# Open-source Measurement of Fast-flux Networks While Considering Domain-name Parking

Leigh B. Metcalf
*Software Engineering Institute*
*Carnegie Mellon University*

Dan Ruef
*Software Engineering Institute*
*Carnegie Mellon University*

Jonathan M. Spring
*Software Engineering Institute*
*Carnegie Mellon University*

## Abstract

*Background:* Fast-flux is a technique malicious actors use for resilient malware communications. In this paper, domain parking is the practice of assigning a nonsense location to an unused fully-qualified domain name (FQDN) to keep it ready for "live" use. Many papers use "parking" to mean typosquatting for ad revenue. However, we use the original meaning, which was relevant because it is a potentially confounding behavior for detection of fast-flux. Internet-wide fast-flux networks and the extent to which domain parking confounds fast-flux detection have not been publicly measured at scale.

*Aim:* Demonstrate a repeatable method for open-source measurement of fast-flux and domain parking, and measure representative trends over 5 years.

*Method:* Our data source is a large passive-DNS collection. We use an open-source implementation that identifies suspicious associations between FQDNs, IP addresses, and ASNs as graphs. We detect parking via a simple time-series of whether a FQDN advertises itself on IETF-reserved private IP space and public IP space alternately. Whitelisting domains that use private IP space for encoding non-DNS responses (e.g. blacklist distributors) is necessary.

*Results:* Fast-flux is common; usual daily values are 10M IP addresses and 20M FQDNs. Domain parking, in our sense, is uncommon (94,000 unique FQDNs total) and does not interfere with fast-flux detection. Our open-source tool works well at internet-scale.

*Discussion:* Real-time detection of fast-flux networks could help defenders better interrupt them. With our implementation, a resolver could potentially block name resolutions that would add to a known flux network if completed, preventing even the first connection. Parking is a poor indicator of malicious activity.

## 1 Introduction

Fast-flux service networks were first reported in 2007, identified as "a network of compromised computer systems with public DNS records that are constantly changing, in some cases every few minutes" [25, §1]. Criminals use the technique to "evade identification and to frustrate law enforcement and anti-crime efforts aimed at locating and shutting down web sites" that are used for abuse or illegal purposes [13, p. 2].

Despite this long history, and a variety of publications on detecting fast flux, there is no maintained, open-source tool that can detect it at scale. The Honeynet Project's own tool for the purpose, Tracker (`http://honeynet.org/project/Tracker`), has a defunct homepage. Tools from the time, such as ATLAS [20] and FluXOR [21], handle on the order of 400 domains. Our tool, Analysis Pipeline, handles networks on the order of 1 million fully-qualified domain names (FQDN, hereafter simply "domain" if the usage is unambiguous). Pipeline simultaneously tracks other network behavior, such as network flow records. Therefore Pipeline can detect when a host connects to an IP address in the fast-flux network in near-real time, for example.

We also measure a phenomenon mentioned but not measured in some older fast-flux detection papers: domain name parking. When a domain is *parked* on an IP address, the IP address to which the domain resolves is inactive or otherwise not controlled by the domain owner. Parking is common practice when a user first registers an effective second-level domain (eSLD) – the registrar supplies a nonsense IP address to prevents DNS errors. However, this parking pattern is distinctive and simple. We look for other, suspicious patterns.

There are multiple distinct senses of the term "domain parking," and our topic is not synonymous

with any other study of which we are aware. Domain parking on private IP address space is, however, a relatively old phenomenon; it is mentioned in some fast-flux identification algorithm studies as an obstacle [35, 14]. This older usage of "domain parking" is our topic of study We use *domain parking on private IP address space* to differentiate it from the newer usage [2, 34] that more accurately is *domain parking on routeable IP addresses for advertisement revenue generation*.

Two of the first studies on parking domains for illicit ad revenue find large-scale use of 4 million to 8 million domains [2, 34]. However, from the authors' description this appears to be more like typosquatting (as described in Szurdi et. al. [31]) than resolution error suppression. We are not studying typosquatting or skimming ad revenue off user typos. Domain parking of the sort we study is a strategy for suppressing domain resolution errors, likely used to keep command and control infrastructure stealthy.

The domain name system permits a variety of different resiliency mechanisms for distributed architectures. Often these have legitimate uses, but malicious actors are equally able to adopt successful techniques. Fast-flux and domain parking of private IP space are both candidates for such abuse. ICANN responded to the security concerns from fast-flux networks in 2009 by stating there would be no policy response [15, p. 10]. Thus, of the six mitigation options outlined in the original Honeynet analysis, five are unlikely or inconsistently applied because they can only be enacted by ISPs or Registrars. The last, "passive DNS harvesting/monitoring to identify A or NS records advertised" as part of fast-flux networks [25, §10], is the approach we implement with Analysis Pipeline. Our detection method is inspired by the Mannheim score [12].

We find a mixture of positive and negative results. On one hand, our measurement of fast-flux networks confirms our hypothesis that this behavior remained prevalent. As one negative example, domain parking on private IP address space is not worth much concern, for fast-flux network detection or otherwise. Negative results are important and useful in shaping future work. Publication bias has been a documented concern in medical literature for 30 years [8]. Despite this attention, publication of negative results has generally dwindled across disciplines. The relative publication frequency of positive results over negative results grew by 22% from 1990 to 2007 [9]. We expect such publication bias away from negative results is a contributing factor to why there seems to be no public measurement of this phenomenon.

Section 2 discusses the common elements of the method between our parking and fast flux measurements, which primarily is the passive DNS data source and our open-source tool Analysis Pipeline. We present our measurement method for FQDNs that exhibit parking on private IP address space in Section 2.1. Section 2.2 describes our fast-flux measurement methods. Section 3 presents the full results. Section 4 interprets and discusses these results.

## 2 Method

Our measurements of domain parking of private IP address space and fast-flux networks use different algorithms over the same data set and implemented using the same open-source tool. We describe the common measurement period, data, and tool here. The methods specific to each parking and fast-flux measurement are described in the following subsections.

We measure activity during over five years of passive DNS data, from January 1, 2012 to June 30, 2017. The data source, the Security Information Exchange (SIE), has been demonstrated to be reasonably representative of the global Internet with a small North American collection bias [29]. This is high-volume passive DNS data, as well as being representative. Each month, the unique FQDNs observed range between 550 million and 1 billion. With the relatively small and stable zone `.edu`, the data source is sufficiently represetative to reconstruct 93% of the zone in five weeks [27].

We use our own data filtering and packing tools, independent from the SIE database. About 35-40 GB of data is ingested daily in compressed `nmsgtool` format [10], including source DNS server and precise time range the response was valid. Unique resource record sets (RRsets) are extracted for each 24-hour day, with a cutoff of 0000 UTC. We store just the fields for `rname`, `TTL`, `type`, and `rdata`. The nmsg data canonicalizes `rdata`, so this field is sorted set of all `rdata` in a single DNS message for a `rname,type,class` triple. The `rname` field is label-wise reversed, so www.example.com becomes com.example.www; this makes sorting and lookup easier, as the TLD is usually a more important key. The RRsets are then simply sorted and unique RRsets stored per day. When compressed with standard tools such as bzip or gzip, this ASCII storage format takes about 5 GB per day.

The rationale for this storage method is similar to that for why SiLK, a netflow analysis tool suite, stores flow in time-sorted, partitioned flat files via the file system rather than in a database [32]. We are

interested in long trends, or retrospective analysis of poorly understood past events. This is different from the use case of many passive DNS users, who are looking for keywords indicating abuse of particular brand names. Our storage format provides details on what domains resolved to on particular days. This allows us to see interleaving changes in domain-IP mappings and large gaps of inactivity that are not possible in a database that only stores first- and last-seen times. Our parking measurement, in particular, requires such granularity.

A final benefit of this time-partitioned storage format is that it is highly parallelizable. Using an HDFS cluster, queries parallelize naturally with each node processing a day. We find that using a database can speed our analysis, especially of fast-flux. However, we parallelize even the database, loading each daily RRset file and then operating over it independently. The overhead of managing a database for all five years of data is superfluous.

Both parking and fast-flux measurements make use of context data to enrich IP addresses. Most importantly, we associate IP addresses with the autonomous system which is advertising it on the relevant day. Autonomous System Number (ASN) attribution is derived from the RouteViews [23] and RIPE NCC RIS [22] data. All ASN data are freely available online [3] under folders for the respective dates. The baseline mapping of ASNs across all IP space uses the open-source SiLK [5] tools for prefix maps and IP sets [32]. Strictly, we count unique routing profiles, not unique ASNs. If an IP address is dual-homed or the global BGP otherwise has consensus that two last-hop ASNs are viable, we mark that IP address with both ASNs. Since our goal is to identify when IP addresses are routed differently to identify stewardship changes, this interpretation is sensible. The geolocation data we use is the public MaxMind GeoLite2 [18].

We relate our analysis algorithms as Analysis Pipeline [24] configuration files. Pipeline is one of the open-source tools associated with SiLK [5]. It is a real-time traffic analyzer that works on IPFIX and network flow records. Pipeline is a Network Behavior Analyzer subtype of an Intrusion Detection System in NIST terminology [26]. As a sort of IDS, Pipeline can run in real time on a network to dynamically detect fast-flux or parking domains and then dynamically add them to a list to watch or block. Thus, with the configurations here and the published SiLK tools, our measurements are readily reproducible in the sense of Feitelson [11], where to reproduce means in a different setting with similar artifacts.

**Algorithm 1** Analysis Pipeline command-line

```
/usr/local/sbin/pipeline  \
        --site-config-file=/usr/local/share/
            silk/silk.conf \
        --alert-log-file=~/AlertLog.txt \
        --aux-alert-file=~/AuxLog.txt \
        --ipfix \
        --time-is-clock \
        --configuration=~/parking.conf \
        --name-files input_files_list
```

Algorithm 1 is an example of how to execute Pipeline. Specifically it calls our parking measurement configuration, listed later in Algorithm 2. It reads DNS records encoded in the standard IPFIX format [6]. A sample python script to convert DNS records from CSV format to IPFIX is available in the pyfixbuf documentation [4]. Passive DNS data can also be converted to CSV or IPFIX directly using the nmsg python bindings [10].

In addition to rote results, we perform some simple summary and context operations. The main summary is based on the effective second level domain (eSLD) of the parked domains. The eSLD of `www.example.com` is simply the SLD `example.com`; however, the eSLD of `www.example.co.uk` includes a third label: `example.co.uk`. To identify eSLDs we use the Mozilla public suffix list (effective_tld_names.dat).

The main context-enrichment operation is to intersect parked domains and their publicly-routable IP addresses with fast-flux domains and IP addresses. The intersection is simple set intersection on the domain names. We do not report time slices of the intersection, simply the intersection between the union of all domains exhibiting parking behavior and the union of all fast-flux domains. We also summarize to eSLD and repeat the intersection.

For some context about malicious intent of parking and fast flux, we associate each set of domains with lists of malicious domains. While we have expressed our doubts about the soundness of evaluating an approach by comparing it to blacklists [19], we have mitigated this error by including as many lists as possible (over 100) and limiting our assumptions of the information provided by this comparison.

We perform blacklist comparisons within 6-month blocks. All blacklist entries over the whole timespan are unioned, and that set is intersected with all domains exhibiting the behavior of interset at any time during the timespan. This mitigates the possibility that it takes some time to identify and blacklist a malicious domain. It has proven logistically impractical to provide a sliding window for blacklist de-

tection. But 6-month windows are pretty broad, as many malicious behaviors are consistently detected and vendors list names within a few hours. At this wide window, we already risk false-positives due to IP-address churn or other exogenous factors overwhelming true blacklist associations. We happen to have more IP-address based blacklists than domain-based ones; it is unclear whether this overestimates IP-address participation in blacklists or underestimates domain names, if either.

## 2.1 Parking Detection

We measure *domain parking on private IP address space* straightforwardly. The algorithm is summarized as follows; we expand the description through the rest of the section. First, we find all DNS IPv4-answer RRsets that refer to private address space to acquire a set of possible domains. We remove whitelisted domains that we have found to use private address space as an information carrier for other services. For all domains that remain after this subtraction, we find all their DNS RRsets for that day and a window three weeks into the future. These three-week time series are examined for transitions from public to private IP-space. We repeat this algorithm for each day in the five-year measurement period, evaluating 1807 three-week windows.

Our first step is to extract or mark all RRsets that contain a private IP address in the `rdata`. Private IP address space is exactly those addresses listed in Table 1. These blocks are selected because they are special-purpose assignments that RFC 6890 lists as either not forwardable by routers or not global, meaning only forwardable in specified administrative zones [7].

This provides a set of `rname` data that have been associated with a private IP address. Most are not parking. Private IP space used to encode various kinds of non-location data, such as responses to lookups on DNSBLs [17]. SURBL provides a good example of how and why their service does this [30]. Seeded by lists of threat intelligence and blacklist providers such as `intel.criticalstack.com`, and refined through human expert analysis, we whitelist 165 DNS zones that consistently encode non-location data.

The process so far yields a list of RRsets with `rdata` in private IP space whose `rname` zones do not have a whitelisted, known use. We next find all RRsets with the same `rname` values and publicly routeable IP addresses within 21 days. These domains transitioned between private and routeable IP address space some time in the 3-week window.

| CIDR block | Justification |
|---|---|
| 0.0.0.0/8 | RFC 1122 |
| 10.0.0.0/8 | RFC 1918 |
| 100.64.0.0/10 | RFC 6598 |
| 127.0.0.0/8 | RFC 1700 |
| 169.254.0.0/16 | RFC 3927 |
| 172.16.0.0/12 | RFC 1918 |
| 192.0.0.0/24 | RFC 6890 |
| 192.0.2.0/24 | RFC 5737 |
| 192.168.0.0/16 | RFC 1918 |
| 198.18.0.0/15 | RFC 2544 |
| 198.51.100.0/24 | RFC 5737 |
| 203.0.113.0/24 | RFC 5737 |
| 224.0.0.0/3 | RFC 1112 |

Table 1: Private IP address space

We define FQDNs that exhibit such a transition as demonstrating *parking behavior* on private IP address space during our observation period.

In order to cover the 5-year time period, we compute this rather simple algorithm over 1800 times. For each day's set of unique RRsets, we extract the domains mapping to private IP address space, remove whitelisted zones, and expand to those also mapping to a public address at some time within three weeks.

For each FQDN that has exhibited parking behavior, we can generate a course-grained time series of the behavior to categorize what occurred. Table 2 demonstrates some sample behavioral groupings. P indicates a day where the only `rdata` was in private IP address space, G indicates a day where the only `rdata` was in globally routeable IP address space, and X indicates a day where both address types were observed, indicating a day a change between parking and active occurred.

The configuration listed in Algorithm 2 runs our method in Pipeline. The SiLK IPset "priv.set" contains exactly the address blocks listed in Table 1.

## 2.2 Method: Fast-flux

Our fast-flux detection algorithm implements prior work, such as the Mannheim score [12]. The main novelty of our work is the scale and duration of our measurement and the use of open-source tools. Detection algorithms were sufficiently well-studied in 2010, and the same concept holds for detection of fast-flux today.

The basis of our fast-flux detection algorithm is that a legitimate administrator owns or rents their infrastructure in a relatively small number of places.

| January: | 1-8 | 9-16 | 17-24 | 25-31 |
|---|---|---|---|---|
| Activation on Jan 19 | PPPPPPPP | PPPPPPPP | PPXGGGGG | GGGGGGG |
| Deactivation on Jan 19 | GGGGGGGG | GGGGGGGG | GGXPPPPP | PPPPPPP |
| alextringham.com | GGGGGGGG | GGGGGGGG | GGGGXPPX | PXPXPPP |
| proxyie.cn | GGXXXXXX | GGGXGXGG | GGPGGXGX | XGGGXGX |
| bnlv.homeip.net | GGGGGPGG | GGGGGGGG | PGPPGGGG | GGGGPGG |

Table 2: Example parking behavior patterns per domain, January 2014. G := only globally routeable IPs observed that day. P := only privately reserved IPs observed. X := both observed on same day.

**Algorithm 2** Parking detection in Pipeline

```
FILTER emptyDomainNames
  dnsRRIPv4Address IN LIST "priv.set"
END FILTER

INTERNAL FILTER emptyDomains
  FILTER emptyDomainNames
  dnsQName domainsWithNoIP 1 DAY
END INTERNAL FILTER

FILTER unparked
  dnsQName IN LIST domainsWithNoIP
  dnsRRIPv4Address NOT IN LIST "priv.set"
END FILTER

EVALUATION unparkedRecords
  FILTER unparked
  CHECK EVERYTHING PASSES
  END CHECK
  ALERT ALWAYS
  ALERT EVERYTHING
END EVALUATION
```

**Algorithm 3** Fast-flux detection in Pipeline

```
PMAP asn "$today.ip2asn.pmap"
FILTER fluxwhitelist
  sourceIPv4Address NOT IN_LIST "priv.set"
  DNS_SLD+TLD(DNS_INVERT(dnsQName)) NOT
      IN_LIST "$today.whitelist"
END FILTER

EVALUATION ipfixFastFluxExample
  FILTER fluxwhitelist
  CHECK FAST FLUX
    IP_FIELD sourceIPv4Address 500
    ASN asn 23
    DNS dnsQName 667 TO myFastFluxDNSs
    NODE MAXIMUM 100000000
    VERBOSE ALERTS
  END CHECK
  CLEAR ALWAYS
END EVALUATION
```

We perform some pilot studies on January 2016 to get a sense of what counts as small. We represent fast-flux networks as graphs of each of three kinds of resource: IP addresses, ASN, and FQDN. The intuition is that shared hosting may have 10,000 domains on a single IP, and if changes to another IP address it is likely one the hosting provider owns. The network identifiers cluster when one or another resource is advertised on a new resource. If two domains both map to 192.168.0.1, and then one is changed to 10.0.1.1, then our algorithm considers both IP addresses as well as both domains to be part of a cluster. The information cluster would also include any domains that had previously mapped to 10.0.1.1 within the time frame, and any IP addresses they had been mapped to, and so on. For our example, imagine this linking brings in 20 more domains, all on the 16 IP addresses in 172.16.0.32/28. However, the AS for all 18 IP addresses is AS112. We do not consider this a fast-flux network, because all the resources are only related to one AS. They are probably related because of a the AS owner doing regular maintenance or load balancing, not fast flux. This example helps show why ASN is needed, rather than relying on CIDR block.

Our results are run with the conservative thresholds that a graph must contain 500 unique IPs, 23 ASNs, and 667 FQDNs to be marked fast flux. We also whitelist any IP addresses in our private IP address space (Table 1) and a FQDN whitelist that created from the Alexa most popular domains list. Resources on these whitelists will not be added to a graph, and so can never be marked fast flux.

The FQDN whitelist captures more stability than naïvely using the Alexa top X. For a given day, we find all names in the Alexa top 24,000 on 330 of the past 365 days. The whitelist functions as a wildcard, not as an FQDN perfect match. So if example.com is on the whitelist, *.example.com will not be watched for fast flux. For this reason, we remove any effective TLDs on the Mozilla public suffix list (version on June 15, 2017), of which there are routinely between 50-100 on the Alexa list. We also remove any well-known dynamic DNS providers from the whitelist. Finally, we make sure that no domains have subdomains on the list; this interferes with the wildcard function of pipeline whitelists. The aver-

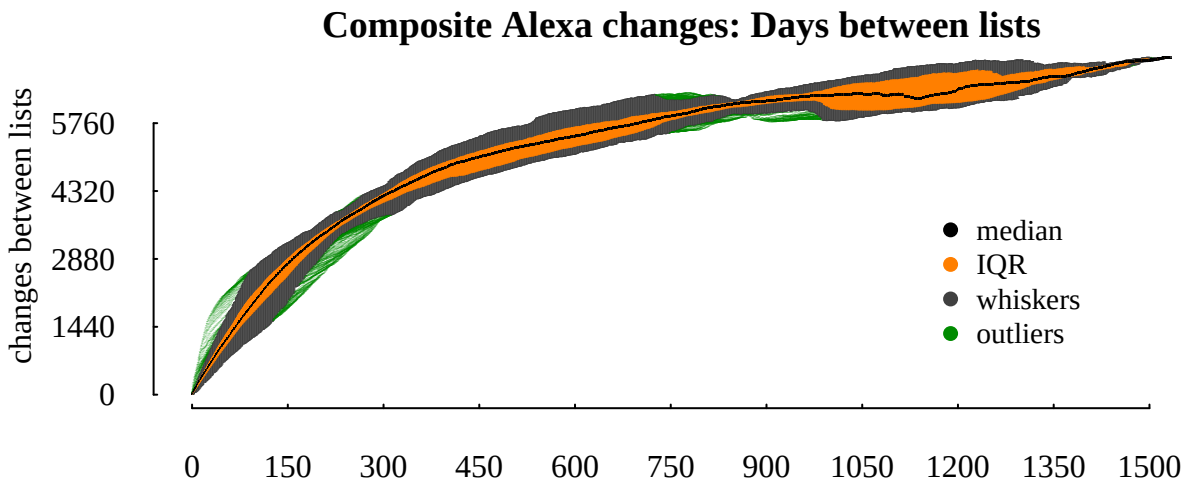


Figure 1: Statistics on changes between Alexa-based whitelists, calculated on the pairwise differences between lists. Changes includes additions and removals. Each value on the X-axis is a color-based box-and-whisker plot representing the distribution of changes for all pairwise sets of lists that many days apart.

age size of the whitelist is 14,847. The minimum is 13,780 (March 7, 2017); the maximum size is 15,946 (April 4, 2013). Even with this algorithm designed for some stability, there significant change, which warrants using a new list each day. Figure 1 displays the distribution of whitelist-entry changes calculated pairwise between all 1492 lists. The first day we have Alexa data available is April 1, 2012, so the first day a composite whitelist is possible is April 1, 2013. We use this April 1, 2013 list for any flux measurements before that date.

We selected a relatively high threshold (500-23-667) after exploratory analysis demonstrated some clearly legitimate uses that exceeded our initial 5-5-5 threshold. Before implementing the Alexa whitelist, we also found Tumblr, as 616 IPs, 10 ASNs, and 10,658,458 FQDNs. But not every large network like this is handled by a whitelist. For example, the network signature 216 IPs, 19 ASNs, 2,341,876 FQDNs is not in fact a fast-flux network, but Tekblue, an ad tracking company. But, because Tekblue is an ad tracker, it does not appear on our Alexa whitelist, on any day. Ampproject.net also consistently produced huge fast-flux-like clusters—such as 755 IPs, 15 ASNs, 533,082 FQDNs and 671 IPs, 12 ASNs, 534,956 FQDNS—but is never on our whitelist. These are evidence for increasing our ASNs threshold. We also find evidence to increase our FQDN threshold. The graph 9,051 IPs, 102 ASNs, 63 FQDNs is Akamai. NTP device pools also produce quite strange signatures, such as 2100 IPs, 845 ASNs, 122 FQDNs and 2,507 IPs, 910 ASNs, 267 FQDNs.

We did a sample run at 500-23-667 to check results. The graph with the fewest ASNs passing these thresholds was 27; its domains often have a suspicious pattern that appears machine-generated. There are some names that look human-generated; however, they may be compromised. At the least, it is not something to obviously exclude by increasing the thresholds. The test run reduced the results from 513 (with 5-5-5 thresholds) to 196 distinct, non-overlapping flux networks. But these 196 still capture almost all of the IP addresses from the 513; 99.7% of the unique IP addresses across flux networks remain in the results with the increased threshold. Therefore, while our thresholds are conservatively high, we still find significant malicious activity while reducing obvious false positives.

One may wonder if publishing such a detection threshold would benefit adversaries more than defenders. However, forcing adversaries to keep smaller, disjoint networks would reduce their reliability and increase their management effort. Adversaries would no longer be able to use any resource if its FQDN or IP is associated with a known, live flux network. Pipeline can issue such alerts in real time, as new resources are seen and added to known networks. Potentially, this means many communications can be blocked at the first instance, preventing even one use of the FQDN or IP if it is added to a known flux network. Such before-first-use blocking is recommended by Spring [28] as necessary to keep adversaries from profiting.

Analysis Pipeline (version 5.0 and later) includes a primitive data element for fast-flux networks [24].

| FQDNs | intersection | of flux | of park |
|---|---|---|---|
| 2012-1 | 14609 | 0.0008 | 0.2069 |
| 2012-2 | 12103 | 0.0009 | 0.0286 |
| 2013-1 | 10930 | 0.0006 | 0.1990 |
| 2013-2 | 11662 | 0.0006 | 0.1126 |
| 2014-1 | 11106 | 0.0006 | 0.1362 |
| 2014-2 | 30346 | 0.0007 | 0.0714 |
| 2015-1 | 61259 | 0.0010 | 0.1756 |
| 2015-2 | 40824 | 0.0008 | 0.0800 |
| 2016-1 | 32687 | 0.0004 | 0.0934 |
| 2016-2 | 46291 | 0.0004 | 0.1125 |
| 2017-1 | 56758 | 0.0006 | 0.0458 |

Table 4: Half-yearly intersections of domains exhibiting parking and fast-flux networks. Values range $[0, 1]$.

| IPs | intersection | of flux | of park |
|---|---|---|---|
| 2012-1 | 523056 | 0.0147 | 0.9127 |
| 2012-2 | 2565808 | 0.0717 | 0.8847 |
| 2013-1 | 220651 | 0.0052 | 0.8934 |
| 2013-2 | 257741 | 0.0083 | 0.8644 |
| 2014-1 | 364870 | 0.0127 | 0.9415 |
| 2014-2 | 730484 | 0.0149 | 0.9421 |
| 2015-1 | 478327 | 0.0086 | 0.9100 |
| 2015-2 | 557457 | 0.0109 | 0.8911 |
| 2016-1 | 1567197 | 0.0268 | 0.9039 |
| 2016-2 | 1562149 | 0.0169 | 0.7736 |
| 2017-1 | 837599 | 0.0141 | 0.7640 |

Table 5: Half-yearly intersections of IP addresses exhibiting parking and fast-flux networks. Values range $[0, 1]$.

Pipeline builds a connected graph of ASN, FQDN, IP address tuples. If the connected graph passes a threshold for all three resources, that graph is considered to be a fast flux network. Algorithm 3 is the Pipeline configuration that implements our detection algorithm. The alerts can be configured to report the whole connected graph, or just lists of domains or IP addresses. Algorithm 4 shows how to use such output lists.

## 3 Results

Fast-flux networks do not overlap much with resources exhibiting parking. Neither fast-flux nor parking overlap much with blacklisted resources. Parking behavior is present, but small in the scheme of the global internet. Fast-flux networks, on the other hand, appear to make use of a large number of internet resources.

Figure 2 plots the number of unique FQDNs and effective SLDs used each day for parking. Our parking algorithm uses a 3-week window to detect parking, which has an effect of smoothing out the day-to-day changes. The median FQDNs parking on a given day is 13,300, in a median of 4,750 eSLDs.

Figure 3 captures the fact that fast-flux networks have a much bigger footprint. Many days have over 10,000,000 IPs and 20,000,000 FQDNs involved in fast-flux.

Table 4 reports the overlap between fast-flux and parking FQDNs detected in each half-year. Table 5 reports the analogous overlap for IP addresses. Since there are fewer resources that evidence parking, these intersections make up a larger share of parking than of fast flux.

Given the conservative (i.e., large) definition of flux network size we set, it is unlikely these collections of internet resources have a benign purpose. Our initial pilot study (using 5-5-5 as a threshold) contained common internet services such as ad networks, content distribution networks, and the NTP servers. The threshold of 500-23-667 excludes such benign services, based on our expert analysis of the results.

Despite the fact we do not have a benign explanation for this behaviour, as Figure 4 and Figure 5 demonstrate, few IP addresses that have participated in a fast-flux network are on any blacklists. The median monthly blacklist intersections range between 100,000 and 400,000; roughly 4-8% of the flux networks. The exception seems to be overlap during 2017 between flux networks and FQDN blacklists.

Parking, likewise, is uncommonly blacklisted. Table 3 displays the results for both FQDNs and live IP addresses associated with parking behavior. Because so few domains park, the contribution of parking to blacklists is negligible. However, parking is also not a reliable indicator of blacklist membership. Fewer than 2% of domains and between 5-10% of IPs that exhibit parking end up on blacklists.

Our results for January 2014 are available for download (see `http://www.cert.org/downloads/name-parking-patterns-certcc-2014-57.txt`) in the format of a domain name followed by the behavior pattern encoded as in Table 2.

FQDNs from the Alexa top 100 occasionally were found on our parking list [1]. We manually removed about 30 Alexa top 100 domains from the results each period. The root cause for these anomalous DNS responses is not known.

| | FQDNs | % lists | % parking | IPs | % lists | % parking |
|---|---|---|---|---|---|---|
| 2012-1 | 3033 | 0.0469 | 4.2959 | 87069 | 0.2079 | 14.1214 |
| 2012-2 | 2513 | 0.0272 | 0.5929 | 85853 | 0.2525 | 2.5345 |
| 2013-1 | 1746 | 0.0139 | 3.1782 | 26479 | 0.0941 | 10.1526 |
| 2013-2 | 4439 | 0.0883 | 4.2842 | 13300 | 0.1473 | 4.1119 |
| 2014-1 | 2005 | 0.0723 | 2.4587 | 33654 | 0.1342 | 8.4516 |
| 2014-2 | 6596 | 0.0631 | 1.5530 | 51061 | 0.1702 | 5.7200 |
| 2015-1 | 5616 | 0.0459 | 1.6101 | 47535 | 0.1565 | 8.6189 |
| 2015-2 | 8339 | 0.0639 | 1.6332 | 49728 | 0.2010 | 7.3737 |
| 2016-1 | 6802 | 0.0279 | 1.9426 | 91716 | 0.2647 | 4.9624 |
| 2016-2 | 4295 | 0.0164 | 1.0442 | 100620 | 0.3417 | 4.5519 |
| 2017-1 | 5315 | 0.0136 | 0.4292 | 64859 | 0.1462 | 10.332 |

Table 3: Half-yearly intersections of resources exhibiting parking and blacklists. Percentages range $[0, 100]$

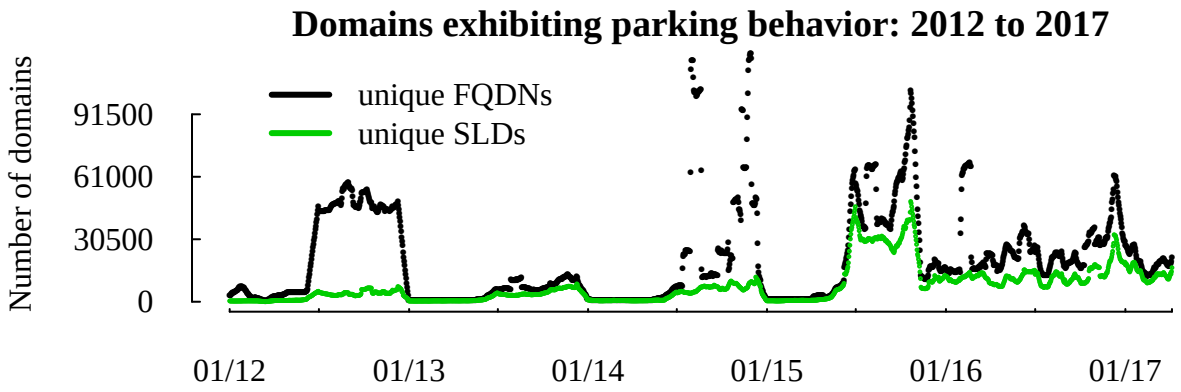


Figure 2: Three-week rolling window of unique domains and eSLDs exhibiting parking behavior from beginning of 2012 to April 2017.

Some parking domains used dynamic DNS services; however, usage is minimal. We compared the results to a list of 71 known dynamic DNS providers. The bulk were hosted on two providers: dyndns.org or on some name affiliated with no-ip. These are the two biggest providers, so this distribution is expected based on market share.

## 4 Conclusions

We shall discuss our fast-flux results first, and then our parking results.

Fast-flux networks remain remarkably common 10 years after first reported use by malicious actors. The lack of intersection with blacklists despite the obviously suspicious nature of this behavior is especially noteworthy. We suspect that fast-flux networks are used for intermediary malicious behavior, such as providing clandestine communication to already infected hosts. It is also possible that blacklist vendors do not bother to list something that they know will change within a very short interval. We also have not ruled out all possible alternative interpretations; for example, peer-to-peer networks. However, if this were the case, we would still expect our flux results to be a superset of malicious flux networks. Excessive poor detection precision would decrease the number of flux resources on blacklists, but it does not explain why so few of our blacklist entries are in flux networks.

We have designed our method to capture non-benign fast-flux networks. Our first attempt at setting thresholds captured many recognizable, legitimate internet services. However, as described in Section 2.2, we eliminated all obvious and large false positives. Our observation is limited by the data source. However, again, this has a known bias, and it is known to be comprehensive. We do not make any claims that we can project our observations onto the parts of the internet we do not observed. However, the absolute numbers of resources participating in fast-flux we detect are quite large. We do not need

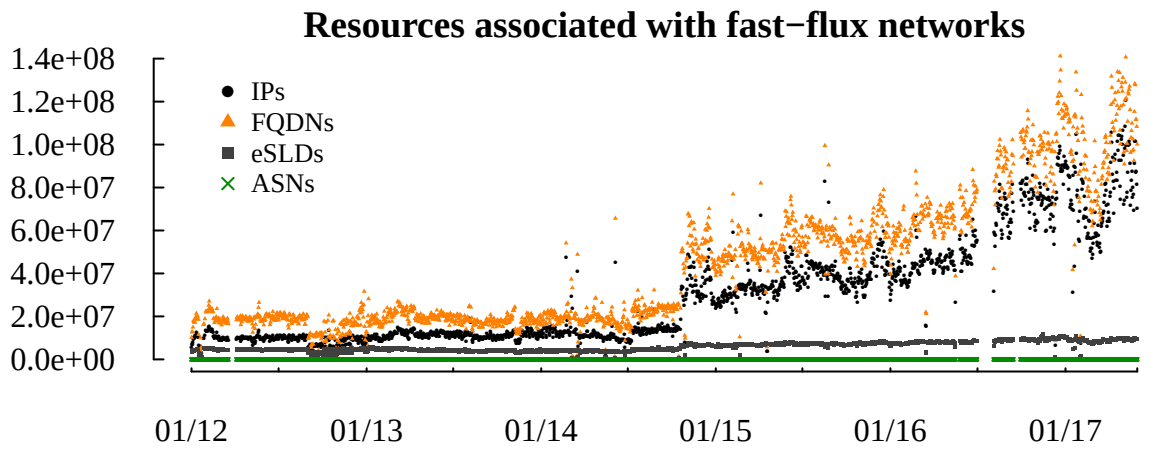


Figure 3: Total unique network resources of different types associated with fast-flux networks every day. Gap in July 2016 is a collection error.

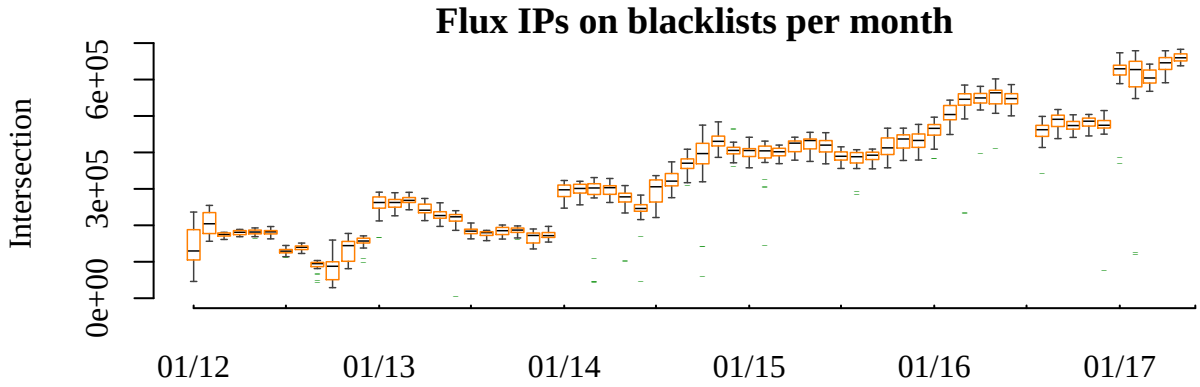


Figure 4: Summary statistics fast-flux–blacklist intersection. Each day's flux from January through June 2013 is intersected with all 1H2013 blacklist data, for example, and the daily intersections are summarized in monthly box plots. The whisker length is 1.5 times the inter-quartile range (IQR).

to project onto a target population. The measurement as-is finds over 100 million IPs and FQDNs some days in 2017. We have high confidence that few of these resources are participating for benign reasons. Even in the unlikely event that the rest of the internet sees no other unique resources participating in fast-flux, these values are worrisome.

As demonstrated repeatedly, the contents of individual blacklists rarely overlap [19, 16, 33]. One plausible explanation for this disjointedness is that blacklists track a lot of ephemeral IP addresses. However, if fast-flux would have a statistical impact on this disjointedness, it would need to represent more than 1% of blacklist identifiers: as we found in this study.

Our leading interpretation of the fact that our fast-flux results are mutually disjoint with the blacklists we have access to is the following. None of the blacklists are tracking fast-flux. This interpretation is consistent with prior interpretation of the blacklist disjointedness generally, which is that each list is good, but very precise about what it is following and from what sensor vantage [19]. This interpretation is strengthened by the long observation time, over many years, and the consistency of both the size of the fast-flux networks and the lack of blacklist overlap during that time.

Algorithm 4 is an example of how the results from fast-flux measurement can be applied to continuous network situational awareness. We do not report an evaluation on a live network due to data access and publication issues; however, we provide the configuration so that network operators can apply it consistently as a test and compare results privately. The result would be that once enough domain-IP pairs are looked up to form a fast-flux network, one alerts

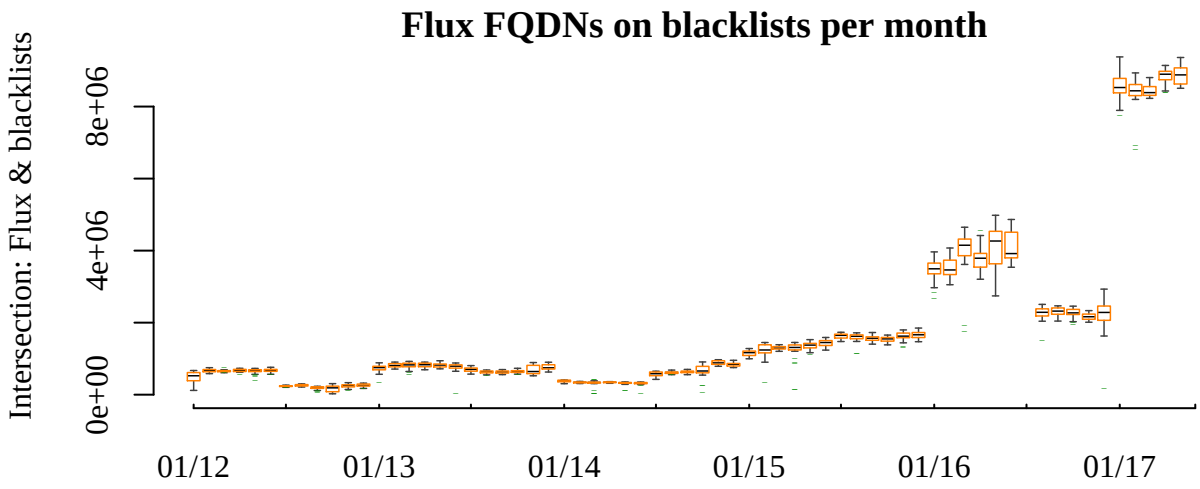


Figure 5: Summary statistics fast-flux–blacklist intersection. Plot follows the same conventions as Figure 4.

**Algorithm 4** Sample fast-flux watch-list usage in Pipeline

```
FILTER watchlistFromFastFlux
 sourceIPv4Address IN LIST ipList
END FILTER
EVALUATION alertWatchlist
 FILTER watchlistFromFastFlux
 ALERT EVERYTHING
 CHECK EVERYTHING PASSES
 END CHECK
END EVALUATION
```

on new connections if, at the time of the first connection attempt, they would add to a known fast-flux network. Defenders can effectively cap the viable size of fast-flux networks, reducing their usefulness to adversaries.

We can confirm that domains exhibiting parking on private IP addresses does not likely confound fast-flux network detection. We can also help explain why the recent literature uses 'parking' to mean typo-squatting for revenue generation: the older usage we study is much less common. Although parking on private IP addresses is rare, it is still an odd behavior. Further analysis may evidence what these resources are used for. However, given how long it takes to detect such parking, it is likely not the best use of defender resources. This assessment may change domains parked on private IP address space are used for high-impact attacks; however, our observations do not evaluate this concern. Future work should occasionally re-validate this assessment.

There are possible alternative interpretations of our parking results. Perhaps adversarial capability in utilizing parked domains in this way is still in an early phase of development. Alternatively, the domains exhibiting this kind of parking may be malicious, but simply are not found by any detection method used by the blacklists we compare against.

We have presented a combination of surprising results, on fast-flux, and unsurprising results, on parking. More notable is the importance of the method of long-term internet measurement in detecting trends and making conclusions. Passive DNS remains a useful tool, because it supports such long time scales while still keeping wide coverage feasible. However, our results also rely on archiving many years of contextual data, for routing, blacklists, and Alexa's top domains. The providers of these data do not have much incentive to store and archive long spans of data. Routeviews and RIPE RIS do a good job collecting this routing information. Similar initiatives for other internet metadata would improve the community's ability to pursue research.

## Acknowledgment

Thanks to Emily Sarneso for her help with IPFIX.



This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

umentation. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution. Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works. External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

* These restrictions do not apply to U.S. government entities.

## References


[1] Alexa. Alexa Internet, inc. – top sites. `http://www.alexa.com/topsites`, January 13, 2013.

[2] Alrwais, S., Yuan, K., Alowaisheq, E., Li, Z., and Wang, X. Understanding the dark side of domain parking. In *23rd USENIX Security Symposium (USENIX Security 14)* (San Diego, CA, Aug 2014), USENIX Association.

[3] CERT/NetSA at Carnegie Mellon University. CERT/CC Route Views Project Page. `http://routeviews-mirror.cert.org`. [Accessed: Feb 13, 2017].

[4] CERT/NetSA at Carnegie Mellon University. pyfixbuf. `http://tools.netsa.cert.org/pyfixbuf/index.html`. [Accessed: Jan 24, 2017].

[5] CERT/NetSA at Carnegie Mellon University. SiLK (System for Internet-Level Knowledge). `http://tools.netsa.cert.org/silk`. [Accessed: Feb 4, 2017].

[6] Claise, B., Trammell, B., and Aitken, P. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011 (INTERNET STANDARD), Sept. 2013.

[7] Cotton, M., Vegoda, L., Bonica, R., and Haberman, B. Special-Purpose IP Address Registries. RFC 6890 (Best Current Practice), Apr. 2013.

[8] Dickersin, K., Chan, S., Chalmersx, T., Sacks, H., and Smith, H. Publication bias and clinical trials. *Controlled clinical trials 8*, 4 (1987), 343–353.

[9] Fanelli, D. Negative results are disappearing from most disciplines and countries. *Scientometrics 90*, 3 (2012), 891–904.

[10] FarSight Security, Inc. nmsgtool. `https://archive.farsightsecurity.com/nmsgtool/`, Sep 25, 2013. [Accessed: Aug 12, 2014].

[11] Feitelson, D. G. From repeatability to reproducibility and corroboration. *ACM SIGOPS Operating Systems Review 49*, 1 (2015), 3–11.

[12] Holz, T., Gorecki, C., Rieck, K., and Freiling, F. C. Measuring and detecting fast-flux service networks. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium* (February 2008).

[13] ICANN. SSAC advisory on fast flux hosting and dns. Tech. Rep. SAC-025, Internet Corporation for Assigned Names and Numbers – Security and Stability Advisory Committee, March 2008.

[14] Knysz, M., Hu, X., and Shin, K. G. Charlatans' web: Analysis and application of global IP-usage patterns of fast-flux botnets. *University of Michigan Ann Arbor* (2011), 1–22.

[15] Konings, M. Final report of the gnso fast flux hosting working group. Tech. rep., Internet Corporation for Assigned Names and Numbers – Generic Names Supporting Organization, August 2009.

[16] Kührer, M., Rossow, C., and Holz, T. Paint it black: Evaluating the effectiveness of malware blacklists. Tech. Rep. TR-HGI-2014-002, Ruhr-Universität Bochum, Horst Görtz Institute for IT Security, June 2014.

[17] Levine, J. DNS Blacklists and Whitelists. RFC 5782 (Informational), Feb. 2010.

[18] MaxMind. Geolite2 free downloadable databases. `http://dev.maxmind.com/geoip/geoip2/geolite2/`, Jan 28, 2014.

[19] Metcalf, L. B., and Spring, J. M. Blacklist ecosystem analysis: Spanning Jan 2012 to Jun 2014. In *The 2nd ACM Workshop on Information Sharing and Collaborative Security* (Denver, Oct 2015), pp. 13–22.

[20] Nazario, J., and Holz, T. As the net churns: Fast-flux botnet observations. In *Malicious and Unwanted Software (MALWARE)* (Sep 2008), IEEE, pp. 24–31.

[21] Passerini, E., Paleari, R., Martignoni, L., and Bruschi, D. Fluxor: detecting and monitoring fast-flux service networks. *Detection of Intrusions and Malware, and Vulnerability Assessment* (2008), 186–206.

[22] RIPE Network Coordination Center. Routing information service (RIS). http://www.ripe.net/data-tools/stats/ris/routing-information-service, January 3, 2012.

[23] Route-Views. University of oregon route views project. http://www.routeviews.org, January 3, 2012.

[24] Ruef, D. Analysis pipeline v.5.6. http://tools.netsa.cert.org/analysis-pipeline5/index.html, Jan 7, 2017. [Accessed Jan 8, 2017].

[25] Salusky, W., and Danford, R. Know your enemy: Fast-flux service networks. Tech. rep., The Honeynet Project, July 13, 2007.

[26] Scarfone, K., and Mell, P. Guide to intrusion detection and prevention systems (IDPS). Tech. Rep. SP 800-94, U.S. National Institute of Standards and Technology, Gaithersburg, MD, Feb 2007.

[27] Spring, J. M. Large scale DNS traffic analysis of malicious internet activity with a focus on evaluating the response time of blocking phishing sites. Master's thesis, University of Pittsburgh, 2010.

[28] Spring, J. M. Modeling malicious domain name take-down dynamics: Why eCrime pays. In *eCrime Researchers Summit (eCRS)* (San Francisco, Sep 2013), IEEE, pp. 1–9.

[29] Spring, J. M., Metcalf, L. B., and Stoner, E. Correlating domain registrations and DNS first activity in general and for malware. In *Securing and Trusting Internet Names: SATIN* (Teddington, UK, Mar 2011).

[30] SURBL. Implementation guidelines. http://www.surbl.org/guidelines, Dec 9, 2011. [Accessed: Aug 1, 2014].

[31] Szurdi, J., Kocso, B., Cseh, G., Spring, J. M., Felegyhazi, M., and Kanich, C. The long "taile" of typosquatting domain names. In *23rd USENIX Security Symposium* (San Diego, Aug 2014), USENIX Association, pp. 191–206.

[32] Thomas, M., Metcalf, L., Spring, J. M., Krystosek, P., and Prevost, K. SiLK: A tool suite for unsampled network flow analysis at scale. In *IEEE BigData Congress* (Anchorage, Jul 2014), pp. 184–191.

[33] Verizon. 2015 data breach investigations report (DBIR). Tech. rep., 2015.

[34] Vissers, T., Joosen, W., and Nikiforakis, N. Parking sensors: Analyzing and detecting parked domains. In *Proceedings of the ISOC Network and Distributed System Security Symposium (NDSS)* (San Diego, CA, February 2015).

[35] Yadav, S., Reddy, A. K. K., Reddy, A., and Ranjan, S. Detecting algorithmically generated malicious domain names. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (2010), ACM, pp. 48–61.