Paint it Black: Evaluating the Effectiveness of Malware Blacklists

Marc Kührer, Christian Rossow, and Thorsten Holz

Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
{firstname.lastname}@ruhr-uni-bochum.de

Abstract. *Blacklists* are commonly used to protect computer systems against the tremendous number of malware threats. These lists include abusive hosts such as malware sites or botnet Command & Control and dropzone servers to raise alerts if suspicious hosts are contacted. Up to now, though, little is known about the *effectiveness* of malware blacklists. In this paper, we empirically analyze 15 public malware blacklists and 4 blacklists operated by antivirus (AV) vendors. We aim to categorize the blacklist content to understand the nature of the listed domains and IP addresses. First, we propose a mechanism to identify parked domains in blacklists, which we find to constitute a substantial number of blacklist entries. Second, we develop a graph-based approach to identify sinkholes in the blacklists, i.e., servers that host malicious domains which are controlled by security organizations. In a thorough evaluation of blacklist effectiveness, we show to what extent real-world malware domains are actually covered by blacklists. We find that the union of all 15 public blacklists includes less than 20% of the malicious domains for a majority of prevalent malware families and most AV vendor blacklists fail to protect against malware that utilizes *Domain Generation Algorithms.*

Keywords: Blacklist Evaluation, Sinkholing Servers, Parking Domains

1 Introduction

The security community needs to deal with an increasing number of malware samples that infect computer systems world-wide. Many countermeasures have been proposed to combat the ubiquitous presence of malware [1–4]. Most notably, researchers progressively explored network-based detection methods to complement existing host-based malware protection systems. One prominent example are endpoint reputation systems. The typical approach is to assemble a blacklist of endpoints that have been observed to be involved in malicious operations. For example, blacklists can contain domains of Command & Control (C&C) servers of botnets, dropzone servers, and malware download sites [5]. Such blacklists can then be queried by an intrusion detection system (IDS) to determine if a previously unknown endpoint (such as a domain) is known for suspicious behavior.

Up to now, though, little is known about the *effectiveness* of malware blacklists. To the best of our knowledge, the completeness and accuracy of malware blacklists was never examined in detail. Completeness is important as users otherwise risk to miss notifications about malicious but unlisted hosts. Similarly, blacklists may become outdated if entries are not frequently revisited by the providers. While an endpoint may have had a bad reputation in the past, this might change in the future (e.g., due to shared hosting).

In this paper, we analyze the effectiveness of 15 public and 4 anti-virus (AV) vendor malware blacklists. That is, we aim to categorize the blacklist content to understand the nature of the listed entries. Our analysis consists of multiple steps. First, we propose a mechanism to identify parked domains, which we find to constitute a substantial number of blacklist entries. Second, we develop a graph-based approach to identify sinkholed entries, i.e., malicious domains that are mitigated and now controlled by security organizations. Last, we show to what extent real-world malware domains are actually covered by the blacklists.

In the analyzed blacklist data we identified 106 previously unknown sinkhole servers, revealing 27 sinkholing organizations. In addition, we found between 40 - 85% of the blacklisted domains to be unregistered for more than half of the analyzed blacklists and up to 10.9% of the blacklist entries to be parked. The results of analyzing the remaining blacklist entries show that the coverage and completeness of most blacklists is insufficient. For example, we find public blacklists to be impractical when it comes to protecting against prevalent malware families as they fail to include domains for the variety of families or list malicious endpoints with reaction times of 30 days or higher.

Fortunately, the performance of three AV vendor blacklists is significantly better. However, we also identify shortcomings of these lists: only a single blacklist sufficiently protects against malware using *Domain Generation Algorithms* (DGAs) [3], while the other AV vendor blacklists include a negligible number of DGA-based domains only. Our thorough evaluation can help to improve the effectiveness of malware blacklists in the future.

To summarize, our contributions are as follows:

- We propose a method to identify parked domains by training an SVM classifier on seven inherent features we identified for *parked* web sites.
- We introduce a mechanism based on blacklist content and graph analysis to effectively identify malware sinkholes without *a priori* knowledge.
- We evaluate the effectiveness of 19 malware blacklists and show that most public blacklists have an insufficient coverage of malicious domains for a majority of popular malware families, leaving the end hosts fairly unprotected. While we find blacklists operated by AV vendors to have a significantly higher coverage, up to 26.5% of the domains were still missed for the majority of the malware families, revealing severe deficiencies of current reputation systems.

2 Overview of Malware Blacklists

Various malware blacklists operated by security organizations can be used to identify malicious activities. These blacklists include domains and IP addresses, which have been observed in a suspicious context, i.e., hosts of a particular type such as C&C servers or—less restrictive—endpoints associated to malware in general. Table 1 introduces the 15 public malware blacklists that we have monitored for the past two years [6]. For the majority of blacklists, we repeatedly obtained a copy every 3 hours (if permitted). The columns *Current* state the number of entries that were listed at the end of our monitoring period. The columns *Historical* summarize the entries that were once listed in a blacklist, but became delisted during our monitoring period. For reasons of brevity, we have omitted the number of listed *IP addresses* per blacklist, as we mainly focus on the blacklisted *domains* in our analyses. For all listed domains, we resolved the IP addresses and stored the name server (NS) DNS records. If blacklists contained URLs, we used the domain part of the URLs for our analysis.

Four blacklists are provided by `Abuse.ch`, of which three specifically list hosts related to the *Palevo* worm and the banking trojans *SpyEye* and *ZeuS*. The *Virustracker* project lists domains generated by DGAs, and the *Citadel* list includes domains utilized by the *Citadel* malware (that was seized by Microsoft in 2013 [7]). *UrlBlacklist* combines user submissions and other blacklists, covering domains and IPs of various categories, whereas we focus on the malware-related content. The *Exposure* [4] blacklist included domains that were flagged as malicious by employing passive DNS (pDNS) analysis. The *Abuse.ch AMaDa* and the *Exposure* lists were discontinued, yet we leverage the collected historical data.

Table 1. Observed content of the analyzed malware blacklists (‡ denotes C&C blacklists)

Blacklist	Domains (in #)		Observ. (days)
	Current	Historical	
AMaDa [8]‡	0	1,494	267
Citadel [7]‡	4,634	0	66
Cybercrime [9]‡	1,070	0	121
Exposure [4]	0	107,183	559
Malc0de [10]	2,121	20,135	832
MDL Hosts [11]	1,653	11,996	832
MDL ZeuS [11]‡	12	1,675	829
MW-Domains [12]	23,396	37,490	832
Palevo Tracker [8]‡	35	147	542
Shadowserver [13]‡	0	0	832
Shallalist [14]	20,677	48	320
SpyEye Tracker [8]‡	123	956	832
UrlBlacklist [15]	127,745	281	824
Virustracker [16]	12,066	56,269	196
ZeuS Tracker [8]‡	759	8,042	832

Besides these public blacklists, we have requested information from four antivirus (AV) vendors, namely *Bitdefender TrafficLight* [17], *Browserdefender* [18], *McAfee Siteadvisor* [19], and *Norton SafeWeb* [20]. These blacklists cannot be downloaded, but we can query if a domain is listed. We thus do not know the overall size of these blacklists and omit the numbers in Table 1.

Datasets. We divide the 15 public blacklists into three overlapping datasets. The first dataset, referred to as $S_{C\&C}$, consists of domains taken from the sources primarily listing endpoints associated to C&C servers, denoted by ‡ in Table 1. We extend $S_{C\&C}$ with the IP addresses to which any of these domains at some point resolved to. The second, coarse-grained dataset S_{Mal} includes the domains that were at any time listed in any of the 15 blacklists (including $S_{C\&C}$) and the resolved IPs. Last, we generate a third dataset S_{IPs}, covering all currently listed IP addresses by any of the 15 public blacklists (i.e., 196,173 IPs in total). This dataset will help us to verify if blacklists contain IPs of sinkholing servers.

Paper Outline. Motivated by the fact that blacklists contain thousands of domains, we aim to understand the nature of these listings. We group the entries in four main categories: domains are either i) unregistered, ii) controlled by parking providers, iii) assigned to sinkholes, or iv) serve actual content. Unregistered domains can easily be identified using DNS. However, it is non-trivial to detect parked or sinkholed domains. We thus propose detection mechanisms for these two types in Section 3 (parking domains) and Section 4 (sinkholed domains). In Section 5, we classify the blacklist content and analyze to what extent blacklists help to protect against real malware. Note that a longer version of this paper with more technical details is available as a technical report [21].

3 Parking Domains

Parking domains make up the first prominent class of blacklist entries. They are mainly registered for the purpose of displaying web advertisements, so called ads. Typically no other, real content is placed on these domains. As domains associated with malicious activities tend to be parked to monetize the malicious traffic [22], we expect parked domains to constitute a substantial number of blacklist entries. Unfortunately, parking services have diverging page templates to present the sponsored ads. As such, it is not straightforward to identify these sites, e.g., with pattern-matching algorithms. In order to identify parking domains in the blacklists, we thus introduce a generic method to detect parked domains that can cope with the diversity of parking providers.

3.1 Datasets

We first assemble a labeled dataset by manually creating patterns and applying pattern-matching algorithms [23, 24]. Note that these patterns are far from complete due to the high diversity of page templates. We leverage the resulting dataset as ground truth to evaluate our generic detection model for parked domain names later on. We generate the labels based on Li *et al.*'s [22] observation that parking providers either modify the authoritative NS sections of a domain to point to dedicated parking NS or employ web-based (i.e., HTTP-, HTML-, or JavaScript-based) redirections to forward users to the final parking content.

Based on our recorded DNS information, we first label domains following the DNS-based type of redirection. That is, we analyze the 233,772 distinct name servers aggregated while processing the blacklist data. We split the NS hostnames into tokens and searched for terms indicating parking such as `park`, `sell`, and `expired` and labeled NS whose hostnames match one of these terms as potential parking name servers. We monitored a fraction of parked domains that switched their authoritative NS to a different parking provider. As a result, we extracted the domains that used the parking NS identified in the previous step from the aggregated DNS data, requested latest `NS` records for each domain, and inspected the most frequently used NS. In addition, we consulted the *DNS DB* [25], a *passive DNS* (pDNS) database. That is, for each identified parking

NS, we requested 50,000 randomly selected domains the NS was authoritative for, obtained current `NS` records for each domain, and again checked the NS hostnames against terms indicating parking behavior. Overall, using these techniques and manual inspection, we identified 204 NS operated by 53 parking providers.

A minority of parking services employ web-based techniques to redirect users to the actual parking content. The DNS-based methods discussed so far did not detect these providers. However, we identified parked domains that are often transferred between providers, thus we assume that some domains found in pDNS data of the previously identified parking NS at some point have relocated to providers utilizing web-based redirection techniques. To identify these services, we extracted 10,000 randomly chosen domains from the pDNS data of each parking NS, analyzed the domain redirection chains, and identified 14 patterns of *landing pages* [21] to which users are redirected to when visiting parked domains. These landing pages belong to parking, domain, and hosting providers.

Finally, we use the parking NS and landing pages to manually extract 47 descriptive strings, in the following referred to as *identifiers* (IDs) [21]. These IDs can be found in the HTTP responses of many parked domains (e.g., `<frame src="http://ww[0-9]{1,2}` and `landingparent`). We use these IDs to create the *parked domains* dataset $\mathcal{P}$ that consists of 5,000 randomly chosen domains from the pDNS database we find to utilize a verified parking NS or include at least one identifier. We further create a dataset $\mathcal{B}$ of benign (i.e., non-parked) domains. We utilize the Top 5,000 domains taken from the Alexa Top Ranking [26] and verify that none of these domains trigger a landing page or ID match.

3.2 Feature Selection and Classification

Pattern matching allowed us to identify a subset of all parking services. However, we seek to identify intrinsic characteristics of parking websites that are more generic than the manually assembled classification described above. We thus studied subsets of our benign and parked domain sets and identified two, respectively, five generic features based on HTTP and HTML behavior.

The first HTTP-based feature is determined by the redirection behavior when domains are directly accessed without specifying any subdomains. For benign domains, automated redirection to the common `www` subdomain is often enforced. Parked domains, in contrast, typically do not exhibit similar behavior.

Our second feature is based on the observation that parked domains deliver similar content on random subdomains and the domains itself while benign domains tend to serve differing content for arbitrary subdomains (if at all). We measure the normalized Levenshtein ratio [27] between the HTML content gathered by accessing the domain and a randomly generated subdomain. If the HTTP request for the subdomain failed (e.g., due to DNS resolution), the feature is set to -1, otherwise the value is in the range from 0 (no similarity) to 1 (equal).

The first HTML-based feature is derived from the observation that many parked domains display sponsored ads while the textual content is negligible. Contrary, most benign domains deliver a substantial amount of human-readable content in the form of coherent text fragments. Our third feature thus defines the ratio of human-readable text in relative to the overall length in returned web content after removing HTML tags, JavaScript codes, and whitespaces.

Next, we outline three features to express the techniques that landing pages utilize to embed parking content. That is, we account for the observation that most parked domains use JavaScript or frames to display sponsored ads. In the fourth feature, we measure the ratio of JavaScript code. In the fifth feature, we count the number of `<frame>` tags on landing pages. As many page templates utilizing frames contain only the basic HTML structure and the frameset, the frame count is particularly powerful in combination with the ratio of human-readable text (feature 3). A fraction of parked domains, however, do not rely on JavaScript or frames and directly embed the referral links into the HTML code. We observed many of these parking providers to specify rather long attributes in the referral `<anchor>` tags (e.g., multiple mutual IDs in the `href` attribute). As parked domains tend to serve numerous referral links, the average length of `<anchor>` tags is expected to be considerably higher than in content served by the majority of benign domains, as expressed in the sixth feature.

The seventh feature is defined by the `robots` value specified in the `<meta>` tag. Parked domains in our dataset either did not specify a `robots` value (thus using the default `index + follow`) or defined one of the values `index + nofollow`, `index + follow`, or `index + follow + all`. Parking providers monetize the domains and are interested in promoting their domains, thus permitting indexing by search engines. In contrast, benign sites often customize the indexing policies—we identified 31 different `robots` values. As the `robots` value is a concatenation of tokens, we mapped all possible single tokens to non-overlapping bitmasks and use the numerical value of the bit-wise OR of all tokens as feature.

Most parking services rely on JavaScript to display referral links and advertisements. The HTML-based features (3 - 7) thus require JavaScript execution when aggregating the feature values. As the initially served content before executing JavaScript and the final content after executing JavaScript both are characteristic for parked domains (and might be entirely different, e.g., when JavaScript is used for redirection), we obtain two feature values for each of the HTML-based features accordingly, resulting in 12 feature values per domain.

We use these 12 feature values to classify domains as either parked or benign (i.e., non-parking). We evaluated our approach for different types of machine-learning algorithms using RapidMiner [21, 28] and achieved the best results for *support vector machines* (SVMs) using the *Anova* kernel [29].

3.3 Evaluation

Cross-fold Validation. We evaluate the feature set with a 10-fold cross validation using all domains in our *benign* $\mathcal{B}$ and *parking* $\mathcal{P}$ sets and achieve an average detection rate of 99.85% correctly classified domains while the false positive (FP) rate is at 0.11% and the false negative (FN) rate at 0.04%.

Individual Dataset. To evaluate our approach on an individual dataset and discuss false positives and negatives as suggested by Rossow *et al.* [30], we split the 10,000 labeled *benign* and *parked* domains into a training set S_{Train} consisting of 1,000 benign and 1,000 parked domains and a test set S_{Test} that includes the remaining 8,000 domains. The resulting detection model correctly classifies 7,969 domains in S_{Test} (99.6%) as benign or parked, resulting in 5 FPs (0.1%) and 26 FNs (0.3%). When investigating the FPs, we find each domain to have a ratio of less than 20% for human-readable text (feature 3) in combination with a high average length of `<anchor>` tags (feature 6). Further, all domains respond to random subdomain requests and serve similar web content (i.e., the normalized Levenshtein ratio ≥ 0.9). When analyzing the 26 FNs, we find domains that either switched between redirecting to parking and benign content or delivered parking content on the second visit. As we visited each domain only once during feature attribution, we did not observe parking behavior for these domains.

Real-World Data. Finally, we verify our approach on real-world data containing significantly more unlabeled domains. We obtained the Top 1M domains from the Alexa Ranking 12 weeks after the Top 5k domains were gathered for the *benign set* $\mathcal{B}$. We expect only a few parked domains in this dataset, thus we mainly are interested if our approach can handle the diverse page structures of benign web pages without high FP rates. We could aggregate feature values for 891,185 domains while the remaining domains either did not resolve to IP addresses or provide web content within a time frame of 15 seconds, respectively, replied with blank content or HTTP error codes. We further remove 957 domains already covered by S_{Train}, thus the resulting set S_{Alexa} is defined by 890,228 domains. We then match the content of each domain against the IDs and landing pages introduced in Section 3.1 to estimate a lower bound of FPs and FNs. We cannot ensure the correctness of the IDs, hence might erroneously flag benign domains as parked. We thus manually verify potential false classifications.

Table 2. Results of S_{Alexa} and $S_{Current}$ (Parked = Domains flagged as parked by IDs or classifier (CL), INT = Intersection of domains flagged as parked by IDs and CL, FI = Domains falsely flagged as parked by IDs, New = Domains detected by CL but not found by IDs)

		Parked (#)					Rates (%)		
Subset	Size	ID	CL	INT	# FI	# New	CD	FP	FN
S_{Alexa}	890,228	5,208	8,709	4,596	71	626	99.5	0.4	0.1
$S_{Current}$	33,121	3,747	5,623	3,027	28	2,336	(98.7)	0.8	(0.5)

As shown in Table 2, we achieve a correct detection rate (CD; the sum of true positive and true negative rate) of 99.5%, a FP rate of 0.4%, and a FN rate of 0.1%. The IDs flag 5,208 domains as parked, yet we find 71 of the domains to be incorrectly flagged. The classifier marks 8,709 domains as parked of which 4,596 domains are verified by the IDs. Of the remaining domains we find 626 to be parked that are not detected by the IDs, resulting in 5,222 parked domains detected by the classifier. These results indicate that 0.6% of the Alexa 1M domains, i.e., more than 1/200 of the most popular domains, are parked. More specifically, we identify 36 parked domains in the Alexa Top 10k while 432, respectively, 1,170 domains are parked in the Top 100k and Top 250k, showing that the majority of parked domains are not ranked in the Top 250k Alexa. During the manual verification process, we find the vast majority of parked domains to be associated to domain resellers such as *Above*, *GoDaddy*, and *Sedo* [21].

We now turn back to our original goal, i.e., classifying the content of blacklists. We thus extracted currently blacklisted domains from our blacklist set S_{Mal} twelve weeks after generating the *benign* $\mathcal{B}$ and *parking* $\mathcal{P}$ sets used for S_{Train}. We name this dataset $S_{Current}$ and again remove domains already included in S_{Train}. Of the 158,648 currently listed domains, we obtained feature values for 33,121 domains. The remaining domains either were unregistered or replied with HTTP error codes. The classifier defines 5,623 domains as parked, of which 3,027 domains are verified by the IDs. When manually investigating the remaining 2,596 domains flagged by the classifier, we identify 2,336 parked domains not detected by the IDs and 260 FPs (0.8%). The FPs are mostly caused by adult content and web directory sites with similar characteristics as parked domains. When taking a closer look at the initial high number of 692 FNs, we find 538 domains not serving parking content at all (i.e., referral links). More precisely, one domain reseller causes most of the FNs, as we identify 506 domains (73.1% of all FNs) redirecting to `hugedomains.com`, providing web content not exhibiting common parking behavior. To evaluate if our approach fails to detect domains associated with this reseller due to missing training data, we adjusted S_{Train} to cover a partition of these domains and find the detection model to correctly classify these domains as parked, reducing the FN rate to 0.5%.

4 Sinkholes

Next to parking domains, also so called sinkholing servers (*sinkholes*) are prominent types of blacklist entries. Sinkholes are operated by security organizations to redirect malicious traffic to trusted hosts to monitor and mitigate malware infections. In order to track sinkholes in our blacklist data, we first identify intrinsic characteristics of these servers. We thus obtained an incomplete list of sinkhole IPs and domains by manual research and through collaboration with partners. In pDNS, we then observed that domains associated with sinkholes tend to resolve to the corresponding IPs for a longer period of time, thus the monitored DNS `A` records are persistent. Contrary, malicious domains tend to switch to various IPs and Autonomous Systems (AS) within a short time frame to distribute their activities to different providers [5]. We also found sinkholed domains switching to other sinkholes provided by the same organization or located in the same AS, and discovered domains that were relocated to other sinkhole providers.

Sinkhole operators often use their resources to monitor as many domains of a malware family as possible. We thus find sinkhole IP addresses to be typically assigned to numerous (up to thousands of) domains. In the majority of cases, the domains resolving to a specific sinkhole IP shared the same NS such as `torpig-sinkhole.org` or `shadowserver.org`. We thus argue that if multiple domains resolve to the same IP address but do not utilize the same NS, the probability that this IP is associated with a sinkhole is considered to be low.

Another observation is the content sinkholes serve upon HTTP requests. When requesting content from randomly chosen sinkholed domains using `GET / HTTP/1.1`, we find sinkholes to either not transfer any HTML data (i.e., closed

HTTP port or web servers responding with 4xx HTTP codes) or serve the same content for all domains as monitored for `zinkhole.org`. We thus assume that domains resolving to the same set of IP addresses but serving differing content do not belong to sinkholes and are rather linked to services such as shared hosting.

4.1 Sinkhole Identification

Based on these insights we introduce our approach to identify sinkholes in the blacklist datasets $S_{C\&C}$ and S_{Mal}. The datasets consist of currently listed and historical domains and the IPs to which any of the domains resolved to. For each domain, we aggregate current DNS records and web content while we obtain reverse DNS records, AS and online status details, and web content for all IPs.

Filtering Phase. In a first step, we aim to filter IP addresses sharing similar behavior as sinkholes to eliminate potential FPs. We thus remove the IPs associated with parking providers using the detection mechanism introduced in Section 3. To identify IPs of potential shared hosting providers serving benign or malicious content, we analyze the aggregated HTTP data. We define IPs to be associated with shared hosting when we obtain varying web content (i.e., normalized Levenshtein ratio ≤ 0.9) for the domains resolving to the same set of IPs. Furthermore, we expect sinkholes to be configured properly, thus we do not consider web servers as sinkholes that delivered content such as `it works`.

As our datasets might include erroneously blacklisted benign domains, we filter likely benign IPs such as hosting companies and *Content Delivery Networks* with the following heuristic: we do not expect the Alexa Top 25k domains to be associated to sinkholing servers. We thus obtained the HTTP content of each domain, extracted further domains specified in the content, and requested DNS `A` records for all domains. The resulting dataset S_{Benign} includes 105,549 presumably benign IPs. We acknowledge that this list does not remove all false listings in the blacklist datasets, however, this heuristic improves our data basis.

To further reduce the size of the datasets, we eliminate IPs associated to *Fast Flux* with the following heuristic: we define an IP to be associated with Fast Flux when at least 50% of the blacklisted domains currently resolving to this IP are found to be Fast Flux domains, whereas we define a Fast Flux domain as follows: i) the domain resolved to more than 5 distinct IPs during our observation time and ii) at least half of these IPs were seen within two weeks. As we expect the ratio of fast flux domains associated to individual sinkhole IP addresses to be rather low, we assume to not remove any sinkholing servers.

Graph Exploration. The actual sinkhole identification follows the intuition that IPs of sinkholes mostly succeed malicious IPs in the chain of resolved IPs for a high number of domains and are persistent for a longer period of time. For each dataset $S_{C\&C}$ and S_{Mal}, we map this assumption onto a separate directed graph $G = (V, E)$, whereas the domains and IPs in the datasets are represented as vertices $v \in V$. The edges $e \in E$ are determined by the relationship between the domains and IPs. We define $u \in V$ to be a parent node of v when there exists a directed edge $e = (u, v)$ and define $w \in V$ to be a child node of v when there exists a directed edge $e = (v, w)$. The edges $e \in E$ are defined as follows:

(i) For each domain $v \in V$, we add a directed edge $e = (v, w)$ if domain v at some point resolved to IP $w \in V$.

(ii) For each domain $v \in V$, we add an edge $e = (w_o, w_n)$ if v resolved to IP $w_o \in V$, switched to a new IP $w_n \in V$, and never switched back to IP w_o.

In step (i), we assign the resolved IPs to each domain in our datasets. In step (ii), we add a domain's history of `A` records (i.e., resolved IPs) to the graph.

We name $deg^{-}(v)$ the in-degree of node v, resembling the number of parent nodes. In our graph model, the in-degree represents the number of domains that currently are or were once resolving to node v and the number of IPs preceding v in the resolver chain. For sinkholes, the in-degree is considerably higher than the average in-degree as sinkholes usually succeed malicious IPs in the chain of resolved IPs and a single sinkhole IP is often used to sinkhole multiple domains.

We further refer to $deg^{+}(v)$ as the out-degree of node v, resembling the number of child nodes, e.g., IP addresses that followed node v in the resolver chain. We find the out-degree of sinkhole IPs to be significantly lower than the average out-degree because sinkhole IPs are persistent for a longer period of time. As a result, the ratio $R = \frac{deg^{-}(v)}{deg^{+}(v)}$ is expected to be high for sinkholes.

We use the resulting graph to create a list of potential sinkholes S_{pot} by adding all IP addresses $v \in V$ which meet these requirements:

(i) The IP address must respond to ICMP Echo or HTTP requests.

(ii) At least D domains are currently resolving to this IP, whereas the value D is defined by the average number of active domains per IP in our set.

(iii) The ratio R exceeds a threshold T, whereas T is defined as the average ratio of all IP addresses $v \in V$.

(iv) All domains associated with a single IP address utilize the same NS.

We then manually verify each IP in S_{pot} whether it is a FP or associated with a sinkhole by analyzing the utilized NS, served web content, reverse DNS record, and AS details, and also employ a service provided by one of our collaboration partners listing known sinkhole IPs. Verified sinkholes are added to the set S_{ver}.

We chose these rather hard requirements as most sinkhole operators have little incentive to disguise the existence of their sinkholes. We thus hypothesize that this list of requirements will even hold once our sinkhole detection technique is known. However, as we might have missed sinkhole IPs due to the strict requirements for S_{pot}, we explore the neighboring IP addresses of S_{ver} in the second phase of the sinkhole identification. Before doing so, we extract the NS of the domains resolving to the IPs in S_{ver}, manually check whether the NS are specifically used in conjunction with sinkholed domains and if so, we add the NS to a trusted set S_{NS}. Further on, to also detect inactive sinkholes at a later stage, we create a mapping of trusted NS and the AS the corresponding sinkhole $s \in S_{ver}$ is located in, defined by $S_{NS\_AS} = \{(ns_s, AS_s) \mid ns_s \in S_{NS}\}$.

Sinkhole operators might relocate domains to different sinkholes in the same organization and AS, thus we explore the parent and child nodes of each sinkhole to identify yet unknown sinkholes. For each $ip \in V$, we check whether ip is a parent or child node of a known sinkhole $s \in S_{ver}$, whereas we only consider IPs abiding $AS_{ip} = AS_s$. If ip is found to be a neighboring node of at least two sinkholes, we define ip to be a potential sinkhole and add it to S_{pot}. Further, ip is added to S_{pot} when it is a parent or child node of at least one sinkhole in the same AS and the domains resolving to both IP addresses share the same NS.

To identify sinkholes which cannot be found by exploring parent and child nodes, we leverage the trusted name servers $ns \in S_{NS}$. As we defined these NS to be exclusively used for sinkholed domains, we check if the authoritative NS of the domains currently resolving to each IP $ip \in V$ can be found in S_{NS}.

The previous exploration mechanisms traced active sinkholes only as we require domains to resolve to the IPs of potential sinkholes. Our blacklist dataset also includes historical data, thus we are interested in obtaining a list of sinkholes which were active in the past. Inactive sinkholes presumably do not have domains currently resolving to them, hence we cannot leverage the NS data as conducted in the previous step. Instead, we examine the domains which once resolved to each $ip \in V$ in our dataset, obtain the currently most utilized name server ns, and check if ns is covered by S_{NS}. If $ns \in S_{NS}$ is true, the ip is either of malicious character and the domains once resolving to ip are now sinkholed or we identified an inactive sinkhole and the domains were relocated to other sinkholes. To distinguish between malicious and sinkhole IP we check if (ns_{ip}, AS_{ip}) is listed in S_{NS_AS}. If this is true, we add ip to S_{pot} as we assume that malicious IPs are not located in the same AS in which we found verified sinkholes.

4.2 Evaluation

We now evaluate our method on the datasets $S_{C\&C}$ and S_{Mal}. On $S_{C\&C}$, the filtering step removed 1,144 IPs listed in S_{Benign} or associated with parking providers or Fast Flux. The resulting graph consists of 41,269 nodes and 371,187 edges. In the first phase of the graph exploration our approach adds 20 IPs to S_{pot}, which we manually verified to be associated with sinkholes. In the second phase, we identify 6 sinkholes by exploring the parent and child nodes of the already verified sinkholes, 11 sinkholes by analyzing the actively used NS, and 8 sinkholes by exploring the NS of historically seen domains. Table 3 outlines the operators of the verified sinkholes and the number of distinct AS. The sinkholes listed as *Others* are associated with organizations such as *Abuse.ch* and *Echo-Source*. In total, we discovered 45 sinkholes in $S_{C\&C}$ without any false positives.

On the larger and more distributed dataset S_{Mal}, we filter 7,349 IPs, resulting in a graph of 277,315 nodes and 4,690,369 edges. The first phase of the graph exploration identifies 80 IPs to be potential sinkholes. We are able to verify 59 of these IPs to be associated with sinkholes and find 10 IPs to serve 403 (Forbidden) and 404 (Not Found) HTTP error codes or empty HTTP responses for all associated domains. Another 7 IPs do not accept HTTP requests due to the HTTP port being closed. We assume that these 17 IPs are either associated with sinkholes or hosting companies, which deactivated misbehaving accounts or servers. The remaining 4 IPs in S_{pot} are considered to be FPs as two IPs serve benign content (i.e., related to adult content and the DNS provider `noip.com`), one IP replies with a single string for all known domains, and the last IP is still distributing malicious content. Based on the 59 verified sinkholes, we perform the second phase and detect 14 sinkholes by exploring the parent and child nodes, 19 sinkholes by monitoring the actively utilized NS, and 14 sinkholes by exploring the NS of previously seen domains. In *Others*, we summarize operators such as *Fitsec*, *Dr. Web*, and the *U.S. Office of Criminal Investigations*.

Table 3. Sinkhole IPs identified in $S_{C\&C}$ and S_{Mal}

		$S_{C\&C}$		S_{Mal}	
Organization	# AS	Active	Inactive	Active	Inactive
Anubis Networks	1	1	0	4	0
Cert.pl	1	4	0	4	0
GeorgiaTech/SinkDNS	5	0	0	8	1
Microsoft	3	7	2	11	2
Others	17	4	1	18	4
PublicDomainRegistry	7	10	7	20	11
Shadowserver	1	0	0	5	0
Torpig-Sinkhole	2	4	1	8	2
Zinkhole	1	4	0	7	1

Our detection technique identified 106 IPs, which we verified to be associated with sinkholes, 17 IPs of potential sinkholes, and 4 IPs, which are falsely added to S_{pot} in the first exploration phase. The second phase does not cause any FPs but doubles the number of sinkholes.

5 Blacklist Evaluation

Based on the findings in the previous sections, we now proceed to analyze the content of the monitored malware blacklists in regards to multiple characteristics.

5.1 Classification of Blacklist Entries

We introduced detection mechanisms for parked domains and sinkholing servers, which are covered by blacklists. Table 4 outlines how many of the currently listed domains ($S_{Current}$) and IPs (S_{IPs}) can be assigned to one of these categories.

Table 4. Classification of $S_{Current}$ and S_{IPs}

	(in %)		Sinkholed		
Blacklist	Unreg.	Parked	Domains	(% / #)	# IPs
Citadel	23.6	0.2	70.4	3,263	n/a
Cybercrime	40.1	1.6	4.2	45	0
Malc0de	12.0	1.5	0.0	0	0
MDL Hosts	18.0	3.0	0.4	6	n/a
MDL ZeuS	41.6	0.0	8.3	1	0
MW-Domains	52.1	2.4	2.8	659	n/a
Palevo Tracker	0.0	0.0	2.9	1	1
Shallalist	45.7	10.9	0.9	190	1
Shadowserver	n/a	n/a	n/a	n/a	4
SpyEye Tracker	47.2	0.0	19.5	24	2
UrlBlacklist	72.3	3.1	1.7	2,211	3
Virustracker	85.1	8.7	3.5	426	n/a
ZeuS Tracker	52.0	0.3	0.1	1	0

The `Abuse.ch` blacklists as well as *MDL ZeuS* include a low number of parked domains. In contrast, we observe a high number of parked domains for blacklists that have only a few historical entries (cf. Table 1 in Section 2). Particularly for *Shallalist* and *UrlBlacklist*, we assume that the listed domains are not reviewed periodically as more than 57%, respectively, 77% of all domains are either non-existent, parked, or associated to sinkholes while the number of historical entries is almost negligible. When taking a look at *Virustracker*, we find 8.7% of the currently listed domains to be parked. *Virustracker* consists of DGA-based domains next to a partition of hard-coded malware domains that are valid and blacklisted for a longer period of time. The classification results indicate that the hard-coded domains are parked significantly more often than the DGA-based domains, i.e., when inspecting a random subset of 25 DGA-based and 25 hard-coded domains, only a single DGA-based domain was parked while more than 40% of the hard-coded domains were associated to parking. We thus assume that many of the persistent domains are parked to monetize the malicious traffic.

5.2 Blacklist Completeness

Next, we aim to answer how complete the blacklists are, i.e., we measure if they cover all domains for popular malware families. We thus turn from analyzing *what is* listed to evaluating *what is not* blacklisted. To the best of our knowledge, we are the first to analyze the completeness of malware blacklists. Estimating the completeness is challenging as it requires to obtain a ground truth first, i.e., a set of domains used by each malware family. To aggregate a dataset of malicious domains we leverage analysis reports of our dynamic malware analysis platform Sandnet [31]. We inspect the network traffic of more than 300,000 malware samples that we analyzed since Mar. 2012 and identify characteristic patterns for the C&C communication and egg download channels of 13 popular malware families. Our dataset includes banking trojans, droppers (e.g., *Gamarue*), ransomware (e.g., *FakeRean*), and DDoS bots (e.g., *Dirtjumper*), thus represents a diverse set of malware families. Per malware family, we manually identify typical communication patterns and extract the domains for all TCP/UDP connections that match these patterns. Next to regular expressions, we use traffic analysis [32] and identify encrypted C&C streams using probabilistic models [33] to classify the malware communication. We ensured that these fingerprints capture generic characteristics per malware family, guaranteeing that the number of false negatives is negligible (see [32] and [33] for details). We manually verified a subset of the suspicious communication streams and did not identify any false classifications. Admittedly, our dataset is limited to a small subset of the overall malware population only. Given the subset of malware samples, the set of extracted domains is thus by no means complete. However, our dataset serves as an independent statistical sample. In addition, polymorphism creates tens of thousands new malware *samples* daily, whereas the number of new malware *variants* (e.g., using different C&C domains) is much lower [30]—indicating that our dataset achieves reasonable coverage, as also indicated in the experiments.

We evaluate the completeness of the blacklists by computing the ratio of the malware domains observed in Sandnet that are also blacklisted. Table 5 outlines our evaluation results per family. The second column shows the number of domains we obtained from Sandnet per family. The remaining columns represent the results for particular blacklist datasets as introduced in Section 2, while S_{AV} is defined by the union of all four AV vendor blacklists. Our analysis shows that the public blacklists detect less than 10% of the malicious domains for eight ($S_{C\&C}$) and five (S_{Mal}) malware families, respectively. As a result, the detection capabilities of an IDS or AV software using these blacklists is insufficient, even when combining multiple blacklists that employ different listing strategies. The public blacklists do achieve detection rates higher than 50% for particular families because of highly specialized listing policies such as in the `Abuse.ch` trackers and Microsoft's list of *Citadel* domains, yet they fail to detect the other families—even though families such as *Sality* are known since 2003.

Table 5. Coverage of malware domains

		Public (%)		AV Vendors (%)		
Family	# Dom.	$S_{C\&C}$	S_{Mal}	S_{AV}	S_{BD}	S_{MA}
Citadel	225	87.6	89.3	96.0	57.3	79.1
Dirtjumper	47	2.1	2.1	80.9	63.8	40.4
FakeRean	34	0.0	17.7	73.5	50.0	58.8
Gamarue	127	6.3	18.9	86.6	62.2	47.2
Gbot	321	0.0	0.0	100.0	77.3	100.0
Palevo	58	51.7	58.6	93.1	63.8	89.7
Ponyloader	210	4.3	21.0	95.7	71.4	65.7
Pushdo	42	0.0	9.5	92.9	64.3	78.6
Rodecap	9	11.1	11.1	100.0	66.7	44.4
Sality	417	0.0	1.2	82.3	73.4	27.3
SpyEye	145	56.6	57.9	83.4	26.9	61.4
Tedroo	7	0.0	0.0	85.7	57.1	28.6
ZeuS	47	51.1	53.2	95.7	51.1	61.7

Three of the blacklists operated by AV vendors perform significantly better. Looking at the union of the blacklists, at least 70% of the domains per family are detected. More than 90% of the domains were listed for seven of the 13 families. We also look at the breakdown of S_{AV}, i.e., how well the individual blacklists perform. Table 5 includes the two blacklists that perform best: S_{BD} is operated by *Bitdefender* and S_{MA} by *McAfee*. Surprisingly, these blacklists have a non-negligible separation—combining them significantly increases the overall coverage for many families. We do not list the remaining two blacklists due to space constraints, however, note that *Norton* performs similar to *Bitdefender* and *McAfee* while *Browserdefender* fails to detect any domain for the majority of families and covers only 2 - 7% of the domains for the other malware families.

5.3 Reaction Time

For the domains seen in Sandnet which are also covered by S_{Mal}, we additionally estimate the *reaction time* of the blacklists. That is, we measure how long it takes to blacklist the domains once they were seen in Sandnet. As the domains could have been performing malicious activities before we observed them in Sandnet, the presented reaction times are lower bounds. We therefore obtained pDNS records and VirusTotal [34] analysis results to investigate the history of each domain. In total, we could aggregate pDNS records for 81.3% of all domains and obtained information from VirusTotal for 98% of the domains.

We determined the reaction times for each combination of public blacklist and malware family. Yet, for reasons of brevity we focus on a few interesting combinations only. Figure 1 illustrates a CDF of the reaction times of four blacklists, respectively, blacklist combinations. The y-axis shows the reaction time per blacklist entry in days and the x-axis depicts the ratio of domains with this reaction time. Negative y-values indicate that the domain was first seen in the blacklists and then observed in Sandnet, pDNS, or VirusTotal. Positive y-values denote that a blacklist lagged behind. The y-values of blacklisted domains that are not found in pDNS or VirusTotal are set to the negative infinity.

The black solid line represents the reaction time of the blacklists provided by `Abuse.ch` (*Palevo*, *SpyEye*, and *ZeuS*) and the corresponding domains as seen in Sandnet. About 23.3% of the domains were listed by the blacklists before they appeared in Sandnet, respectively, 76.7% of the domains were seen in Sandnet first. As depicted by the black dotted line, we find 37.9% of the domains to be blacklisted before appearing in VirusTotal. Approximately 64.7% of the domains were seen in Sandnet and added to the blacklists on the same day. The reaction time of `Abuse.ch` was less than a week for 80.2% of the Sandnet domains and the blacklists included already 96.6% of the domains within 30 days. The results show an adequate reaction time for the `Abuse.ch` blacklists, although the completeness is not ideal (cf. Section 5.2). The black dashed line illustrates the results obtained for the `Abuse.ch` blacklists and pDNS. We could not obtain pDNS records for 27.6% of the domains, i.e., these domains, although monitored in multiple sandbox environments, were never seen in the *DNS DB* database. Another 3.4% of the domains were blacklisted before the domains appeared in pDNS, while 10.4% of the domains were blacklisted and seen in pDNS on the same day. The remaining 58.6% of the domains were seen in pDNS on average 334 days before appearing in the blacklists. These domains either performed malicious activities before becoming blacklisted or—more likely—performed benign actions before turning malicious.

We observe different results for the reaction times of the other three blacklists shown in the graph. The reaction time of *UrlBlacklist* was higher than a month for 53.5% of the domains. Similarly, the blacklist *MW-Domains* has a reaction time of at least 30 days for 39.7% of the domains. After four months, the coverage of all three blacklists was still below 90%. In general, the low number of domains that appeared in Sandnet *after* they were blacklisted (negative y-values) indicates that our ground truth dataset is up-to-date.

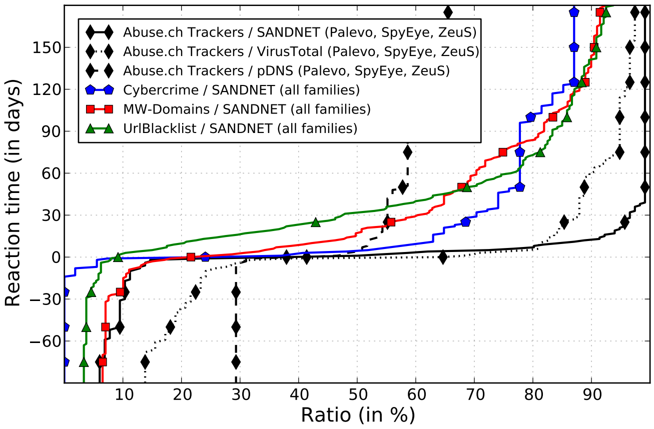

Fig. 1. Reaction times of selected blacklists

5.4 DGA-based Domains

Malware that employs DGAs to dynamically create domains—typically derived from the current date—imposes additional difficulties to blacklist operators. First, DGA-based domains are valid for a limited time span, thus often change. Ideally, blacklists would include these domains *before* they become valid. Second, most of the domains are never registered or seen active, e.g., when dynamically analyzing malware samples. Yet, DGA-based malware is on the rise [3], hence networks protected by blacklists would benefit from DGA-based listings.

We evaluate the coverage of DGA-based domains in the blacklists for five prevalent malware families. We implemented the DGAs for these families after obtaining the algorithms from partners or using reverse engineering. Four families generate domains every day, whereas the *ZeuS P2P* domains are valid for 7 days. We again measure the completeness and determine the reaction time for each family, i.e., how many days it takes to blacklist a domain once it becomes valid. We further estimate the rate of registered domains in S_{Mal} by leveraging the recorded DNS data (i.e., we check if the domains resolved to IP addresses at the time the domains were valid). As the dataset S_{Mal} contains all the domains that were listed by *any* of the 15 public blacklists at some point in time since 2012, it should also include DGA-based domains that were valid in the past.

Table 6. Coverage of DGA-based domains

	S_{Mal}				S_{AV}		
		Ratio (%)				Ratio (%)	
Family	# Domains	Listed	Reg.	t_{React}	# Domains	Listed	Reg.
Bamital	84,136	21.9	11.7	-1	104	75.0	50.0
Conficker	7,354,415	1.6	0.2	1	50,500	94.3	4.8
Flashback	4,045	18.0	15.3	71	5	0.0	100.0
Virut	8,089,752	0.2	0.004	13	10,000	97.9	1.7
ZeuS P2P	131,000	28.9	0.2	1	1,000	99.5	4.5

Table 6 illustrates the listing behavior we monitored in the period Jan. 2012 to Mar. 2014 for the public blacklists (first major column) and on a typical weekday in Mar. 2014 for the AV vendor blacklists (second major column). In total, less than 1.2% of all domains were listed by the public blacklists. On the positive side, blacklists have a low reaction time for three families (if they blacklist a domain). On the downside, 82.1% of the matches are found in the blacklist *Virustracker* only. When removing *Virustracker* from S_{Mal}, the reaction times increase significantly for most families (i.e., *Bamital*: 12 days, *Conficker*: 12 days, *Flashback*: 381 days, *Virut*: -271 days, and *ZeuS P2P*: 16 days). Before removing *Virustracker* from S_{Mal}, we find 0.2% of the *Virut* domains to be blacklisted. After removing *Virustracker*, we find merely 167 domains (0.002%) to be listed. Due to the generic structure of *Virut* domains, we assume that these domains are not listed to protect against *Virut* in particular but rather because they were related to other malicious activities. The reaction time confirms our assumption as it is not reasonable to blacklist domains 271 days before they become active for a single day.

We also determine the coverage of the AV vendor blacklists regarding DGA-based domains. To avoid requesting millions of domain names, we divide our analysis into two steps. To measure if the blacklists protect against threats of DGA-based domains that are currently active, we request listing information and DNS `A` records for all the domains that are valid on the day we perform this experiment (i.e., 03/24/2014). Second, we analyze if blacklists include domain names which become active in the future. We thus also request a sample of DGA-based domain names (i.e., a random selection of at most 10 domains per day and malware family, respectively, type for *Conficker*) that will be valid between 03/25/2014 and 04/24/2014, i.e., up to 31 days ahead of the day of requesting.

For the domains valid on the day of performing our experiment, we find 76.1%, respectively, 28.9% of the *ZeuS P2P* domains to be blacklisted by *McAfee* and *Bitdefender* and observe the best coverage for *Norton* as most of the results in Table 6 are caused by this blacklist—with a single exception. *Norton* lists 95.5% of the *ZeuS P2P* domains while the union of all AV vendor blacklists increases the coverage up to 99.5%. For the remaining blacklists and malware families, we find a negligible number of listed domains (if domains are listed at all). When taking a closer look at the registered domains that day, we find half of the *Bamital* domains and most of the domains for *Conficker B/C* and *Flashback* to be sinkholed. Further on, four domains of *ZeuS P2P* are sinkholed while the 168 registered *Virut* domains are associated to parking providers and benign web pages. In conclusion, a partition of valid domains is sinkholed by security researchers, yet the remaining domains could be used for malicious activities. We thus recommend to blacklist each DGA-based domain for security reasons (i.e., to trigger alerts). For the domains getting active in the near future, we again find the blacklist provider *Norton* to perform best. Except for *Flashback* (no listed domain) and *Bamital* (coverage of 46.5%), we find *Norton* to include at least 94.5% of the domains for each of the remaining families. For the other families and blacklists, we again observe a negligible number of listed domains.

Our analysis shows that as of today, only one blacklist can reasonably protect against any of the five DGAs used in our experiments. This is surprising to us, given the fact that—once the DGA is known—the DGA-based domains can be accurately predicted unless there are external dependencies (e.g., DGAs utilizing lists of popular feeds from social network web pages). One of the reasons could be that DGAs are often used as a C&C backup mechanism only. For example, *Zeus P2P* uses a DGA only if its peer-to-peer communication fails [35]. Another reason could be that DGA-based domains may, by coincidence, collide with benign domains. Still, as these issues can be overcome, the potential of including DGA-based domains is unused in most of the nowadays blacklists.

6 Discussion and Future Work

We showed that our parking detection approach can effectively distinguish parked and benign domains. As our features depend on the content delivered by parking services such as sponsored ads, domain resellers serving benign content and parked domains exhibiting parking behavior different from the expected however cannot be effectively identified by our detection model. This is particularly problematic when parking providers block us, e.g., for sending too many requests. Parking services employ different types of blocking (e.g., provide error messages, benign content, or the parking page template without any referral links). To avoid getting blocked, we could distribute the requests to several proxy servers or rate-limit our requests. Further, domains might perform cloaking [36], i.e., provide malicious content for real users while serving parking content for automated systems. We leave a detection of cloaking domains for future work and acknowledge that a large number of parked domains alone does not necessarily imply that a blacklist is not well-managed. We also have to keep in mind that the parking IDs might be biased in respect to the language of the blacklist content, as we obtained the IDs by leveraging the NS used by the blacklisted domains.

However, our dataset does not include any national blacklists, which primarily list domains of a specific country or language. While performing the manual verification for the real-world datasets in Section 3.3, we monitored many domains providing content in foreign languages that were flagged as parked by the classifier. This shows that our approach is largely language-independent.

The proposed sinkhole detection method relies on the blacklists to observe behavior that can be attributed to sinkholes. As such, our detection capabilities are limited to sinkholes that are blacklisted. We could use the identified sinkhole dataset as ground truth and leverage techniques such as passive DNS analysis to identify further potential sinkholes [37]. Additionally, the quality of our approach depends on the accuracy of the blacklists. If blacklists contain too many benign domains that cannot be filtered, e.g., by removing Alexa Top 25k, parking, and shared hosting IPs, we might flag benign IP addresses as potential sinkholes.

Our evaluation on the completeness of blacklists is limited to estimating lower bounds as SANDNET only covers a random subset of all samples of the active malware families. Consequently, we may have missed malicious domains in SANDNET. We aim to scale up malware execution to achieve a higher coverage.

We classified the blacklist content as parked, sinkholed, or unregistered and analyzed the completeness of the blacklists in regards to domains of various malware families. Yet, the blacklists also include domains we could not classify accordingly, leaving 23.7% of the currently blacklisted domains to be unspecified. These domains might also include potential false listings, e.g., caused by erroneous setups of analysis back-ends or insufficient verification of domains that are flagged to be potentially malicious. False listings, however, are hard to identify as each blacklist applies its own listing strategy and might include domains of malware families that are not present in SANDNET and the DGA-based domain dataset. Analysis techniques to identify potential false listings thus require a thorough evaluation of correctness. We leave the categorization of the so far unclassified domains for future work.

7 Related Work

The effectiveness of malware blacklists is still largely unstudied. In prior work, we proposed a system to track blacklists and presented first details regarding blacklist sizes [6]. With this paper, we extend our work and evaluate malware blacklist effectiveness—motivated by promising results others reported with blacklists in a different context. For example, Thomas *et al.* [38] looked at blacklists in Twitter. Similarly, Sinha *et al.* [39], Rossow *et al.* [40], and Dietrich *et al.* [41] evaluated the strength of blacklists in the context of email spam, while Sheng *et al.* [42] analyzed phishing blacklists.

Concurrent to our sinkhole identification work, Rahbarinia *et al.* developed a system called SinkMiner [37] to identify sinkhole IPs. They leverage pDNS data and *a priori* information about sinkholes to extrapolate to other sinkholes. Our approach does not rely on an initially-known set of sinkholes and, in its simple form, works without pDNS. In addition, we found sinkholes which were not linked to other sinkholes—many of which SinkMiner would miss. Nevertheless, a combination of SinkMiner and our graph-based approach could identify yet unknown sinkholes, as SinkMiner analyzes the global history of domains using pDNS while we are limited to the history of blacklisted domains. We further proposed a more advanced mechanism to identify parking providers. Rahbarinia *et al.* filter for NS that include the term *park* in their hostnames. Yet, of the 204 parking NS identified in Section 3.1 we find 59 NS to not specify this term in their hostnames. Halvorson *et al.* [23,24] identify parked domains by applying regular expressions to the aggregated web content. Instead, we introduced characteristic features for parking behavior and—to the best of our knowledge—are the first to propose a generic mechanism to identify parked domains.

Orthogonal to our work, a number of proposals aim to increase the quality of existing blacklists. Neugschwandtner *et al.* [43] proposed Squeeze, a multi-path exploration technique in dynamic malware analysis to increase the coverage of C&C blacklists. Stone-Gross *et al.* [44] proposed FIRE, a system to identify organizations that demonstrate malicious behavior by monitoring botnet communication. Our findings show that usage of such systems should be fostered.

8 Conclusion

We have shown that blacklists have to be employed with care as the nature of the listings is diverse. First, one needs to keep in mind that also sinkholes may be blacklisted. Second, many parking providers re-use popular malware domains. This is crucial to know, e.g., when blacklists raise false positives or one aims to attribute a reputation to certain providers based on blacklist data. In addition, our evaluation of blacklist coverage indicates how blacklists can be improved in the future as none of the public blacklists is sufficiently complete to protect against the variety of malware threats we face nowadays. We further have shown that most blacklists operated by AV vendors do not cover DGA-based malware to effectively protect users, although integration would be straight-forward. We are confident that our analyses will help to improve blacklists in the future.

9 Acknowledgment

We would like to thank our shepherd Manos Antonakakis for his support in finalizing this paper. We also would like to thank the anonymous reviewers for their insightful comments. This work was supported by the German Federal Ministry of Education and Research (Grant 01BY1110, MoBE).

References

1. Kolbitsch, C., Livshits, B., Zorn, B., Seifert, C.: Rozzle: De-Cloaking Internet Malware. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy. SP '12, Washington, DC, USA, IEEE Computer Society (2012) 443–457

2. Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou, II, N., Dagon, D.: Detecting Malware Domains at the Upper DNS Hierarchy. In: Proceedings of the 20th USENIX Conference on Security. SEC'11, Berkeley, CA, USA, USENIX Association (2011) 27–27
3. Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., Dagon, D.: From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In: Proceedings of the 21st USENIX Conference on Security Symposium. Security'12, Berkeley, CA, USA, USENIX Association (2012) 24–24
4. Bilge, L., Kirda, E., Kruegel, C., Balduzzi, M.: EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis. In: 18th Annual Network and Distributed System Security Symposium, San Diego, US, The Internet Society (2011)
5. Rossow, C., Dietrich, C., Bos, H.: Large-Scale Analysis of Malware Downloaders. In: Proceedings of the 9th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA'12, Berlin, Heidelberg, Springer-Verlag (2013) 42–61
6. Kührer, M., Holz, T.: An Empirical Analysis of Malware Blacklists. Praxis der Informationsverarbeitung und Kommunikation **35**(1) (2012) 11–16
7. Microsoft Corp.: Citadel Botnet. `http://botnetlegalnotice.com/citadel` (2014)
8. Abuse.ch Malware Trackers. `http://www.abuse.ch/` (2014)
9. CyberCrime Tracker. `http://cybercrime-tracker.net` (2014)
10. Malc0de.com. `http://malc0de.com/` (2014)
11. Malware Domain List. `http://www.malwaredomainlist.com/` (2014)
12. Malware-Domains. `http://www.malware-domains.com/` (2014)
13. Shadowserver: Botnet C&C Servers. `http://rules.emergingthreats.net` (2014)
14. Shalla Secure Services. `http://www.shallalist.de/` (2014)
15. URLBlacklist. `http://urlblacklist.com/` (2014)
16. Kleissner & Associates. `http://virustracker.info/` (2014)
17. Bitdefender TrafficLight. `http://trafficlight.bitdefender.com/` (2014)
18. BrowserDefender. `http://www.browserdefender.com` (2014)
19. McAfee SiteAdvisor. `http://www.siteadvisor.com/` (2014)
20. Norton Safe Web. `http://safeweb.norton.com/` (2014)
21. Kührer, M., Rossow, C., Holz, T.: Paint it Black: Evaluating the Effectiveness of Malware Blacklists. Technical Report HGI-2014-002, University of Bochum - Horst Görtz Institute for IT Security (June 2014)
22. Li, Z., Alrwais, S., Xie, Y., Yu, F., Wang, X.: Finding the Linchpins of the Dark Web: A Study on Topologically Dedicated Hosts on Malicious Web Infrastructures. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy. SP '13, Washington, DC, USA, IEEE Computer Society (2013) 112–126
23. Halvorson, T., Szurdi, J., Maier, G., Felegyhazi, M., Kreibich, C., Weaver, N., Levchenko, K., Paxson, V.: The BIZ Top-Level Domain: Ten Years Later. In: Proceedings of the Passive and Active Network Measurement Workshop. Lecture Notes in Computer Science, Berlin, Heidelberg, Springer-Verlag (2012) 221–230
24. Halvorson, T., Levchenko, K., Savage, S., Voelker, G.M.: XXXtortion?: Inferring Registration Intent in the .XXX TLD. In: Proceedings of the 23rd International Conference on World Wide Web. WWW '14, Geneva, Switzerland, International World Wide Web Conferences Steering Committee (2014) 901–912
25. Farsight Security, Inc.: DNS Database. `https://www.dnsdb.info/` (2014)
26. Alexa Internet, Inc.: Top 1M Websites. `http://www.alexa.com/topsites/` (2013)
27. Damerau, F.J.: A Technique for Computer Detection and Correction of Spelling Errors. Commun. ACM **7**(3) (1964) 171–176

28. RapidMiner, Inc. `http://rapidminer.com/` (2014)
29. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel Methods in Machine Learning. Annals of Statistics **36** (2008) 1171–1220
30. Rossow, C., Dietrich, C.J., Kreibich, C., Grier, C., Paxson, V., Pohlmann, N., Bos, H., van Steen, M.: Prudent Practices for Designing Malware Experiments: Status Quo and Outlook. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy. SP '12, San Francisco, CA, IEEE Computer Society (2012)
31. Rossow, C., Dietrich, C.J., Bos, H., Cavallaro, L., van Steen, M., Freiling, F.C., Pohlmann, N.: Sandnet: Network Traffic Analysis of Malicious Software. In: Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security. BADGERS '11, NY, USA, ACM (2011) 78–88
32. Dietrich, C.J., Rossow, C., Pohlmann, N.: *CoCoSpot*: Clustering and Recognizing Botnet Command and Control Channels using Traffic Analysis. Comput. Netw. **57**(2) (2013) 475–486
33. Rossow, C., Dietrich, C.J.: ProVeX: Detecting Botnets with Encrypted Command and Control Channels. In: Proceedings of the 10th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA). Volume 7967 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2013) 21–40
34. VirusTotal. `http://www.virustotal.com/` (2014)
35. Rossow, C., Andriesse, D., Werner, T., Stone-Gross, B., Plohmann, D., Dietrich, C.J., Bos, H.: P2PWNED: Modeling and Evaluating the Resilience of Peer-to-Peer Botnets. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy. SP '13, Washington, DC, USA, IEEE Computer Society (2013) 97–111
36. Kolbitsch, C., Livshits, B., Zorn, B., Seifert, C.: Rozzle: De-cloaking Internet Malware. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy. SP '12, Washington, DC, USA, IEEE Computer Society (2012) 443–457
37. Rahbarinia, B., Perdisci, R., Antonakakis, M., Dagon, D.: SinkMiner: Mining Botnet Sinkholes for Fun and Profit. In: 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats, Berkeley, CA, USENIX (2013)
38. Thomas, K., Grier, C., Ma, J., Paxson, V., Song, D.: Design and Evaluation of a Real-Time URL Spam Filtering Service. In: Proceedings of the 2011 IEEE Symposium on Security and Privacy. SP '11, Washington, DC, USA, IEEE Computer Society (2011) 447–462
39. Sinha, S., Bailey, M., Jahanian, F.: Shades of Grey: On the effectiveness of reputation-based "blacklists". In: Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on. (2008) 57–64
40. Rossow, C., Czerwinski, T., Dietrich, C.J., Pohlmann, N.: Detecting Gray in Black and White. In: MIT Spam Conference. (2010)
41. Dietrich, C.J., Rossow, C.: Empirical Research on IP Blacklisting. In: Proceedings of the 5th Conference on Email and Anti-Spam, CEAS. (2008)
42. Sheng, S., Wardman, B., Warner, G., Cranor, L.F., Hong, J., Zhang, C.: An Empirical Analysis of Phishing Blacklists. In: Proceedings of the Sixth Conference on Email and Anti-Spam. (2009)
43. Neugschwandtner, M., Comparetti, P.M., Platzer, C.: Detecting Malware's Failover C&C Strategies with Squeeze. In: Proceedings of the 27th Annual Computer Security Applications Conference. ACSAC '11, NY, USA, ACM (2011) 21–30
44. Stone-Gross, B., Kruegel, C., Almeroth, K., Moser, A., Kirda, E.: FIRE: FInding Rogue nEtworks. In: Proceedings of the 2009 Annual Computer Security Applications Conference. ACSAC '09, Washington, DC, USA, IEEE Computer Society (2009) 231–240