

DNS Noise: Measuring the Pervasiveness of Disposable Domains in Modern DNS Traffic

Yizheng Chen*, Manos Antonakakis†, Roberto Perdisci‡, Yacin Nadji*, David Dagon*, Wenke Lee*

*College of Computing, Georgia Institute of Technology, {yizheng.chen,yacin.nadji,wenke}@cc.gatech.edu, dagon@sudo.sh

†School of Electrical and Computer Engineering, Georgia Institute of Technology, manos@gatech.edu

‡Department of Computer Science, University of Georgia, perdisci@cs.uga.edu

Abstract—In this paper, we present an analysis of a new class of domain names: *disposable domains*. We observe that popular web applications, along with other Internet services, systematically use this new class of domain names. Disposable domains are likely generated automatically, characterized by a “one-time use” pattern, and appear to be used as a way of “signaling” via DNS queries. To shed light on the pervasiveness of disposable domains, we study 24 days of live DNS traffic spanning a year observed at a large Internet Service Provider. We find that disposable domains increased from 23.1% to 27.6% of all queried domains, and from 27.6% to 37.2% of all resolved domains observed daily. While this creative use of DNS may enable new applications, it may also have unanticipated negative consequences on the DNS caching infrastructure, DNSSEC validating resolvers, and passive DNS data collection systems.

Keywords-Disposable Domain Name; Internet Measurement.

I. INTRODUCTION

The domain name system (DNS) is a critical component of the Internet that maps human-readable names to machine-level IP addresses. Over the years as the Internet evolved, more and more service providers use the DNS in ways for which it was not originally intended. Their primary objective is to make their network operations more *agile* and *scalable*. Such use cases are often found in content delivery networks (CDNs) [1], NXDOMAIN rewriting [2], and URL auto-completion and prefetching [3].

In this paper, we describe a new class of DNS misuse called *disposable domains*. Recently, a number of service providers, such as popular search engines [4], social networks, and security companies [5], began to heavily use automatically generated domain names to convey “one-time signals” to their servers. These disposable domains are often created on demand in large volumes and belong to common parent DNS zones (i.e., same name suffix). Moreover, disposable zones have unique cache hit rate distributions that distinguish them from non-disposable zones.

While these creative ways of using the DNS enable new useful applications and performance improvements for certain types of Internet services, the increasing use of disposable domains may have unanticipated and even negative impacts on day-to-day DNS operations for large Internet Service Providers. Firstly, disposable domain

names are only queried a few times by a handful of clients. However, when a large number of disposable domains come into existence, their queries may fill up the cache of local DNS resolvers. Such an event may cause premature cache evictions of non-disposable domains, and in turn cause DNS service degradation within the ISP network. Secondly, these premature evictions may inflate the traffic between the DNS resolvers and authoritative name servers. The increased traffic can cause DNSSEC-enabled resolvers to perform extra cryptographic operations. Lastly, the pervasiveness of disposable domains in modern DNS traffic can cause a significant increase in the storage cost for passive DNS databases, which are vital for domain reputation systems [6], [7], [8], and represent an irreplaceable tool for the forensic analysis of network security incidents [9], [10], [11], [12].

It is therefore important for the research and operational communities to carefully monitor and analyze the evolution of the DNS usage in today’s Internet. It is also necessary to understand under what conditions the current DNS practices employed by various service providers may result in unexpected operational problems in the near future. In this paper, we design a system to automatically discover DNS zones that use disposable domains and present detailed measurements on how disposable domains are being used by large service providers. Specifically, we make the following contributions:

- We present a study from large scale DNS traffic traces collected at a large north American ISP (namely Comcast) serving millions of end users. Our measurements show, among other interesting facts, that a very significant percentage, 25% of all queried domain names, 33% of all resolved domain names, and 60% of all distinct resource records observed daily are *disposable*.
- In order to properly monitor and measure the network presence of the disposable domains we propose a novel algorithm that automatically finds DNS zones that contain disposable domains. Our algorithm accurately discovers disposable domains by passively monitoring DNS traffic, with 97% true positive and 1% false positive rates. Using our system, over the period of 11

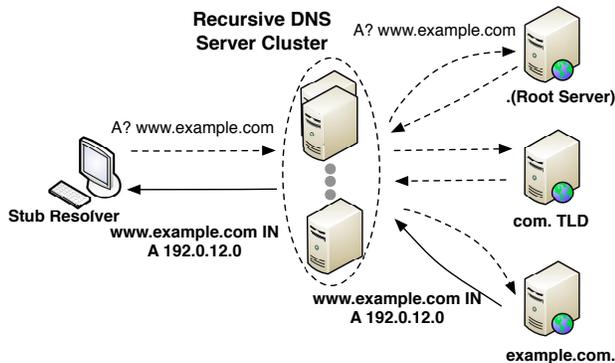


Figure 1: DNS query resolution process.

months, we discovered 14,488 new disposable zones.

- We discuss the possible negative implications that the growth of disposable domains may have on the DNS caching infrastructure, DNSSEC-validating resolvers, and passive DNS data collection systems.

The rest of the paper is organized as follows. In Section II, we provide background on DNS and discuss related work. In Section III, we describe our data collection process and provide an overview of the characteristics of modern DNS traffic observed in our dataset. In Section IV, we define disposable domains and examine their key properties. In Section V, we provide details of our disposable domain miner. In Section VI, we discuss the negative impacts disposable domains have on DNS cache, DNSSEC, and passive DNS database. We conclude the paper in Section VII.

II. RELATED WORK

A. DNS Concepts and Terminology

In most cases, establishing an Internet connection from a client to a server begins with a DNS resolution that maps a domain name (e.g., `www.example.com`) to an IP address (e.g., `192.0.12.0`). As shown in Figure 1, the client (stub resolver) first issues a query to the Recursive DNS server (RDNS). If the resolution request from the client is not in the cache, the RDNS will perform an iterative query. This process begins at the root server and works its way down through the top level domain name (TLD) server and name server of `example.com` until the RDNS server receives the current DNS answer for the original client’s request. Finally, the RDNS server replies to the client with the answer received from the name server of `example.com`.

B. Related Work

1) Passive DNS and DNS Traffic Aggregation:

Weimer [13] was the first to propose passive DNS replication for forensic analysis and network measurement. The implementation *dnstap* passively collects DNS data from a production network to keep historic DNS information. Plonka et al. [14] built *treemap* to collect and analyze passive DNS traces. They separate traffic into three categories:

canonical, overloaded and unwanted. They showed that spikes of DNS traffic are typically unwanted or overloaded traffic. In their taxonomy, unwanted DNS traffic comprises all unsuccessful DNS resolutions (i.e., NXDOMAINs). DNS traffic with purposes beyond mapping domains to IPs are considered overloaded, while the rest are canonical. At that time, the primary application of overloaded DNS traffic was for blacklisting purposes. Disposable domains are more general than the overloaded class. We study DNS zones used for various services in addition to blacklisting.

2) *DNS Traffic Analysis*: CDNs are traditionally used for dynamic request routing via resolution management [1]. Similarly, many Internet services use “domain sharding” to allow parallel client queries to web content [15]. Vixie [16] pointed out numerous problems with DNS-based load balancing. While his work notes the potential decrease in the effectiveness of caching, Vixie’s analysis focused on DNS policy, such as “NXDOMAIN Remapping” for commercial gains, rather than the cache consumption caused by disposable domains. Our work expands on these issues by providing experimental results for the caching performance of disposable domains in general and revealing yet another misuse of the DNS. Work done by Yadav et al. [17] detects algorithmically-generated malicious domain names. Disposable domains are not only generated by an algorithm, but also have low cache hit rate and are not necessarily malicious. Berger et al. [18] studied the dynamics of DNS and proposed stability metrics to classify dynamic and stable domain names. In contrast, our definition of disposable domains is a distinct category. Paxson et al. [19] built a practical system for detecting DNS covert channels, enforcing a 4kB/day information bound after lossless compression for enterprise environment, per user, per destination. However, disposable domains can be stealthy and stay under this threshold. Nevertheless, we can identify them collectively from the view of the entire disposable zone.

3) *DNS Cache Modeling*: Jung et al. [20] presented a trace-driven simulation to measure cache hit rates. Later, they [21] proposed a cache hit rate model based on the renewal model of inter-query arrival times and the *Time To Live* (TTL) values in DNS cache records. They assume (1) every data item has the same TTL value, and (2) a group of clients share a common cache, without local caches present in the client machines such that DNS requests can be inferred from TCP connections. Since their assumptions are not true in our ISP Recursive DNS Server monitoring scenario, we take a black-box analysis approach by evaluating the performance of a server cluster with multiple independent caches.

III. DATA COLLECTION

In this section, we first describe the methodology used to collect the DNS datasets for our study and explain our network visibility within a large North American ISP. Then,

we analyze the collected datasets, and elaborate on the characteristics of modern DNS traffic. This panoramic view of real-world DNS messages is instrumental to the analysis of “disposable” domains that we present in Section IV.

A. Traffic Collection and Datasets

We have visibility of all DNS traffic to and from the recursive DNS (RDNS) servers of a large ISP in the Midwestern US. For quality of service reasons (e.g., load balancing and fault tolerance), the DNS queries from the ISP customers are served by a *cluster* of RDNS servers. This is a fairly typical configuration for recursive DNS servers in large ISPs.

We can monitor traffic “above” and “below” the RDNS servers. For example, considering the DNS resolution scenario in Figure 1, from our network monitoring point, we were able to observe events indicated by solid arrows: (1) DNS responses from the RDNS servers to the client (the stub resolver) (“below” the RDNS servers), and (2) DNS responses from the authoritative name servers to the RDNS servers (“above” the RDNS servers). We only record the answer section of the DNS response packets, which reflect answered queries from client and from RDNS server cluster.

To perform our measurements, we use two types of DNS datasets: a *full* passive DNS (fpDNS) dataset, and a *reduced* passive DNS (rpDNS) dataset. The fpDNS dataset includes all DNS traffic observed at the monitoring point. Each entry in the fpDNS dataset is a resource record (RR), a tuple containing the timestamp of the DNS resolution event (in the granularity of seconds), an anonymized client ID of the host that issued the DNS request, the queried domain name, the DNS query type, the time-to-live value (TTL), and the resolution data (RDATA) contained in the response. The query types in our dataset are A, CNAME, AAAA types. The rpDNS dataset includes the distinct (no duplicates) resource records (RRs) from all *successful* DNS resolutions observed from the same monitoring point in the ISP. DNS requests with no valid response, such as NXDOMAIN, are excluded. We represent the rpDNS dataset in a given day, as tuples containing the queried domain name, the DNS query type, RDATA, and the first date the tuple was seen.

Since we are dealing with real-world ISP-level DNS traces, the size of the compressed fpDNS dataset is around 60GB per day in February, and around 145GB per day in December, 2011. Therefore, we have limits on the amount of data we can obtain over a certain period and the overall monitoring period for our experiment. To that extent, we built the fpDNS dataset using data collected over 24 days: from 02/01/2011 to 02/07/2011, 09/02/2011, 09/13/2011, 11/14/2011, from 11/28/2011 to 12/10/2011, and 12/30/2011. The total size of our fpDNS dataset is 2.67TB. On the contrary, the rpDNS dataset has smaller storage size, as it only contains distinct RRs, with a size of seven to nine GBs per day. Overall, the rpDNS dataset

includes the deduplicated resource record data derived from the fpDNS dataset, for 11/28/2011 to 12/10/2011.

B. Notation

Next, we describe some notation that we will use throughout the rest of the paper. A domain name d consists of a set of labels (or substrings) separated by a period. We refer to the *effective* rightmost label as the *top-level* domain (TLD). This segment captures the delegation aspects of the zone, and not merely a lexical splitting of the domain name. For example, we treat `com.cn` and `co.uk` as effective TLDs, since all further child labels under those zones represent name server delegations, usually to separate organizations and entities, who in turn control what resides in the child zone. This approach is similar to the “public suffix list” from Mozilla [22]. Our definition is a superset of this effort, and corrects the omission of dynamic DNS zones.

The *second-level* domain (2LD) represents the two rightmost child labels separated by a period. Similarly, the *third-level* domain (3LD) consists of the three rightmost labels, and so on. In general, the N^{th} -level domain (NLD) refers to the N rightmost labels. For instance, given domain name $d = a.example.com$, $TLD(d) = com$, $2LD(d) = example.com$, and $3LD(d) = a.example.com$. Throughout the text, we use the notion of “zone” loosely. It can be 2LD, 3LD, or any N^{th} -level domain. We provide further clarifications on the notion of a zone as necessary.

C. Full Passive DNS Database

Before we introduce the notion of “disposable” domain names, we provide some insights from analyzing the fpDNS dataset. From a high-level view, the most interesting properties are the traffic volumes above and below the RDNS servers, the caching properties, and the deduplicated resource record volumes. These analyses will provide some tell-tale signs for disposable domain names, which the DNS community has not thoroughly defined nor studied.

In February there were 4.2 billion RRs observed below the RDNS servers, and 500 million RRs above them. In December the volume increased to 10 to 11 billion RRs observed below the RDNS servers, and 800 million RRs above them. Moreover, in December, we observe approximately 30 million unique domain names every day, where 20 million of them were successfully resolved.

1) *DNS Traffic Volume*: First, we examine the DNS resource record (RR) volumes above and below the recursive DNS servers. As Figure 2 shows, there is an order of magnitude less traffic above the recursive servers than below, as a result of caching. Moreover, we can clearly observe the human-driven diurnal effect on DNS traffic (e.g., the traffic volume dropped after midnight and rose at 10am local time).

In order to put these observations about the DNS resource record volumes into perspective, we selected two of the most

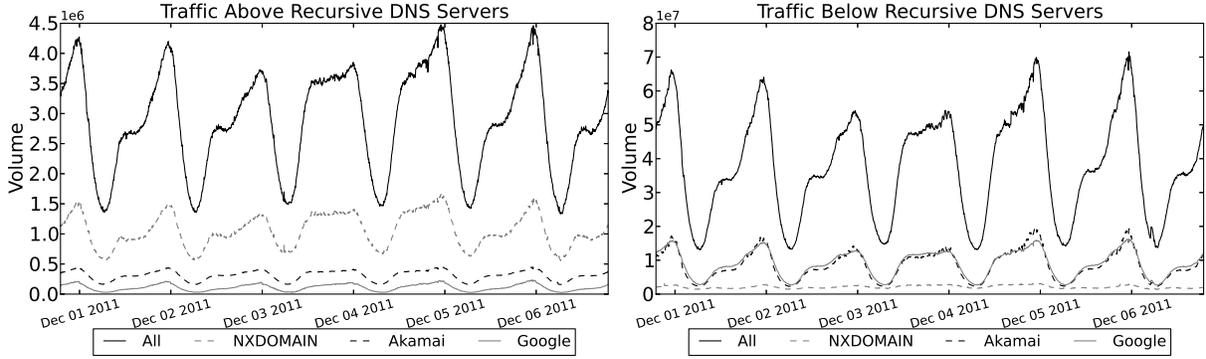
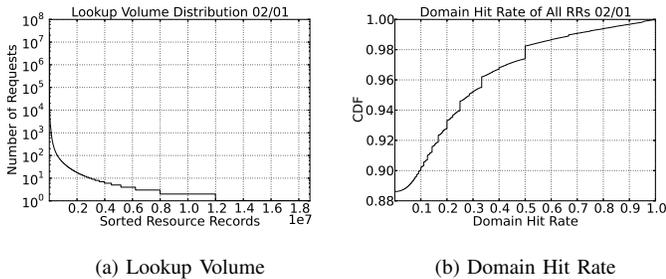
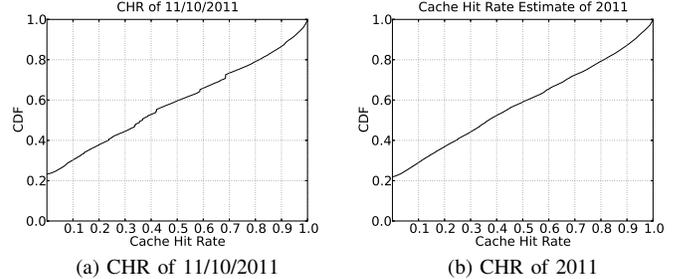


Figure 2: Traffic profile of fpDNS dataset, from 12/01/2011 to 12/06/2011.



(a) Lookup Volume (b) Domain Hit Rate
Figure 3: Long tail of lookup volume and domain hit rate.



(a) CHR of 11/10/2011 (b) CHR of 2011
Figure 4: Cache hit rate distribution from fpDNS.

popular 2LD zones, Google and Akamai¹, and placed them alongside the overall numbers. Google reflects user-driven behavior, such as checking emails or web searches. Zones from Akamai reflect the DNS activity for the largest content delivery network. These two popular zones collectively account for less than half of the total DNS traffic, which clearly shows that there are other zones contributing a non-negligible portion of traffic to our fpDNS dataset.

Additionally we plot in Figure 2 the unsuccessful DNS resolutions (NXDOMAIN). The NXDOMAIN traffic constitutes almost 40% of the traffic above the RDNS servers, and only 6% of traffic below the RDNS servers. This is likely because the resolvers in the monitored networks were not honoring the negative cache, ignoring RFC2308 [23].

We consider the *long tail of lookup volume* to be domain names that receive fewer than 10 lookups per day. In fact, more than 90% of all RRs have lookup volumes lower than 10 on 02/01/2011 (Figure 3a). Moreover, the long tail of lookup volume increased from 90% to 94% in 2011.

2) *DNS Cache Hit Rates*: In order to present the cache hit rate (CHR) observations from the fpDNS dataset, we first define domain hit rate. We consider the domain hit rate of an *object* in the following way:

$$DHR(object) = \frac{\text{Number of Cache Hits in a Day}}{\text{Number of Total Queries in a Day}} \quad (1)$$

We consider a resource record to be the storage *object* in the cache. Every cache hit corresponds to an answer

¹Google: google.com. Akamai: akamai.com, akamai.net, akamaiedge.net, akamaihd.net, edgesuite.net, akamaitech.net, akadns.net, akam.net.

issued from the RDNSs observed below the recursive DNS servers that does not trigger a cache miss. Every cache miss corresponds to an answer issued to the RDNSs observed above the recursive DNS servers. The number of all queries is simply the sum of the answers seen below the recursive DNS servers.

The domain hit rate distribution shows the caching performance of all distinct RRs. For example, Figure 3b presents the cumulative distribution of DHR for 02/01/2011. We can see that 89% of all RRs have domain hit rate of 0%, as part of the DNS long tail phenomenon. Here, we consider the *long tail of domain hit rate* to be domain names with domain hit rate of 0%. Also, we observe that the percentage of RRs with zero domain hit rate increased from 89% to 93% in 2011.

Based on domain hit rate, we define cache hit rate. Given our visibility above and below the recursive DNS servers, and our inability to gain access to the actual recursive DNS software, we choose to treat the recursive DNS servers as a “black box”. In the renewal counting process [21], we are interested in the number of cache hits every time an object is updated in the cache, i.e., every time there is a cache miss. However, we are unable to track the exact hits per cache miss, so we simplify all the hit rates for the same RR as the domain hit rate for the day. For instance, an object can trigger one cache miss with three queries, and another cache miss with two queries, resulting in 0.66 and 0.5 cache hit rate values, respectively. However, what we can measure is that the object triggered 2 cache misses and there were

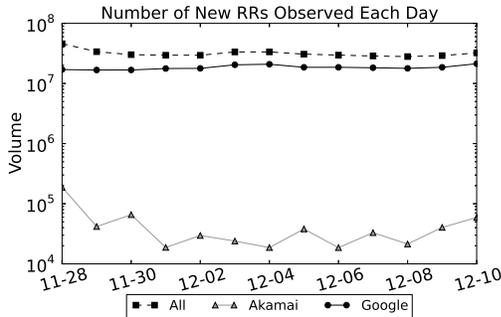


Figure 5: Deduplicated new resource records per day in the rpDNS Datasets from 11/28/2011 until 12/10/2011.

5 total queries for the object in a day, so we consider the cache hit rate to be 0.6 for all 2 misses. More formally, we define the cache hit rate of an *object* as following:

$$CHR_i(object) = DHR(object) \quad (i = 1, 2, 3, \dots, n) \quad (2)$$

$n = \text{Number of Cache Misses in a Day.}$

The cache hit rate distribution is the cumulative distribution of all CHR_i values for all RRs. Figure 4a presents the distribution of CHR for 11/10/2011. The CDF looks like a slightly skewed linear line. The figure shows that 58% cache hit rates are lower than 50%. We also measured the CHR distribution from 13 days (in 2011), which can be seen in Figure 4b. The long-term cache hit rate distribution also follows a similar skewed linear line. Although the distribution approximates each cache hit rate value by the same domain hit rate in the day, we show in Section IV-B that this type of distribution can distinguish between disposable zones and non-disposable zones accurately. Since the distribution reflects the effect of query volume, domain hit rate, and implicitly the TTL, we are able to capture all the information in our classification process by using this distribution.

3) *DNS Deduplication*: We built a reduced passive DNS dataset from our full passive DNS dataset, using 13 days of traffic from 11/28/2011 until 12/10/2011. We deduplicated all the resource records seen during these 13 days, yielding 413,753,934 unique resource records in total.

The volume distribution of newly observed RRs for each day in the rpDNS dataset is shown in Figure 5. It is worth noting that the number of new RRs observed every day decreased by 13,614,102 (30%) on the 13th consecutive day. Looking at the new Akamai RRs, we also observed a slight decrease by 128,957 (69%) records on the 13th day.

An important 2LD zone we explicitly examine here is *google.com*. Despite what we saw as trends from Akamai and the overall rpDNS dataset, Google increases its daily new RRs by 4,264,585 (25%) on the 13th consecutive day. In fact, Google went from 17,015,510 new unique RRs the first day to 21,280,095 new unique RRs the 13th day.

An even more interesting observation is that Google operates 58% of all the RRs in the overall rpDNS dataset. Looking into the actual percentage of unique RRs every day,

Google is responsible for the 37% of the unique RRs on the first day. However, it is responsible for 66% of unique new RRs on the 13th day. It means that Google is constantly producing new RRs as part of its normal DNS operation and these RRs are not reused, effectively making them temporary or “one-time”. In Section V-C, we will elaborate on this DNS phenomenon. We will see that Google utilizes a large number of disposable domains, for what appears to be a measurement experiment over DNS. Below in Section VI, we argue that such use is disposable when the cache hit rate is low or zero, and the TTL is nonetheless non-zero (i.e., placing records in cache that will never be re-queried). In the following section, we will precisely define disposable domains.

IV. DEFINING DISPOSABLE DOMAINS

In this section, we define disposable domain names and elaborate on two key properties: the structure of the DNS zone that facilitates resolutions for disposable domain names and the cache hit rates observed from disposable resource records. Disposable domain names are successfully resolved domain names that have the following two properties:

1. Their name strings are automatically generated. Namely, some software generates them in bulk using an algorithm.
2. The RRs under a given zone are only observed once, or a handful of times, when they are in the recursive DNS servers’ cache. More formally, the RRs of child domains under the zone have a low or close to zero median value in cache hit rate distribution ².

The first property helps us focus on domain names generated automatically. However, being automatically generated is a necessary but insufficient condition to characterize a domain as disposable. In order to fully capture the notion of disposable domains, we must examine their caching properties. An automatically generated domain should be marked as disposable when the cache hit rate of its resource record is very low, and all RRs under the same zone, that are effectively generated by the same algorithm, share similarly low cache hit rates.

Note that because of the definition of the cache hit rate, domains under a zone could be disposable in one network but not another. Since we focused on discovering disposable zones in our network’s traffic, this definition allows us to find these zones and does not preclude our approach from generalizing to other networks. Comparing disposable zones among different networks can help discover globally disposable zones. Due to the coverage of our ISP, however, we expect many of the disposable zones discovered in our network to be disposable in other networks as well.

A. Motivating the DNS Zone Structure

In this subsection we provide three real world examples of zones that facilitate resolutions of disposable domain

²Cache hit rate distribution is defined in Section III-C2.

```
load-0-p-01.up-1852280.mem-251379712-24440832-0-p-50.swap-236691456-297943040-0-p-44.3302068.1222092134.device.trans.manage.esoft.com
load-0-p-49.up-1066332.mem-118550528-17743872-0-p-49.swap-186757120-347877376-0-p-35.3300639.1643250616.device.trans.manage.esoft.com
load-0-p-90.up-41144.mem-193540096-523649024-0-p-19.swap-56713216-477921280-0-p-11.3303042.3049260335.device.trans.manage.esoft.com
load-0-p-08.up-117864.mem-76529664-15839232-0-p-29.swap-13049856-529776640-0-p-02.8551447.2050639502.device.trans.manage.esoft.com
load-0-p-01.up-122977.mem-76460032-16359424-0-p-29.swap-13180928-529645568-0-p-02.8551447.2050639502.device.trans.manage.esoft.com
```

(i)

```
0.0.0.0.1.0.0.4e.135jg5e1pd7s4735ftrqweufm5.avqs.mcafee.com
0.0.0.0.1.0.0.4e.13cfus2drmdq3j8cafidezr8l6.avqs.mcafee.com
0.0.0.0.1.0.0.4e.13kqas3qjj46ttkdhastkrds6.avqs.mcafee.com
0.0.0.0.1.0.0.4e.13pq3hfpunqn1d51pmvbdkk5s6.avqs.mcafee.com
0.0.0.0.1.0.0.4e.13qh71bf782qb54uzz9uhdz4mq.avqs.mcafee.com
```

(ii)

```
p2.a22a43lt5rwfg.ihg5ki5i6q3cfn3n.191742.i1.ds.ipv6-exp.l.google.com
p2.a22a43lt5rwfg.ihg5ki5i6q3cfn3n.191742.i2.v4.ipv6-exp.l.google.com
p2.a22a43lt5rwfg.ihg5ki5i6q3cfn3n.191742.s1.v4.ipv6-exp.l.google.com
p2.a22antzfkdg5g.nay6cy6qq26fr64b.544760.i1.v4.ipv6-exp.l.google.com
p2.a22antzfkdg5g.nay6cy6qq26fr64b.544760.i2.ds.ipv6-exp.l.google.com
```

(iii)

Figure 6: Sample of disposable (I-III) domain names.

names. We will examine some key properties that disposable domain names have by examining passive DNS datasets from major zones like *google.com*. In Figure 6, we can see a few sample domain names from three zones that are disposable. These three zones operated under the control of eSoft (i), McAfee (ii), and Google (iii).

The first example is eSoft, which appears to be a service that employs DNS as a storage communication channel in order to report CPU load, machine up time, memory usage and swap disk usage. For the second example, according to McAfee [5], domains shown in (ii) are used for file reputation queries on behalf of their Global Threat Intelligence File Reputation Service. If any suspicious program executable, Android Application Package File (APK) or Portable Document Format (PDF) file is not detected as malicious by signatures of user’s local Anti-Virus software, the software will generate DNS queries for file classification result from the cloud. A suspicious file is defined to be any file with certain characteristics that malware commonly has, such as whether the executable file is packed. The queried name is typically less than 40 byte, including McAfee version and product information, hash of the suspicious file, fingerprint information, and environmental information. The returned answer from McAfee file reputation server is typically a non-routable IP address in 127.0.0.0/16, where different IP address has different meaning. Lastly, domains shown in (iii) are generated by Google’s IPv6 experiment [4]. A small percentage of Google users are selected for the experiment. Browsers of selected users perform cryptographically signed background requests after users search and get the results. The background requests record IPv4 and IPv6 addresses, image request latency, and User-Agent string for browser and operating system.

Examining the zone structures from Figure 6 shows that the randomly generated part is not always the leftmost child label of the domain. For example, *ipv6-exp.l.google.com* (iii) and *avqs.mcafee.com* (ii) have the leftmost labels (p2 and 0), which are not “random-looking”. Therefore, we need to check whether each group of labels between “.” are generated by an algorithm. Furthermore, disposable domains under the same section of the DNS zone always have the same number of periods (“.”) in the domain. This is probably

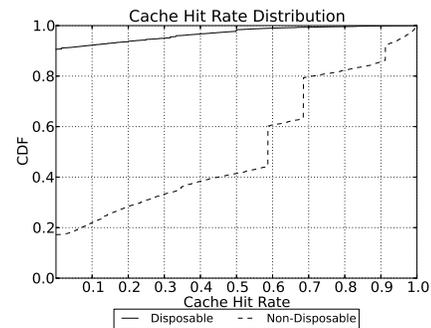


Figure 7: Cache hit rate distribution for disposable and non-disposable zones.

due to the specific protocol used by the zone operator. For example, disposable domains under *avqs.mcafee.com* always have 11 periods in the domain. We must consider the actual structure of the domain names in order to generate statistical features that can be used to identify disposable domains.

B. Motivating the Cache Hit Rate

In general, resource records of disposable domain names are used only once or up to a few times while they are in the recursive DNS servers’ cache. This means that disposable RRs have very low or zero cache hit rates when they are updated in the cache. On the other hand, we observe that non-disposable RRs have relatively good cache hit rates.

We manually labeled 398 zones as disposable, and 401 randomly selected 2LD zones from the top 1,000 Alexa domain names as non-disposable, from traffic observed on 11/10/2011. While there are usually thousands or millions of unique disposable domains seen under disposable zones, we took a conservative approach to include zones with as few as 15 disposable domains because of our limited observation window. Figure 7 shows that 90% of cache hit rates from disposable RRs are zero. On the other hand, 45% of cache hit rates from non-disposable RRs are over 0.58.

Disposable zone operators do not seem to make use of the caching benefit of recursive DNS infrastructure because they use their disposable domain names as temporary domains. Disposable domains are not strictly looked up once only, since software making those queries can sometimes generate the same domain name again. However,

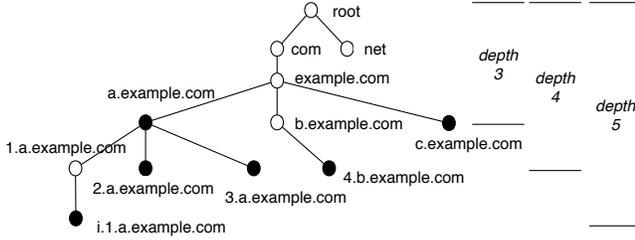


Figure 8: An example of a Domain Name Tree.

when any disposable domain is looked up by anyone in the lifetime of TTL value, it is highly unlikely that the same domain name will be used by any other client. Since disposable zone operators want to have full control over every record under their zone so they can leverage the recursive DNS servers as their temporary infrastructure for purposes other than providing IP addresses. This extra level of control under the disposable zone is very important when operators (i.e., eSoft) want to deliver content in the domain names, use the zone as a channel for customized protocols (DNSBLs, AntiVirus Companies, DNS Tunneling Services), or even to collect metrics (Google IPv6 experiment).

However, a non-disposable zone is not likely to exhibit such *overall* poor caching performance, given all domains under the same parent zone. Lookups to non-disposable RRs are less controlled by the zone operator, since non-disposable domains do not serve one-time purposes. Consequently, non-disposable zones would have a more “natural” cache hit rate distribution, which looks more like the linear cumulative distribution for all resource records in Figure 4.

V. MINING DISPOSABLE DOMAINS

In this section we describe the disposable zone miner we design, implement and use in order to measure the prevalence of disposable domain names in ISP networks. We begin by presenting the necessary features to automatically discover disposable domain names. We then discuss how these features can be used in our disposable zone miner. We conclude this section by providing measurement results from the actual use of the disposable zone miner in a large North American ISP.

A. Statistical Features

We first present the necessary notation used to describe the two statistical feature families. Then we present and motivate the feature families used to transform the DNS zone information into statistical vectors for mining disposable zones.

1) *Domain Name Tree Definition*: For a given set of domain names, we generate a *domain name tree*. The root of the tree is “.” (*root*), the children of the root are the *TLDs*, the children of the *TLDs* are the *2LDs*, and so on.

We categorize the nodes in the tree as black nodes or white nodes. We consider a black node to be every node that has a resource record (RR) in our DNS dataset within the observation period, and the rest are white nodes.

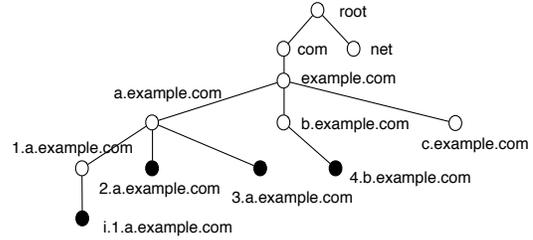


Figure 9: Domain Name Tree after decoloring two nodes.

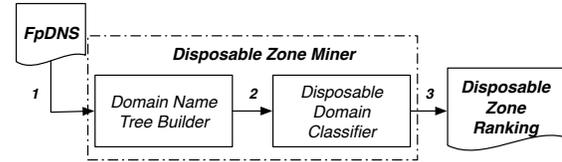


Figure 10: Daily Disposable Zone Ranking Process.

Figure 8 shows the **domain name tree** for the set of RRs of following domain names:

a.example.com, i.1.a.example.com,
2.a.example.com, 3.a.example.com,
4.b.example.com, and c.example.com.

In the tree structure, nodes a.example.com, b.example.com and c.example.com are child nodes of node example.com. Nodes 1.a.example.com, 2.a.example.com and 3.a.example.com are child nodes of node a.example.com. All the nodes under example.com are its descendants. Colored nodes are black nodes, while the others are white nodes. If any node is decolored in the tree, it turns from a black node to a white node. For example, decoloring a.example.com and c.example.com result in the tree in Figure 9.

Based on the structural observations of disposable domain names discussed in Section IV-A, we next group nodes with the same structure. We define the *depth* of a black node as the length of the path up to the root. Nodes within the same group G_k have the same depth k . For all the black descendants of the same zone if they have the same depth, we consider them to have the same structure. For example, to group black nodes under example.com, we would get $G_3 = \{a.example.com, c.example.com\}$, $G_4 = \{2.a.example.com, 3.a.example.com, 4.b.example.com\}$, $G_5 = \{i.1.a.example.com\}$. All groups of domain will be classified either as disposable or non-disposable by Disposable Domain Classifier module, as we will see next.

Our goal is to build statistical features to describe nodes within the same group. We compute six tree-structure features and two cache hit rate features for each set G_k . In order to compute tree-structure features, we need to get the set of labels for each G_k to see whether they are algorithmically-generated. For the previous example, we take the following sets of labels $L_3 = \{a, c\}$, $L_4 = \{a, b\}$, and $L_5 = \{a\}$, that are next to the zone under inspection (i.e. example.com).

2) *Feature Families*: We now discuss the two feature families and the main motivation behind their selection.

Tree Structure Features: For each set G_k , we calculate the corresponding set L_k . Let the Shannon entropy of the characters in the label l be $H(l)$. For all the labels $l_i (i = 1 \dots m)$ in the set L_k , we compute the entropy values $H(l_i)$. We then use as features the cardinality m of the set L_k , the maximum, minimum, average, median, and variance of all $H(l_i)$ values.

Cache Hit Rate Features: For each set G_k , we calculate the domain hit rate (as defined in Section III-C2) and the number of misses for all the resource records of domains in the set G_k , to generate the cache hit rate distribution. From the distribution, we take the median, and the percentage of RRs that have zero cache hit rate as two statistical features for this family.

Features and Group Intuition: In Section IV and Figure 6, we discussed the main zone structural properties of disposable domain names. We saw that operators tend to use algorithms to create domain names “in bulk” under certain levels of the master zone. With the G_k sets, we may capture the properties of the nodes being created by the operators in the same depth from the root of the DNS tree.

The meaning of the entropy features computed over the labels in the corresponding L_k sets are twofold. First, we simply want to see if there are any labels generated by algorithms at the same level of the tree, which could indicate disposable domain names. Second, we want to see if there are outliers in the L_k set using the variance as a guide. For example, a percentage of the nodes in the set are used for disposable domain names. However, there could be some nodes that are created manually and serve non-disposable domain names. We would like to be able to capture these zone characteristics during modeling.

Finally, the cache hit rate features are very influential in our effort to differentiate between disposable and non-disposable G_k sets. As we have extensively discussed in Section IV, median values in the cache hit rate distribution for resource records of non-disposable domain names are significantly higher than the disposable ones. The cache hit rate features provide us with the necessary classification signal to properly model disposable domains.

B. Overview of the Mining System

In Figure 10, we present a process to systematically track and rank zones that facilitate resolutions for disposable domain names over the period of a day. As the daily DNS dataset is being collected (Step 1), it is fed into our system. We first build the Domain Name Tree that reflects the current DNS dataset. This is done by the *Domain Name Tree Builder*, so the *Disposable Domain Classifier* can traverse (Step 2) the zones of the domain name tree, according to Algorithm 1. The output of the miner is (Step 3) the disposable classification score for each zone in the tree.

Algorithm 1 Disposable domain name classification process given the under inspection zone z .

```

1: if There is no black descendants for  $z$  then
2:   return
3: end if
4: From all the black descendants of  $z$ , identify  $G_{k_i}$  and
   generate  $L_{k_i}$ , where  $i = 1, 2, \dots, n$  and  $n$  is the number
   of different depth values under zone  $z$ .
5: Set classifier threshold  $\theta = 0.9$ 
6: for  $i = 1$  to  $n$  do
7:    $p, class = \mathcal{C}(G_{k_i})$ 
8:   if  $class == disposable$  and  $p \geq \theta$  then
9:     for  $j = 1$  to  $m$  (number of nodes in  $G_{k_i}$ ) do
10:      Decolor  $node_j$  in  $G_{k_i}$ 
11:     end for
12:     output  $z, k_i$ 
13:   end if
14: end for
15: for All the child nodes of  $z$  do
16:   Run Algorithm 1
17: end for

```

1) *Domain Name Tree Builder*: This module processes the full passive DNS dataset for the system. Its main functionalities are: i) to assemble the daily domain name tree, and ii) to gather the cache hit rate information for RRs of the resolved domain names. In the domain name tree, we can easily get the depth of black nodes, so when necessary, it can efficiently gather domain names and provide the corresponding *Tree Structure Features* and *Cache Hit Rate Features*.

2) *Disposable Domain Classifier*: The classifier module traverses the domain name tree and classifies the set of domain names in the full passive DNS dataset for a single day. The mining process is composed of two main parts. First, the Algorithm 1 starts with all the effective 2LDs in the domain name tree. Then the algorithm identifies groups of black descendants with the same depth under a zone. Next, the algorithm will generate the corresponding sets G_k and L_k for all possible depth values of k (Line 4, Algorithm 1). Second, the mining process will produce a new statistical model from known zones that facilitate resolutions for disposable domains. And the classifier will classify all the groups in an effort to identify new disposable domain names (Line 6 to 14). Based on a predefined classification threshold (90% similar to the modeling class, Line 5 of Algorithm 1), the classifier will provide a set of classification results for all currently unknown domain names (Line 7). If any group is classified as disposable, nodes in the group are decolored in the tree (Line 9 to 11), and the disposable zone for the group is sent for output (Line 12). Depending on the classification results of each group, the Algorithm 1 will either stop (Line 1 to 3) or recursively

continue to search for disposable zones (Line 15 to 17).

Let the classifier be $\mathcal{C}(G_k) = (p, class)$, where p is the probability of G_k that belongs to $class$. For our training dataset, we use zones manually verified to facilitate disposable and non-disposable domain names. The disposable class contains 398 zones, and the non-disposable class includes 401 2LD zones, as discussed in Section IV-B. The training dataset for our classifier contains a small set of zones in disposable class, which might cause the classifier to be biased; however, we should note that this is the first time that anyone has labeled zones as disposable. Thus, we had to manually label every single zone in the disposable class by inspecting thousands even millions of domain names under each zone. The label in the classification process could be “disposable” or “negative” and it will be accompanied by a confidence score between zero and one. For example, if the label is “disposable” with confidence close to one, this means that domains under the zone with the same depth k are likely to be disposable. Then, we go through all the sub-zones under the inspection zone in the same way, excluding the nodes deemed as “disposable”, and see if there exists a sub-zone used for disposable domains.

Algorithm 1 shows the exact steps of the disposable domain name mining process. Using the example domains from Figure 8 as context, the input to Algorithm 1 is `example.com`. We differentiate the nodes as black or white nodes as we discussed in Section V-A1 and we proceed with the feature computation process. At this point for zone `example.com` we have G_3, G_4, G_5 sets and the corresponding statistical vectors. We classify them against an already trained model and we receive the confidence and class for each vector, i.e., each set G_k . Assuming G_3 is classified as disposable with a confidence over 0.9, `a.example.com, c.example.com` are decolored in the domain name tree, yielding the tree in Figure 9, and the algorithm outputs pair `(example.com, 3)`. Next, Algorithm 1 is run recursively for all child nodes of `example.com`, i.e., `a.example.com, b.example.com, c.example.com`. In the case of `c.example.com`, the recursion would stop since there are no black descendants remaining. For `a.example.com`, child nodes of a disposable zone can be either disposable or non-disposable, depending on the classification results.

C. Results

Our measurement results are summarized in Figure 11, and we will describe the results in detail in this section.

Using traditional model selection methods [24] over the training dataset, we chose LAD decision tree³ as the disposable domain name classifier \mathcal{C} . The classifiers we used in our model selection process in addition to LAD were Naive Bayes, Nearest Neighbors, Neural Networks

³We omit details on the classification accuracy from each classifier used during the model selection in the interest of space.

Category	Results
Classifier Accuracy	97% True Positive Rate 1% False Positive Rate
Number of 2LDs with Disposable Zones	12,397
Number of Disposable Zones	14,488
Industries that use Disposable Domains	Popular Websites, Anti-Virus Companies, DNSBLs, Social Networks, Streaming Services, P2P Services, Cookie Tracking Services, Ad Networks, E-commerce, etc.
Labeled Example	Google, Microsoft, McAfee, Sophos, Sonicwall, Facebook, Myspace, Netflix, Paypal
Newly Found Example	Spamhaus, Mailshell, Photobucket, Quora, Skype, Esomnitore, AdSense, Bluelink Marketing, ClickBank, 207.net
% of Disposable Domains/Queried Domains	Increased from 23.1% to 27.6%
% of Disposable Domains/Resolved Domains	Increased from 27.6% to 37.2%
% of Disposable RRs/All RRs	Increased from 38.3% to 65.5%

Figure 11: Table of measurement results summary.

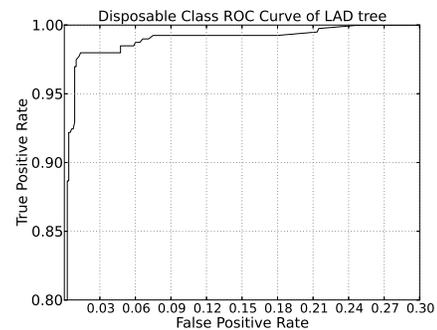


Figure 12: ROC Curve of selected model LAD tree.

and Logistic Regression. To evaluate the accuracy of the classifier, we used the standard 10-fold cross validation methodology [24] on the training dataset. Figure 12 demonstrates the ROC curve of the disposable class for the LAD tree model. Using $\theta = 0.9$ as our threshold, we obtain a true positive rate of 92.4% and a very low false positive rate of 0.6%. If we use the default threshold of $\theta = 0.5$, we have a 1% false positive rate and a 97% true positive rate.

The disposable zone miner was run over 6 days worth of data from one recursive DNS cluster at the North American ISP. Using the fpDNS datasets from these 6 days⁴, we obtain classification results over the unknown portion of the dataset. Over the 6 day period, we found 14,488 zones that use disposable domains, which are under 12,397 unique 2LDs, with a confidence of more than 90%. On average, there are 7 periods in disposable domains, indicating that disposable domains tend to be longer than normal domain names.

1) *Prevalence*: Disposable domains are widely used by various industries, including popular websites (e.g., Google, Microsoft), Anti-Virus companies (e.g., McAfee, Sophos, Sonicwall, Mailshell), DNSBLs (e.g., Spamhaus, countries.nerd.dk), social networks (e.g., Facebook, Myspace), streaming services (e.g., Netflix), P2P services (e.g., Skype), cookie tracking services (e.g., Esomnitore,

⁴02/01, 09/02, 09/13, 11/14, 11/29 and 12/30.

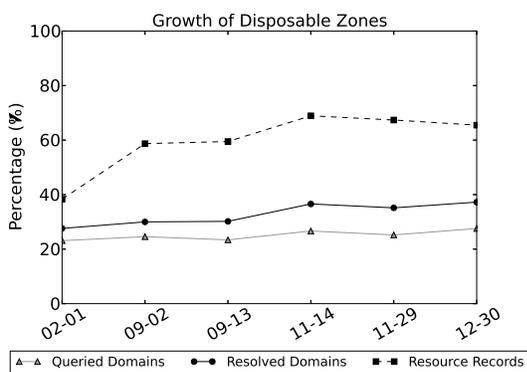


Figure 13: Growth of disposable zones.

2o7.net), ad networks (e.g., AdSense, Bluelink Marketing), e-commerce business (e.g., Paypal, ClickBank), etc. Figure 11 illustrates some examples of labeled disposable zones and newly found disposable zones.

Of the 14,488 disposable zones, we verified that 91 (0.6%) of them were related to content delivery networks (CDNs). We used a customized list containing 451 CDN 2LDs to analyze the result and found that these 91 zones are under 24 (5.3%) 2LDs. It is probably because of some extremely unpopular content being served under specific CDN sub-zones, making the domains appear as disposable in our network. These could be false positives or a result of different level of services provided by CDNs. Since only a small percentage (0.6%) of disposable zones are CDN zones, there is a new class of (disposable) domain names that should be clearly differentiated from CDN related traffic.

2) *Growth*: Disposable domains are not only widely used currently, but are also increasingly being used. Figure 13 shows that for unique domains seen in daily traffic below the recursives the percentage of disposable domains increased from 23.1% to 27.6%. Also, of the daily resolved unique domains the percentage of disposable domains grew from 27.6% to 37.2% over the year of 2011. From traffic during 11/28/2011 to 12/10/2011, we observe that the number of new disposable domains seen every day is always high, around 5 million to 7 million. However, the number of new non-disposable domains dropped from 13 million to 1.6 million. So after one day, more than 50% of new domains seen daily are disposable, and after

Date	Volume < 10	disposable tail	% of all disposable
02/01/2011	90.09%	28.34%	95.95%
09/02/2011	92.77%	50.60%	96.89%
09/13/2011	93.14%	51.21%	97.50%
11/14/2011	94.01%	59.36%	97.80%
11/29/2011	93.83%	57.34%	97.60%
12/30/2011	93.54%	57.17%	98.50%

Table I: Disposable RRs in low lookup volume tail.

13 days, more than 80% of new domains seen daily are disposable, since new disposable domains are constantly

Date	zero DHR	disposable tail	% of all disposable
02/01/2011	88.62%	28.38%	94.48%
09/02/2011	91.59%	50.54%	95.33%
09/13/2011	92.62%	50.93%	96.28%
11/14/2011	93.50%	59.12%	96.73%
11/29/2011	93.02%	57.21%	96.36%
12/30/2011	92.72%	56.96%	97.15%

Table II: Disposable RRs in zero domain hit rate tail.

generated. Moreover, the volume of unique disposable RRs daily increased from 8,111,274 (02/01/2011) to 29,738,493 (12/30/2011), during which 33,704,127 were observed on 11/14/2011. The percentage of daily unique disposable RRs increased from 38.3% to 65.5% (see Figure 13).

Disposable domains are growing in the DNS long tail as well. Table I shows the long tail from the RR lookup volume. Note that the second column presents the size of the tail of all RRs, the third column presents the disposable part of the tail, and the last column presents the fraction of disposable RRs that are in the tail. The disposable RRs represent 28% of the tail on 02/01/2011, and increased to 57% of the entire tail on 12/30/2011. As we observe, between 96% to 98% of all disposable RRs are in the tail. On the other hand, in Table II we can see the statistics of long tail in the domain hit rate distribution of resource records. Around 96% of disposable RRs belong to the tail, and the percentage of domains in the long tail that are also disposable RRs increased from 28% to 57% during 2011. To summarize, disposable RRs are usually present in the DNS long tail and the DNS long tail is increasingly composed of disposable RRs. In the following section, we discuss their potential impact from the DNS operation point of view.

VI. DISCUSSION

In Section V-C, we showed that disposable domains make up about 25% of all unique queried domains, and 27% to 37% of all successfully resolved domains daily. In addition, the number of distinct RRs related to disposable domains represent an average of 60% of all distinct RRs observed in a single day. Also, we offered evidence showing that disposable domains are used by large content providers (e.g., Facebook and Google). In this section, we discuss possible negative effects of the continued growth in the use of disposable domains, and their impact on modern DNS operations and DNS-related systems. Our main objective is to identify and highlight some of these possible effects, so that the operational community can anticipate them and plan ahead in cases where changes to current DNS operations are needed.

A. DNS Caching

In Section IV-B, we showed that disposable RRs are characterized by very low or zero cache hit rates. This is a natural consequence of the “one time use” pattern typical of this new class of domains. As the use of disposable domains increases, the DNS cache may start to be filled with entries

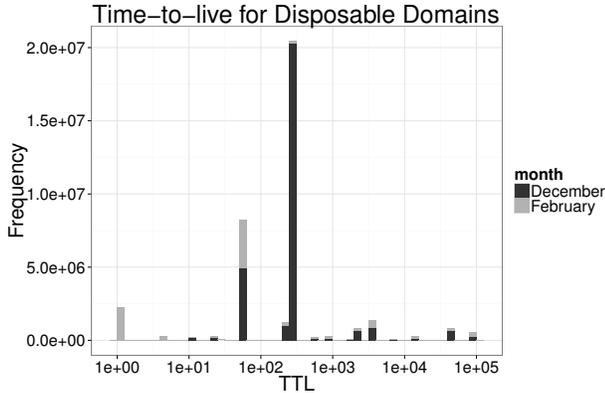


Figure 14: Time-to-live value histogram for disposable domains in February and December, 2011. TTL values range from 0s to 86400s, and values bigger than 86400s are plotted as 86400s.

that are highly unlikely to ever be reused. Assuming a typical Least Recently Used (LRU) cache implementation with a fixed memory allocation (a common configuration in DNS resolvers, to the best of our knowledge), during periods of heavy load (see Figure 2) queries to disposable domains may cause some useful cached non-disposable domains to be prematurely evicted to make room for them. In turn, this may have the effect of inflating the traffic between the DNS resolvers and the authoritative name servers responsible for the evicted non-disposable domains, thus increasing the response latency. If this occurs frequently, caching policies may require adjustments to mitigate the performance decrease, e.g., disposable domains could be treated with low priority.

Forcing disposable domains to use a time-to-live value (TTL) equal to zero is not a feasible solution. First, it may not be feasible to force all the domain owners to set the TTL of disposable domains to zero, since they can freely choose the TTL value they prefer. Figure 14 shows the TTL distribution for disposable domains on 02/01/2011 and 12/30/2011. Note that X axis is log scale and starts from zero. There were 0.8% of disposable domains with a TTL of zero, and 28% of them with TTL = 1 second on 02/01/2011. However, domain owners switched to using relatively larger TTL values over time. For instance, in December, most disposable domains had a TTL of 300s, as we can see from the highest bar in Figure 14. In addition, some recursive DNS software implementations hold resource records into the cache for a minimum number of seconds, even when their TTL is set to zero [25], [26].

B. DNSSEC-Enabled Resolvers

Once DNSSEC is widely deployed, or even under DLV signed zones, eventually every domain name under a zone needs to be signed. There will inevitably be more pressure on *validating resolvers*, which will consume more resources. Clearly, validating signed responses will require higher

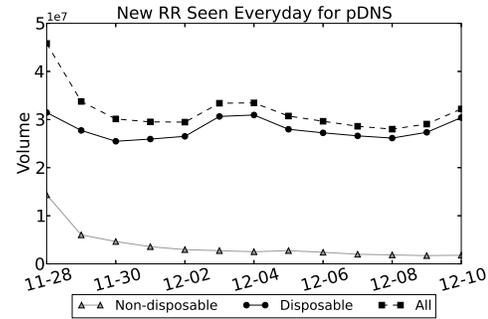


Figure 15: New Resource Records over 13 days.

CPU usage, and increased memory needs due to the larger resource records introduced by DNSSEC specifications (e.g., DNSKEY, DS, RRSIG [27], [28], [29]). Disposable domains will naturally, and potentially dramatically, increase this pressure on validating resolvers. In fact, each queried disposable domain may require an additional signature validation whose result will never be reused. Also, the cache must store not only the disposable RRs, but also their signatures. This problem may be mitigated in part if the authoritative servers responsible for the disposable zones register disposable domains under a single signed wildcard domain, from which the disposable domains are synthesized.

C. Passive DNS Databases

Passive DNS database systems (pDNS-DBs) have recently been adopted by the computer security and networking communities as an invaluable tool to analyze security incidents and assist DNS operations [14], [13], [30]. For example, pDNS-DBs have been extensively used to investigate Operation Aurora [9], attacks to EMC/RSA [10], and malware infections of Stuxnet [11] and Flame [12]. Because these types of security incidents are often discovered months or even years after the attacks first occurred [9], pDNS-DBs play a vital role to efficiently archive long-term historic DNS information. Furthermore, pDNS-DBs are indispensable when constructing dynamic reputation systems [6], [7], [8] for domain names.

Disposable domains have the effect of increasing pDNS-DB storage requirement and potentially the query-response latency, depending on the implementation. In fact, we found that after bootstrapping a pDNS-DB with over 13 days of resolution traffic (see Figure 15), 88% of all unique resource records in the database are disposable, which need to be stored to maintain a full account of historic DNS resolutions. Moreover, the percentage of new RRs related to disposable domains increased from 68% to 94% daily. The problem can be mitigated by filtering disposable domains and storing a single *wildcard* domain in the pDNS-DB. For example, a domain name like 1022vr5.dns.xx.fbcdn.net can be replaced by *.dns.xx.fbcdn.net. Using *wildcard* in the scheme would reduce 129,674,213 distinct disposable resource records we have seen to 945,065 (0.7%) resource records.

VII. CONCLUSION

With this paper we describe and build a disposable zone miner to automatically find disposable domain names. Using traffic from a large ISP in North America, we identified and measured a new category of DNS traffic, the disposable domain, which currently is “lost” in the DNS noise. We show that, on average, disposable domain names are responsible for a significant portion of all domain names observed (25%) and resolved (32%), 60% of unique resource records observed daily, and 88% of all unique resource records observed during our 13 day experiments. Furthermore, we discussed their potential implication to DNS caches, to the DNSSEC deployment and passive DNS data collection systems.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments, and our shepherd Dr. Angelos Stavrou. This material is based upon work supported in part by the National Science Foundation under Grants No. CNS-1017265, CNS-0831300, and CNS-1149051, by the Office of Naval Research under Grant No. N000140911042, and by the Department of Homeland Security under contract No. N66001-12-C-0133. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Office of Naval Research, or the Department of Homeland Security.

REFERENCES

- [1] Z. M. Mao, C. D. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang, “A precise and efficient evaluation of the proximity between web clients and their local dns servers,” in *Proceedings of the General Track of USENIX ATEC*, 2002.
- [2] N. Weaver, C. Kreibich, and V. Paxson, “Redirecting DNS for Ads and Profit,” in *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2011.
- [3] S. Krishnan and F. Monrose, “DNS prefetching and its privacy implications: when good things go bad,” in *Proceedings of USENIX Workshop on LEET*, 2010.
- [4] S. H. Gunderson, “Global IPv6 statistics: Measuring the current state of IPv6 for ordinary users,” in *Proceedings of the Seventy-third Internet Engineering Task Force*, 2008.
- [5] McAfee, “Faq for global threat intelligence file reputation,” <https://kc.mcafee.com/corporate/index?page=content&id=KB53735>, 2013.
- [6] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, “Building a Dynamic Reputation System for DNS,” in *Proceedings of USENIX Security Symposium*, 2010.
- [7] M. Antonakakis, R. Perdisci, W. Lee, D. Dagon, and N. Vasiloglou, “Detecting Malware Domains at the Upper DNS Hierarchy,” in *Proceedings of USENIX Security Symposium*, 2011.
- [8] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, “Exposure: Finding malicious domains using passive dns analysis,” in *Proceedings of NDSS*, 2011.
- [9] M. Antonakakis, C. Elisan, D. Dagon., G. Ollmann, and E. Wu., “The command structure of the Aurora botnet,” https://www.damballa.com/downloads/r_pubs/Aurora_Botnet_Command_Structure.pdf, 2010.
- [10] U. Rivner, “Anatomy of an attack,” <http://blogs.rsa.com/anatomy-of-an-attack/>, 2011.
- [11] N. Falliere, L. O. Murchu, and E. Chien, “W32.stuxnet dossier,” http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf, 2011.
- [12] Global Research & Analysis Team (GReAT) Kaspersky Lab, “Full analysis of flame’s command & control servers,” http://www.securelist.com/en/blog/750/Full_Analysis_of_Flames_Command_Control_servers, 2012.
- [13] F. Weimer, “Passive dns replication,” in *17th Annual FIRST Conference*, 2005.
- [14] D. Plonka and P. Barford, “Context-aware clustering of dns query traffic,” in *Proceedings of ACM SIGCOMM conference on Internet measurement*, 2008.
- [15] S. Souders, “Sharding dominant domains,” <http://www.stevesouders.com/blog/2009/05/12/sharding-dominant-domains/>, 2009.
- [16] P. Vixie, “What dns is not,” *Queue*, no. 10, Nov. 2009.
- [17] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan, “Detecting algorithmically generated malicious domain names,” in *Proceedings of ACM SIGCOMM conference on Internet measurement*, 2010.
- [18] A. Berger and E. Natale, “Assessing the Real-World Dynamics of DNS,” in *Traffic Monitoring and Analysis*, ser. Lecture Notes in Computer Science, A. Pescapè, L. Salgarelli, and X. Dimitropoulos, Eds. Springer Berlin Heidelberg, 2012, vol. 7189, pp. 1–14.
- [19] V. Paxson, M. Christodorescu, M. Javed, J. Rao, R. Sailer, D. Schales, M. P. Stoecklin, K. Thomas, W. Venema, and N. Weaver, “Practical comprehensive bounds on surreptitious communication over dns,” in *Proceedings of USENIX Security Symposium*, 2013.
- [20] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, “Dns performance and the effectiveness of caching,” *IEEE/ACM Trans. Netw.*, 2002.
- [21] J. Jung, A. Berger, and H. Balakrishnan, “Modeling TTL-based internet caches,” in *Proceedings of INFOCOM*, 2003.
- [22] Mozilla Foundation, “Public suffix list,” <http://publicsuffix.org/>.
- [23] M. Andrews, “Negative caching of dns queries (dns ncache),” <http://www.ietf.org/rfc/rfc2308.txt>, March 1998.
- [24] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. Wiley-Interscience, 2000.
- [25] A. Kumar, J. Postel, C. Neuman, P. Danzig, and S. Miller, “Common DNS Implementation Errors and Suggested Fixes,” <http://www.ietf.org/rfc/rfc1536.txt>, October 1993.
- [26] D. Barr, “Common dns operational and configuration errors,” <http://www.ietf.org/rfc/rfc1912.txt>, February 1996.
- [27] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, “Dns security introduction and requirements,” <http://www.ietf.org/rfc/rfc4033.txt>, March 2005.
- [28] —, “Resource records for the dns security extensions,” <http://www.ietf.org/rfc/rfc4034.txt>, March 2005.
- [29] —, “Protocol modifications for the dns security extensions, rfc 4035,” <http://www.ietf.org/rfc/rfc4035.txt>, March 2005.
- [30] I. S. Consortium. (2004) SIE@ISC : Security Information Exchange. <https://sie.isc.org/>.