco Mellia, Maurizio M. Munafo, Ram Keralapura, and Antonio Nucci. DNS to the rescue: Discerning content and services in a tangled web. In *Proc. of the ACM Internet Measurement Conference*, pages 413–426, Boston, MA, November 2012.

[5] S. Bortzmeyer. DNS query name minimisation to improve privacy. Work in progress (Internet draft draft-bortzmeyer-dns-qname-minimisation-02), May 2014.

[6] Carna Botnet. Internet census 2012: Port scanning /0 using insecure embedded devices. web page http://census2012.sourceforge.net/paper.html, March 2013.

[7] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, October 2001.

[8] Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone. *Classification and Regression Trees*. Chapman and Hall, 1984.

[9] Nevil Brownlee. One-way traffic monitoring with iatmon. In *Proc. of the Passive and Active Measurement Workshop*, pages 179–188, Vienna, Austria, March 2012.

[10] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. Mapping the expansion of Google's serving infrastructure. In *Proc. of the ACM Internet Measurement Conference*, pages 313–326, Barcelona, Spain, October 2013. ACM.

[11] Sebastian Castro, Duane Wessles, Marina Fomenkov, and kc Claffy. A day at the root of the Internet. *ACM SIGCOMM Computer Communication Review*, 38(5):41–46, October 2008.

[12] Jakub Czyz, Michael Kallitsis, Manaf Gharaibeh, Christos Papadopoulos, Michael Bailey, and Manish Karir. Taming the 800 pound gorilla: The rise and decline of NTP DDoS attacks. In *Proc. of the ACM Internet Measurement Conference*, pages 435–448, Vancouver, BC, Canada, November 2014. ACM.

[13] Jakub Czyz, Kyle Lady, Sam G. Miller, Michael Bailey, Michael Kallitsis, and Manish Karir. Understanding IPv6 Internet background radiation. In *Proc. of the ACM Internet Measurement Conference*, pages 105–118, Barcelona, Spain, 2013.

[14] Alberto Dainotti, Claudio Squarcell, Emile Aben, Kimberly C. Claffy, Marco Chiesa, Michele Russo, and Antonio Pescape. Analysis of country-wide internet outages caused by censorship. In *Proc. of the ACM Internet Measurement Conference*, pages 1–18, Berlin, Germany, November 2011.

[15] Peter B. Danzig, Katia Obraczka, and Anant Kumar. An analysis of wide-area name server traffic: A study of the Domain Name System. In *Proc. of the ACM SIGCOMM Conference*, pages 281–292, January 1992.

[16] DNS-OARC. Day in the life of the internet (DITL) 2014. https://www.dns-oarc.net/oarc/data/ditl, April 2014.

[17] Zakir Durumeric, Michael Bailey, and J. Alex Halderman. An Internet-wide view of Internet-wide scanning. In *Proc. of the 23rd USENIX Security Symposium*, pages 65–78, San Diego, CA, August 2014. USENIX.

[18] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, and J. Alex Halderman. The matter of Heartbleed. In *Proc. of the ACM Internet Measurement Conference*, pages 475–488, Vancouver, BC, Canada, November 2014. ACM.

[19] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide scanning and its security applications. In *Proc. of the USENIX Security Symposium*, pages 605–620, Washington, DC, USA, August 2013. USENIX.

[20] Robert Edmonds. ISC passive DNS architecture. Technical report, Internet Systems Consortium, Inc., March 2012.

[21] Xun Fan, Ethan Katz-Bassett, and John Heidemann. Assessing affinity between users and CDN sites. In *Proc. of the 7th IEEE International Workshop on Traffic Monitoring and Analysis (TMA)*, pages 95–110, Barcelona, Spain, April 2015. Springer.

[22] Kensuke Fukuda and John Heidemann. Detecting malicious activity with DNS backscatter. In *Proceedings of the ACM Internet Measurement Conference*, pages 197–210, Tokyo, Japan, October 2015. ACM.

[23] Hongyu Gao, Vinod Yegneswaran, Yan Chen, Phillip Porras, Shalini Ghosh, and Jian Jiang Haixing Duan. An empirical reexamination of global DNS behavior. In *Proc. of the ACM SIGCOMM Conference*, pages 267–278, Hong Kong, China, 2013.

[24] Carrie Gates. Coordinated scan detection. In *Proc. of the ISOC Network and Distributed System Security Symposium*, San Diego, CA, February 2009. The Internet Society.

[25] Kenneth Geers, Darien Kindlund, Ned Moran, and Rob Rachwald. World War C: Understanding nation-state motives behind today's advanced cyber attacks. Technical report, FireEye, September 2014.

[26] John Heidemann, Yuri Pradkin, Ramesh Govindan, Christos Papadopoulos, Genevieve Bartlett, and Joseph Bannister. Census and survey of the visible Internet. In *Proc. of the ACM Internet Measurement Conference*, pages 169–182, Vouliagmeni, Greece, October 2008. ACM.

[27] Thorsten Holz, Christian Gorecki, Konrad Rieck, and Felix Freiling. Measuring and detecting fast-flux service networks. In *Proc. of the ISOC Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2008. The Internet Society.

[28] Keisuke Ishibashi, Tsuyoshi Toyono, Katsuyasu Toyama, Masahiro Ishino, Haruhiko Ohshima, and Ichiro Mizukoshi. Detecting mass-mailing worm infected hosts by mining DNS traffic data. In *Proc. of the ACM SIGCOMM MineNet Workshop*, pages 159–164, Philadelphia, PA, August 2005.

[29] Julian Kirsch, Christian Grothoff, Monika Ermert, Jacob Appelbaum, Laura Poitras, and Henrik Moltke. NSA/GCHQ: The HACIENDA program for internet colonization. *C'T Magazine*, Aug.15 2014.

[30] Matthew Lentz, Dave Levin, Jason Castonguay, Neil Spring, and Bobby Bhattacharjee. D-mystifying the D-root address change. In *Proc. of the ACM Internet Measurement Conference*, pages 57–62, Barcelona, Spain, October 2013. ACM.

[31] Kirill Levchenko, Andreas Pitsillidis, Neha Chachra, Brandon Enright, Márk Félegyházi, Chris Grier, Tristan Halvorson, Chris Kanich, Christian Kreibich, He Liu, Damon McCoy, Nicholas Weaver, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. Click trajectories: End-to-end analysis of the spam value chain. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 431–446, Oakland, CA, USA, May 2011. IEEE.

[32] Zhichun Li, Anup Goyal, Yan Chen, and Aleksandar Kuzmanovic. Measurement and diagnosis of address misconfigured P2P traffic. In *INFOCOM'10*, pages 1–9, San Diego, CA, March 2010.

[33] MaxMind LLC. GeoIP. http://www.maxmind.com/geoip.

[34] Damon McCoy, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Shining light in dark places: Understanding the tor network. In *Proc. of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, pages 63–76, Leuven, Belgium, July 2008.

[35] Jon Oberheide, Manish Karir, Z. Morley Mao, and Farnam Jahanian. Characterizing dark DNS behavior. In *Proc. of the 4th International Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, pages 140–156, Lucerne, Switzerland, July 2007. Springer.

[36] Robert O'Harrow, Jr. Cyber search engine Shodan exposes industrial control systems to new risks. *The Washington Post*, June 3 2012.

[37] OpenDNS. DNSCrypt: Introducing DNSCrypt. web page http://www.opendns.com/about/innovations/dnscrypt/, January 2014.

[38] Ruoming Pang, Vinod Yegneswaran, Paul Barford, Vern Paxson, and Larry Peterson. Characteristics of Internet background radiation. In *Proc. of the ACM Internet Measurement Conference*, pages 27–40, Sicily, Italy, 2004.

[39] Nicole Perlroth. Thought safe, websites find the door ajar. *New York Times*, page A1, Apr. 9 2014.

[40] Dave Plonka. Flawed routers flood university of wisconsin internet time server. http://pages.cs.wisc.edu/~plonka/netgear-sntp, 2003.

[41] David Plonka and Paul Barford. Context-aware clustering of DNS query traffic. In *Proc. of the ACM Internet Measurement Conference*, pages 217–229, Vouliagmeni, Greece, October 2008.

[42] Niels Provos. A virutal honeypot framework. In *Usenix Security Symposium 2004*, pages 1–14, San Diego, CA, August 2004.

[43] Lin Quan, John Heidemann, and Yuri Pradkin. Trinocular: understanding Internet reliability through adaptive probing. In *Proc. of the ACM SIGCOMM Conference*, pages 255–266, Hong Kong, China, August 2013.

[44] Lin Quan, John Heidemann, and Yuri Pradkin. When the Internet sleeps: Correlating diurnal networks with external factors. In *Proc. of the*

ACM Internet Measurement Conference, page to appear, Vancouver, BC, Canada, November 2014. ACM.

[45] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[46] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. On measuring the client-side DNS infrastructure. In *Proc. of the ACM Internet Measurement Conference*, pages 77–90, Barcelona, Spain, October 2013.

[47] Farsight Security. SIE (Security Information Exchange). https://www.farsightsecurity.com/Services/SIE/, 2013.

[48] Shadow server foundation. http://www.shadowserver.org/.

[49] Stuart Staniford, James A. Hoagland, and Joseph M. McAlerney. Practical automated detection of stealthy portscans. *Journal of Comptuer Security*, 10(1):105–136, 2002.

[50] Ionut Trestian, Supranamaya Ranjan, Aleksandar Kuzmanovi, and Antonio Nucci. Unconstrained endpoint profiling (Googling the Internet). In *Proc. of the ACM SIGCOMM Conference*, pages 279–290, Seattle, WA, Aug 2008.

[51] USC/LANDER project. Internet address census, datasets internet_address_census it63w, it63c, it63j, it63g, it64w, it64c, it64j, it64g. web page http://www.isi.edu/ant/lander, January (it63) and April (it64) 2015.

[52] Paul Vixie. Passive DNS collection and analysis, the 'dnstap' approach. Keynote talk at FloCon, January 2014.

[53] Florian Weimer. Passive DNS replication. In *Proc. of the 17th Forum of Incident Response and Security Teams (FIRST)*, Singapore, April 2005.

[54] Duane Wessels and Marina Fomenkov. Wow, that's a lot of packets. In *Proc. of the Passive and Active Measurement Workshop*, La Jolla, CA, April 2003.

[55] Wikipedia. Gini coefficient. http://en.wikipedia.org/wiki/Gini_coefficient, 2015.

[56] Eric Wustrow, Manish Karir, Michael Bailey, Farnam Jahanian, and Geoff Houston. Internet background radiation revisited. In *Proc. of the 10th ACM Internet Measurement Conference*, pages 62–73, Melbourne, Australia, November 2010. ACM.

[57] Sandeep Yadav, Ashwath Kumar, Krishna Reddy, A.L. Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated malicious domain names. In *Proc. of the ACM Internet Measurement Conference*, pages 48–61, Melbourne, Australia, November 2010.

[58] Bojan Zdrnja, Nevil Brownlee, and Duane Wessels. Passive monitoring of DNS anomalies. In *Proc. of the 4th International Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, pages 129–139, Lucerne, Switzerland, 2007.

[59] Liang Zhu, Zi Hu, John Heidemann, Duane Wessels, Allison Mankin, and Nikita Somaiya. Connection-oriented DNS to improve privacy and security. In *Proc. of the 36th IEEE Symposium on Security and Privacy*, pages 171–186, San Jose, Californa, USA, May 2015. IEEE.

APPENDIX A
DETAILS ABOUT CONFIRMING LABELED GROUND-TRUTH

We generate moderate to large lists of potential IP addresses in each application class from external sources (described below), then intersect it with the top-10000 originators in dataset by the number of queriers. We then verify the intersection manually. We determine the verifying application class of each originator manually using the methods described below. We added two categories (push and update) after examining all of the top 100 and 1000 largest originators, respectively.

Validation for each class is as follows:

ad-tracker We identify servers associated with advertising affiliation and tracking systems (such as doubleclick.net and kauli.net) by crawling around 20 blog pages. We also registered with ad affiliation programs for four services and added the servers they recommended.

cdn We identified content-distribution networks for several large providers (Akamai, Edgecast, CDNetworks, ChinaCache, and, CloudFlare) based on their reverse domain names and WHOIS on their IP addresses.

cloud We crawled IP addresses of front-end hosts for Google services (such as map, drive, and news) and Dropbox from Japan and the US.

crawler We confirm web crawler IP addresses from UserAgent strings in web server logs of NII, originator reverse domain names, and http://botvsbrowsers.com.

dns We started with all root DNS servers, country-code TLD servers, then we added servers for large Japanese ISPs that had large originator footprints.

mail We confirm mail originators by examining over 100 Japanese mailing lists operated by mail ASPs and companies themselves. We see about 300 IP addresses over about three months in 2014. We also identify mail originators for cloud-based mail including Google, Microsoft, Apple, Facebook, Linkedin, and Amazon, as confirmed by domain names in e-mail message headers.

ntp We consider well-known Stratum 1 servers and crawled IP addresses of NTP servers in Open NTP project (at *.pool.ntp.org).

push We list IP addresses that send packets with TCP port 5223 (Lync mobile client push), as observed in sampled netflow traffic from the University of Tokyo.

p2p We identify IP addresses running DHT-based BitTorrent from htindex.com in fall 2014 (As of 2015, this service no longer operates).

scan We confirm scanners by looking at traffic in two darknets (one a /17 and the other a /18 prefix) located in Japan. A confirmed scanner sends TCP (SYN only, not SYN-ACK), UDP, or ICMP packets to more than 1024 addresses in at least one darknet, or is a known research scanner (such as Trinocular [43] and shadowserver [48]).

spam We confirm spammer IP addresses with data from DNS blacklists (DNSBL) run by 9 organization (badips, barracuda, dnsbl.sorbs, inps.de, junkemail, openbl, spamhaus, spamrats, spam.dnsbl.sorbs); we consider only the spam portion of blacklists that distinguish spam from others such as ssh-brute force.

update We identified 5 large software update services hosted in Japan corresponding to computer and printer vendors such as Sony, Ricoh, and Epson.

We identified these classes after iteration and manual examination of data, traces, and applications. They share in common that a single originator touches many targets that make reverse DNS queries, but activity is sometimes initiated by the originator (spam, mail, and scan) and other times likely a side effect of activity at the target (such as update, ad-tracker, and cdn).

APPENDIX B
UNDERSTANDING ORIGINATORS WITH LARGE FOOTPRINTS

We next look at several originators with large footprints (many queriers) to characterize the kinds of activity we find.

B-A Important originators in a country-code authority

First we validate originators that rank highly by number of queriers and compare them against external sources. Table VII lists the top 30 high rank originators in JP-ditl, sorted by the number of unique querier IPs per originator.

We correlate these originators found by our approach against several external sources of information. The DarkIP column shows the number of darknet IP addresses receiving more than one packet whose source IP address is matched to originator IP. This column shows external evidence that the originator is a network scanner. Column BLS (Blacklist Spam) and BLO (Blacklist Other) show DNS-based blacklist information: many blacklists include this originator as either malicious or other. Maliciousness here includes activities such as scanning, ssh attack, phishing, spammer and so on. Finally, the class shows how we classify that originator using random-forest classification.

From this table, we understand that four out of 30 originators are "clean", not showing up on external sources. Many of the rest are scanners (showing up on the darknet column) or spammers (appearing in blacklists). Ranks 5-9 send TCP port 80 probes and are all located in the same /24 subnet in a commercial cloud provider, although most of them are judged as clean in existing black lists. Rank 2 is the ad network host and Rank 3 is a known scanner labeled as scan-ssh in the case study (a host running Trinocular, operated by USC/ISI). It is omitted there because Trinocular is adaptive, avoiding probing non-responsive darknet space. In addition, rank 12, located in a small university, looks benign but we found a web page about a large scale mail account hacking incident occurred in a mail server during the measurement period. Finally, our manual investigation revealed that originator ranked 14 is used for software update in a Japanese computer vendor, and end-hosts query the FQDN of the originator IP.

B-B Important Originators from a Root Authority

Table VIII shows similar top originators, but based on what appears in the M-ditl dataset taken from the M-root DNS server. At first, the many CDNs appear as top originators. CDNs are more common here because they use names with

dataset	ad-track	cdn	cloud	crawler	dns	mail	ntp	p2p	push	scan	spam	update	total
JP-ditl	15	8	-	-	26	44	10	37	-	25	64	6	**235**
B-post-ditl	13	29	16	17	16	46	5	-	12	29	35	-	**214**
M-ditl	13	36	16	16	17	50	8	-	12	33	43	-	**240**
M-sampled	54	81	82	35	52	111	-	-	73	124	136	-	**746**

TABLE VI: Number of examples of each application class in labeled ground-truth, per dataset.

rank	originator	No.queriers	TTL	DarkIP	BLS	BLO	class	note
1	209.11.251.22*	37136	1h	0	3	3	spam	home
2	199.199.253.166*	34082	†15m	0	0	0	ad	ad
3	198.245.9.213*	31481	1d	0	0	0	scan	scan-icmp
4	217.244.159.155*	30468	8h	0	1	0	spam	home
5	217.235.158.247*	30240	1h	1002	0	0	scan	tcp80
6	217.235.158.176*	29036	1h	963	0	0	scan	tcp80
7	217.235.158.71*	28954	1h	956	1	0	scan	tcp80
8	217.235.158.203*	28598	1h	881	0	0	scan	tcp80
9	217.235.158.140*	28236	1h	543	1	0	scan	tcp80
10	217.235.158.169*	27992	1h	1001	0	0	scan	tcp80
11	199.106.226.197*	25086	8h	0	3	2	spam	home
12	199.244.128.240*	23862	†10m	0	0	0	spam	nxdom (incident)
13	217.231.31.107*	23750	1d	0	4	3	spam	home
14	199.196.81.153*	23267	F	0	0	0	update	sony
15	59.70.195.55*	23232	1d	0	3	3	spam	home
16	217.165.32.172*	22526	1d	0	2	0	spam	ns
17	216.104.82.227*	22287	1d	0	3	1	spam	home
18	209.10.165.232*	20949	1d	0	2	2	spam	home
19	59.239.61.50*	20672	1d	0	2	1	spam	other
20	217.247.217.113*	20482	10m	0	3	2	spam	other
21	208.230.234.88*	20267	1d	0	1	1	spam	other
22	209.0.5.148*	20006	12h	0	3	0	spam	mail
23	59.65.97.12*	19945	1d	0	3	1	spam	home
24	217.240.44.7*	19886	1d	0	2	0	spam	home
25	214.204.213.92*	19754	10m	0	2	1	spam	home
26	217.252.134.219*	18869	8h	0	1	1	spam	home
27	199.106.209.97*	18467	8h	0	2	1	spam	home
28	199.204.159.229*	18145	†1d	0	4	3	spam	nxdom
29	199.105.172.122*	17666	†20m	0	1	0	spam	nxdom
30	217.231.17.155*	16707	500	0	1	0	spam	other

TABLE VII: Frequently appeared originators in JP-ditl. IP addresses with astrisks (*) use prefix-preserving anonymization (non-astrisks are not anonymized). A dagger (†) in TTL field indicates negative cache value and F means failure (servfail or not reached).

short TTLs (to support DNS-based CDN selection), and there are few CDN hosts covered by address space for which JPRS is an authority that would appear in JP-ditl. This difference highlights the global scope of a root authority compared to the regional focus on JP-dtil.

We also find some originators that are maintained by security vendors. We verified these manually and these indicate many clients checking back these services for updates. Originators relating to spamming and mailing lists do not appear in this M-dtil list. One possible reason for their omission is that the spatial distribution of spammers and mailing list is limited in country level, and less opportunity to identify as top talkers.

Scanner related originators are more common on the list. Most of them are scanning port 22 (ssh) over wide range of networks. An interesting point is that some of them are not reported in our darknet data, likely because it is relatively small. However, DNS backscatter confirms similar kind of activities as scanners.

In addition, Google IP addresses appear in the list. The first is the Google Public DNS service and the other appears to be related to cloud hosting and Google drive (it is not related to mail). Google's IP addresses in the list are categorized into three patterns: (1) public DNS, (2) cloud service, (3) mail service, and (4) web crawling. IP addresses showing Google web crawling are not appeared on the whole top 10000 list. This suggests that Google's crawling is highly distributed, so no individual crawling IP address is prominent. We do find web crawlers operated by other companies in the M-ditl top 10000 list, although the total number of IPs for such crawlers is relatively small.

In summary, our analysis of JP-ditl and M-ditl demonstrates that most top originators that appear on from DNS backscatter correspond to understable, large-scale activities in the Internet, and often malicious activity. Also, we emphasize that our classification algorithm labels reasonable originator class to originators.

APPENDIX C
DYNAMIC FEATURES OF CASE STUDIES

This section provides some additional data showing how dynamic features vary across different classes of originator in our labeled, ground-truth data.

When we look at numbers of queriers over the day (Figure 16), diurnal trends are strongly visible in several of cases: scan-icmp, ad-net, cdn, and mail. These activities are influenced by user behavior. CDN and ad-net track human activity. Mail-list is a run by a Japanese newspaper, showing strong, very regular mass-mailing early in the day and during local business hours.

rank	originator	No.queriers	TTL	DarkIP	BLS	BLO	class	note
1	94.27.210.131*	1201	†1m	0	0	0	cdn	barracuda
2	100.86.180.136*	1172	†10m	0	0	1	cdn	edgecast
3	119.106.67.236*	1096	†2d	49K	0	1	scan	tcp22 nxdom (CN)
4	34.186.138.129*	1046	F	0	0	3	scan	unreach (CN)
5	12.174.108.117*	986	†5m	0	0	0	cdn	cdnetworks
6	210.152.126.146*	918	F	0	0	0	cdn	akamai
7	8.8.8.8	866	1d	0	0	1	dns	google
8	210.152.126.143*	838	F	0	0	0	cdn	akamai
9	210.152.126.142*	824	F	0	0	0	cdn	akamai
10	175.45.161.87*	816	†1h	0	1	1	cdn	gmocloud
11	90.21.41.201*	814	†43m	0	0	0	ad	integral ad
12	219.171.37.153*	763	†2h	0	0	0	cdn	akamai
13	219.171.37.154*	741	†2h	0	0	0	cdn	akamai
14	175.45.239.19*	719	5m	0	0	0	cdn	myvps
15	90.21.41.175*	696	†43m	0	0	0	ad	integral ad
16	109.179.191.58*	689	F	0	1	0	scan	netguard
17	201.225.208.68*	643	F	0	0	0	scan	samsung-de
18	145.1.114.160*	621	†1h	0	0	0	cdn	cdnetworks
19	114.108.104.142*	587	F	0	1	2	scan	unreach (CN)
20	73.105.58.230*	586	†1d	0	0	0	cdn	other
21	100.214.209.120*	578	1d	0	2	2	scan	home (US)
22	201.138.164.158*	558	F	34	0	3	scan	unreach (CR)
23	117.248.235.158*	554	1d	0	2	2	spam	home (TW)
24	59.246.74.97*	516	†5m	0	0	0	cdn	cdnetworks
25	114.108.104.197*	501	F	0	0	2	scan	unreach (CN)
26	114.108.104.201*	490	F	0	0	2	scan	unreach (CN)
27	129.187.116.145*	483	†11h	0	2	2	spam	nxdom (ID)
28	114.108.104.161*	469	F	0	0	2	scan	unreach (CN)
29	221.128.188.73*	466	F	0	0	0	scan	skype
30	54.84.6.244*	463	5m	0	0	0	cdn	amazonaws
47	86.96.136.57*	405	†3h	0	0	0	cloud	google
53	215.152.209.167*	395	F	49K	0	2	scan	tcp22 nxdom (PK)
74	119.106.67.235*	363	†2d	49K	0	1	scan	tcp21 unreach (CN)

TABLE VIII: Frequently appeared originators in M-ditl. IP addresses with astrisks (*) use prefix-preserving anonymization (non-astrisks are not anonymized). A dagger (†) in TTL field indicates negative cache value and F means failure (servfail or not reached).

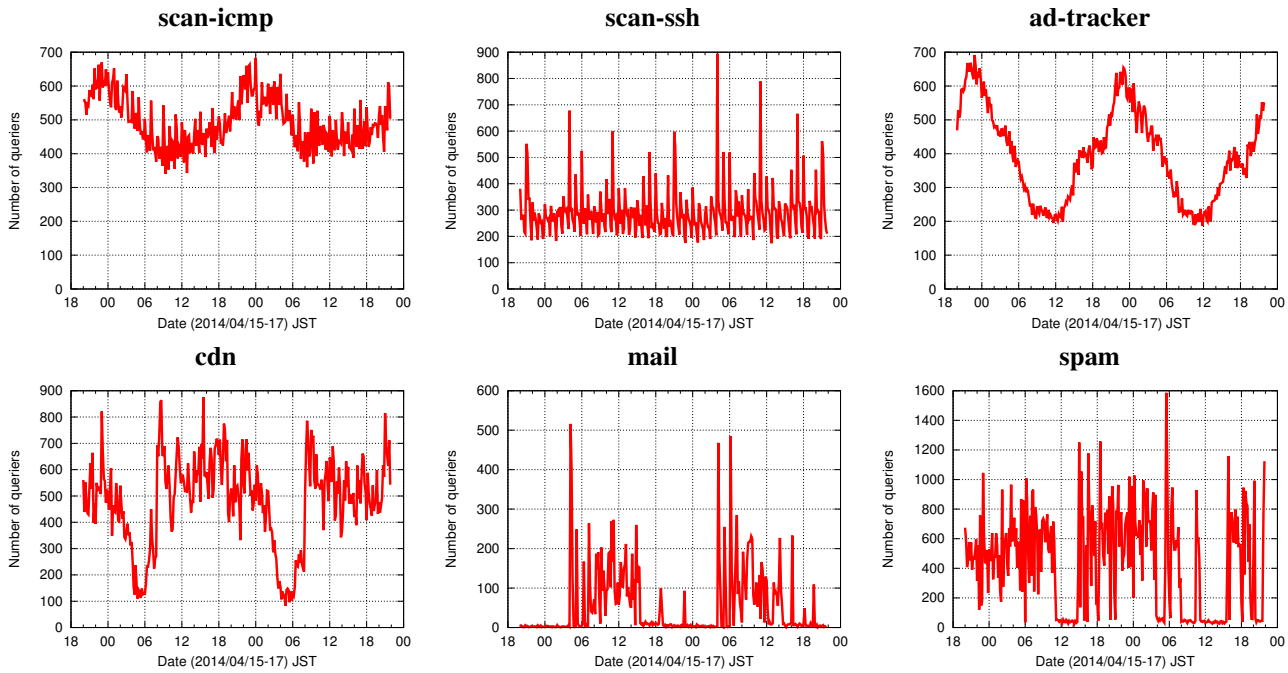

Fig. 16: Counts of queries per minute over 50 hours for Daily variation in number of queriers for our case studies. (Dataset: JP-ditl.)

Scan-ssh and spam show little correspondence to time-of-day. We suspect both are automated, although lulls in spam are perhaps due to initiation of different spam activity.

Finally, scan-icmp is somewhat unusually—it is fully automated but shows a diurnal pattern, unlike scan-ssh. This outage detection algorithm is adaptive, probing more persistently when networks are less active [43]. Recent work shows that in many countries the IP address space is less used at night [44], explaining why this robot shows diurnal traffic.

APPENDIX D
ACKNOWLEDGMENTS

We thank Yuri Pradkin for B-Root data collection, Akira Kato for M-Root data, and Yoshiro Yoneya and Takeshi Mitamura for JP data. We thank Xun Fan for input about Google and Akamai sites in Japan. We thank Terry Benzel, Kenjiro Cho, Ethan Katz-Bassett, and John Wroclawski comments on this paper.

Kensuke Fukuda's work in this paper is partially funded by Young Researcher Overseas Visit Program by Sokendai, JSPS KAKENHI Grant Number 15H02699 and 15KK0019, and the Strategic International Collaborative R&D Promotion Project of the Ministry of Internal Affairs and Communication in Japan (MIC) and by the European Union Seventh Framework Program (FP7/2007- 2013) under grant agreement No. 608533 (NECOMA). The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the MIC or of the European Commission.

John Heidemann and Abdul Qadeer's work in this paper is partially sponsored by the Department of Homeland Security (DHS) Science and Technology Directorate, HSARPA, Cyber Security Division, via SPAWAR Systems Center Pacific under Contract No. N66001-13-C-3001, and via BAA 11-01-RIKA and Air Force Research Laboratory, Information Directorate under agreement number FA8750-12-2-0344. The U.S. Government is authorized to make reprints for Governmental purposes notwithstanding any copyright.The views contained herein are those of the authors and do not necessarily represent those of DHS or the U.S. Government.

Kensuke Fukuda is an Associate Professor at Information Systems Architecture Research Division, National Institute of Informatics and Department of Informatics, School of Multidisciplinary Sciences, The Graduate University for Advanced Studies (SOKENDAI). He received his PhD in 1999 from Keio University. His research interests span Internet traffic analyses, anomaly detection, modeling networks and QoS over the Internet.

John Heidemann is Research Professor and Senior Project Leader at USC / ISI, Marina Del Rey, California, USA. He received his PhD in 1995 at University of California, Los Angeles. His research interests include observing and analyzing Internet topology and traffic to improve network reliability, security, protocols, and critical services.

Abdul Qadeer is a PhD student at University of Southern California, Los Angeles, USA. He works with Analysis of Network Traffic group at USC's Information Sciences Institute, Marina Del Rey. His broad research interests include distributed systems, computer networks, operating systems, and computer architecture.