Practical Attacks Against DNS Reputation Systems

Tillson Galloway[*], Kleanthis Karakolios[*],
Zane Ma[†], Roberto Perdisci[‡*], Angelos Keromytis[*], Manos Antonakakis[*]
[*]*Georgia Institute of Technology*, [†]*Oregon State University*, [‡]*University of Georgia*

Abstract—**DNS reputation systems are a critical layer of network defense that use ML to identify potentially malicious domains based on DNS-related behaviors. Despite their importance in protecting against spam, malware, and social engineering, little is known about the adversarial robustness of real-world DNS reputation systems. This work takes a first look at general attacks against DNS reputation systems. To overcome the black-box setting of deployed DNS reputation systems, we begin by creating an open-source reference DNS reputation system that 1) overcomes common pitfalls in data collection, preprocessing, training, and evaluation found in prior work, 2) approximates DNS reputation systems from prior research, and 3) enables future reproducible research. We find that general adversarial ML techniques are impractical due to a highly constrained input space, complex feature interdependencies, and difficult inversion from feature vectors to raw input samples. We then implement two classes of practical attacks, mimicry and popularity manipulation, that achieve high success rates against both our reference model and a popular commercial DNS reputation system, highlighting the transferability of the attacks to the real world. Finally, we develop constraint models that assess the time and financial cost required to execute our attacks. Using these models, we demonstrate that an adversary with $10 can evade a leading security vendor with a 100% success rate in two weeks.**

1. Introduction

The Domain Name System (DNS) is the ubiquitous network protocol for mapping semantic domain names to resource records (RRs) containing important information such as a domain's IP address. Internet communication (e.g., web, email, etc.) often begins with a DNS request, making DNS a useful layer for monitoring network traffic. Benign and malicious actors often differ in their DNS configuration, leading to the development of DNS reputation systems [3], [4], [10], [50], [36], [81] that utilize machine learning (ML) to accurately detect suspicious domains and, blocking network attackers at the DNS chokepoint. Commercial DNS reputation systems are widely deployed for combatting malicious activities such as spam and malware [21], [80], [88], [60].

Like all ML-based security systems, DNS reputation systems must account for adversarial circumvention, where an attacker renders the ML model ineffective by introducing either false negatives or false positives. However, despite the broad deployment and importance of DNS reputation systems, prior examination of adversarial robustness is qualitative and unsystematic. This is due to challenges surrounding 1) accessibility of data and reference DNS reputation models, and 2) the relative complexity of ML pipelines for DNS reputation.

This work presents the first quantitative and systematic study of adversarial attacks against DNS reputation systems. To address the challenge of accessibility, we begin by building a DNS reputation model based on crowd-sourced threat feeds and open-source threat intelligence datasets. Our reference model utilizes a wide range of features that broadly represent reputation systems that operate on DNS data from recursive resolvers. We then release our data collection pipeline, features, and model to eliminate accessibility challenges for future researchers.

The second challenge facing adversarial evaluation of DNS reputation systems is that conventional adversarial ML (AML) techniques do not apply. Most AML techniques [16], [85], [45], [34] manipulate the feature space to generate adversarial feature vectors. Unfortunately, DNS reputation systems rely extensively on contextual data to generate features, so the problem space (i.e., possible real-world inputs) to feature space mapping is both non-invertible and non-differentiable. This makes it difficult to convert perturbed feature vectors back into real-world DNS traffic. Furthermore, the scale of contextual feature extraction data precludes existing adversarial techniques that work directly in the problem space [77].

Finally, although successful adversarial attacks may be *possible* in the real world, not all of them are *practical*, since they may still face external constraints that make them unrealistic (e.g., using expensive server hosting providers). To bridge this gap, we evaluate the primary practical constraints bounding an attacker: time and financial cost. We ultimately find that DNS reputation models are highly susceptible to practical adversarial attacks, and our analysis suggests directions for fortifying existing models.

To overcome these challenges, we craft a new set of mimicry attacks that operate directly in the problem space and account for contextual feature extraction processes. Mimicry attacks attempt to mirror the domain behavior of other domains, which can lead to inducing false positives (evasion) or false negatives in the model (sabotage). To study these attacks systematically, we perform mimicry across several different feature classes (e.g., infrastructure, popularity, etc.) and find that such attacks can achieve up to 100% evasion rate against our reference DNS reputation system and a 100% evasion rate against the reputation system deployed by a leading security vendor (SV1). In both cases, these attacks cost attackers under $10 for an entire campaign and require only two weeks of effort. Importantly, we perform all evasion attacks while preserving the semantics of the malicious domain (i.e., its resolution to malicious infrastructure). We also achieve targeted, false positive-inducing sabotage attacks that can result in a denial of service for benign domains, and find that this is effective against SV1's system.

Overall, this work answers the following research questions:

- **RQ1:** To what extent are DNS reputation systems vulnerable to adversarial attacks, and how do these vulnerabilities manifest in operational systems?
- **RQ2:** Under what conditions can a *practical* adversary with limited knowledge of a model's dataset and feature extraction process perform these attacks?
- **RQ3:** How do different feature classes or preprocessing steps contribute to the susceptibility of DNS reputation systems to adversarial attacks?
- **RQ4:** What steps can be taken to mitigate these attacks?

In answering these questions, we make the following contributions:

- We perform the first adversarial analysis of DNS reputation systems that utilize a large contextual dataset for feature computation during both model training and testing/inference. To ensure practicality, we perform attacks directly in the problem space to work around the non-invertible and non-differentiable feature mapping of DNS reputation models.
- In collaboration with a popular security vendor, we assess the resilience of both self-implemented and commercial DNS reputation systems against evasion techniques. Our findings reveal that seemingly simple attacks can achieve success rates as high as 100%.
- We apply novel input constraints that help to distinguish *possible* attacks from *practical* attacks along two dimensions: time and financial cost.
- We highlight assumptions made in prior academic work in DNS reputation systems that may leave reproductions vulnerable to adversarial attacks, then guide future work by providing recommendations and implementations of improvements that can be made in models' training processes[1].

2. Background

2.1. Domain Name System

The Domain Name System (DNS) maps a DNS name to its corresponding resource record(s), which are frequently type A/AAAA records containing IPv4/IPv6 addresses. To resolve a DNS name, a host queries a recursive DNS server, which iteratively traverses the hierarchy of authoritative name servers (NS), starting from the global DNS root servers, then the domain's TLD zone NS, and ultimately ending at the authoritative NS that contains the requested record. We refer to a full domain name (e.g., `a.b.ieee.org`) as the *fully qualified domain name (FQDN)*. The publicly registrable portion of a domain (e.g., `ieee.org`, `bbc.co.uk`, or `example.s3.amazonaws.com`) is called the *effective* second-level domain, and the domain suffix is called the effective top-level domain. In this work, we will use TLD to refer to the effective TLD, 2LD to refer to the effective second-level domain (also commonly referred to as SLD), and 3LD to refer to the effective third-level domain. On occasion, we will depart from this convention to refer to the *simple* TLD of a domain name (e.g., `.uk` is the simple TLD of `bbc.co.uk`).

Active DNS Lookups Active DNS lookups (**aDNS**) can be performed on domain names extracted from public datasets, including zone files[2] and public blocklists [44], [87]. Active DNS datasets are limited by the scope of public data: TLD-level zone files only contain a domain's 2LD, and CT logs only include websites with valid SSL certificates, which misses non-HTTPS names and names hidden by wildcard certificates.

Recursive DNS Traffic Recursive resolvers are a common source of passively collected DNS records (**pDNS**). Data collection occurs either "above" the recursive resolver (between the recursive resolver and authoritative DNS servers) or "below" the recursive (between the recursive resolver and its clients). In either case, a recursive DNS perspective provides data for a range of domains, depending on the number and composition of hosts that utilize the recursive resolver(s). Recursive resolvers can be deployed for a private network, or they can be publicly deployed as "open" recursive resolvers.

Authoritative DNS Traffic Authoritative DNS servers possess deep visibility into all DNS traffic within the zones that the server controls. For a TLD authoritative NS, this includes full temporal and client recursive visibility, but also client subnet visibility in some instances [29]. Obtaining access to this data is challenging as popular NS are operated by a handful of TLDs and domain registrars.

2.2. DNS Reputation Systems

DNS reputation systems are machine learning (ML) systems that assign a reputation score to an input domain name, where higher reputation scores reflect legitimate Internet services and low reputation maps to domains associated with malicious activities. Reputation scores ultimately correspond to either binary (i.e. malicious/benign) or multiclass (e.g., C2, phishing, spam, benign, etc.) labels. DNS reputation systems are regularly retrained and can identify new, emergent malicious domains before they appear on static blocklists. DNS reputation systems model domain behavior using a variety of data sources: aDNS [66], pDNS for recursive resolvers [3], [10], [70], [53], [19], [66] or authoritative nameservers [4], or DNS registrars [36], [81]. Many forms of machine learning have been applied to DNS reputation systems: supervised ML [3], [10], neural networks [50], unsupervised ML [61], [20], [5], [74], and graph algorithms [55], [42]. Several works have developed specialized DNS reputation systems that identify specific forms of DNS abuse, such as domain generation algorithms (DGA) [74], [5], [75], [20], malicious flux service networks [61], [62], [37], or domains used by particular malware families [66]. This

1. https://github.com/Astrolavos/dns-reputation-system-sp2024

2. Country code TLDs (e.g., .eu, .tv, etc.) are controlled by the authoritative country and largely unobtainable to the public.

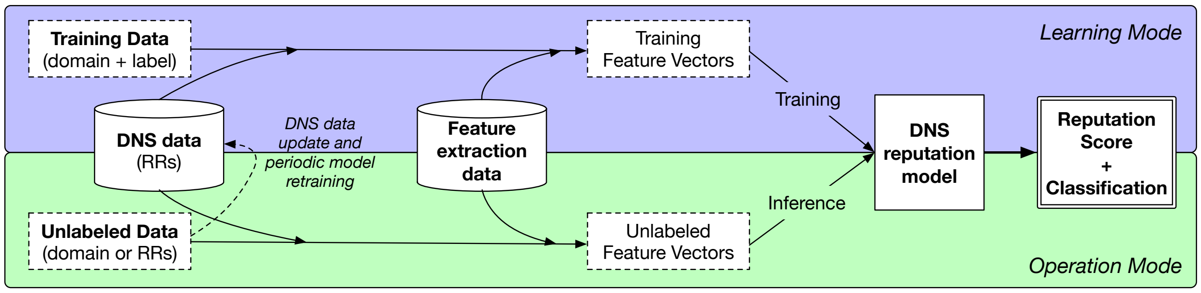


Figure 1: **DNS reputation system overview**—A pDNS-based model is shown. Training and inference both rely on shared, contextual feature extraction and DNS datasets that are updated and used for periodic model retraining.

work considers adversarial attacks against classifiers that operate on recursive DNS traffic and active DNS lookups to detect emerging and existing families of attacker-owned malware and spam domains. Appendix A1 summarizes and categorizes existing DNS reputation systems according to their *network visibility* and their *type of modeling*.

2.2.1. Statistical Classifiers. Classification-based reputation systems generate a set of statistical features for a domain and directly classify it as malicious or benign. Although decision trees (random forests, J48) are the most common classification algorithm [3], [10], [4], [70], [36], [19], there is also recent work in using deep learning (e.g., neural networks, LSTMs) for classification [50].

2.2.2. Semi-Supervised Classifiers. These reputation systems first use unsupervised learning to *cluster* similar domains and then apply statistical ML to *classify* each cluster as malicious or benign. Most systems cluster domains based on similarity in their resolved IPs, but Pleiades [5] performs two alternative forms of clustering: the first is based on lexical features, and the second is spectral clustering based on hosts that query the domains.

2.2.3. Graph-based Models. Graph-based reputation systems model DNS traffic as a bipartite *domain resolution graph*, $G = (V_D, V_I, E)$ where the vertices in V_D are domains, the vertices in V_I are IP addresses, and $(d, i) \in E$ if domain name d has resolved to IP i. Khalil et al. [42] proposed a *domain graph*, $\hat{G} = (D, \hat{E})$ where all D are domain names and $(d_1, d_2) \in \hat{E}$ if domain names d_1 and d_2 have been resolved to the same IP. In this construction, the weight of the edges is proportional to the number of shared IPs between two domains. Using these graphs, a domain's reputation is based on its nearest malicious/benign neighbors, as well as belief propagation methods [55].

2.2.4. Training Process. DNS reputation systems differ from conventional ML pipelines (e.g., classification of executable binaries or images). As depicted in Figure 1, DNS reputation systems have two distinct modes: a learning mode and an operation mode. For both phases, the features are not computed directly from a single input sample. Rather, feature values incorporate several metadata sources, including popularity lists, WHOIS/IPWHOIS records, and characteristics of nearby domains. Features are highly contextual and interdependent, which makes adversarial DNS reputation systems poor candidates for reverse feature-mapping analysis [65]. For example, one class of features in pDNS-based systems is IP-related historic domain names (IP-RHDNs, detailed in Section 4). IP-RHDNs contain the set of all domains that point to a domain's historically resolved IPs, including unlabeled, non-training domains. During model training, a snapshot of the DNS database is used to compute labeled features and generate a reputation model. The underlying database continues to update after the model is trained, so feature computation for a given domain d potentially changes every time a new RR is added to the DNS data source. This functionality is intentional, as DNS reputation systems must constantly adapt to the rapidly evolving Internet.

2.2.5. Real-world Deployments. Security organizations deploy operational DNS reputation systems to curb malicious domain usage either through DNS-based blackhole lists (DNSBLs) ([41], [58]) or at the registry level ([69], [81]). Automated systems are often paired with expert analysis for false positive reduction and human-in-the-loop learning [58]. These services vary in their target subclass of malicious domains: for example, Nordspam [58] and Invaluement [41] focus on spam domains, DGArchive [33] exclusively covers randomly generated domains (DGA) used in malware, and ThreatLog [86], Premadoma [81] and Radix [69] cover generic malicious content, including malware, spam, and phishing domains. These systems also vary in their data collection processes. For example, registry-level reputation systems may have unfettered access to authoritative DNS data and registration information, while malware reputation systems gather domains by executing malware in a sandbox [86]. In this paper, we focus on DNSBL-based lists that gather domains through a malware sandbox.

2.3. Attacks on DNS Reputation Systems

Adversarial ML aims to modify samples either at training time (poisoning [83]) or at inference/test time (evasion [9]) to compromise the confidentiality, integrity, or availability of an ML model. Most adversarial ML research directly manipulates the feature space ([85], [45], [34], [27], [16]) to craft samples of a particular class. However, these techniques are not applicable when the feature space is non-invertible/non-differentiable [65] and adversarial feature vectors cannot be converted into realistic input samples. DNS reputation systems have complex mappings from problem space (real-world inputs) to feature space–their features capture aggregate statistics about network infrastructure, domain popularity, lexical features, and resolution behavior. Sheatsley et al. proposed a general system to infer the constraints of the problem space [77] to generate practical adversarial examples against gradient-based classifiers. Unfortunately, the computational intensity of computing contextual DNS reputation features limits the utility of this technique.

An alternate path for identifying attacks is to manipulate the problem space directly. Pierazzi et al. [65] outlined two generic search strategies to create adversarial samples in the problem space. The first relies on local gradient approximation to inform input mutations, while the second approach creates an approximate inverse of feature mapping. Both of these approaches require white box access to a model's training set and parameters. Our paper focuses on a black-box threat model, where this level of access is unavailable. Instead, we introduce a domain-specific technique on DNS reputation systems based on the framework for problem space attackers [65], which defines semantic-preserving transformations in the problem space as functions that preserve semantics, are plausible, and are robust to pre-processing techniques. Our study identifies specific real-world transformations and provides a model to strategically compose them while being constrained by semantics, time, and financial cost. Our approach is broadly related to AML attacks against malware classification, which aim to disguise a malicious sample as a benign sample while preserving malicious functionality. The proposed attack strategies fall into three general categories: (i) camouflage attacks ([26], [72], [40]) where the attack tries to conceal suspicious characteristics of the malicious sample through obfuscation and encryption, (ii) noise addition attacks ([17], [65], [73], [15]), and (iii) changing the nature of the malicious sample ([17], [40], [64], [84]).

Prior studies of adversarial robustness for DNS reputation systems are largely qualitative and ad hoc. In their registration-time malicious domain detector, Hao et al. [36] mention the financial cost of evasion by noting that high-reputation registrars cost more than low-reputation registrars. They further evaluate model robustness by comparing model performance after removing certain feature groups. Desmet et al. [81] discuss three evasion patterns for registration-based systems (e.g., temporally random actions, dormant periods, and excess registration) and argue that these behaviors all increase cost to the attacker. Le Pochat et al. [66] propose strategies to evade lexical, popularity, TLS, and time-based features, but qualitatively conclude that they require high financial and management investment in an individual domain, which malware operators tend to avoid due to fast blocklisting. Our work invalidates many of these assumptions for reputation systems based on recursive DNS and demonstrates that the financial and management investment is lower than previously suggested.

3. Threat Model

This section defines the attacker's goals, capabilities, and knowledge about the target system; furthermore, we introduce the financial and time constraints that influence attack feasibility.

3.1. Attacker Goals

The attacker's goal is to manipulate a target system's computed reputation score for a specific domain name, denoted as d, through changes to one or more domains under their control. We focus on domain evasion attacks, detailed below. Sabotage attacks, which cause misclassification of a benign domain as malicious, are more difficult to test ethically in deployed systems and beyond the scope of this work, although we briefly discuss them in Section 7.3. Interestingly, due to the online learning property and periodic retraining of DNS reputation systems, our inference-time attacks may also poison subsequent model updates, though we do not explore such effects in depth.

An evasion attack attempts to maximize the reputation of d such that the model misclassifies d as benign. The attacker's motivations rely heavily on the context of their domain. For example, the goals and constraints facing an attacker utilizing DNS to facilitate command-and-control (C2) operations are different than an attacker who seeks to increase the deliverability of a spam campaign. Due to the difficulty of sending spam (highlighted in Section 7.2), we primarily focus on domains used in malware campaigns. In a real-world setting, candidate domains often come from static or dynamic analysis of known malicious binaries. Many malicious binaries reach out to both benign and malicious domains, so propagating the malicious label of a binary to its associated domains is a classification problem. If an attacker can evade this system, they can avoid blocklisting their domains despite their malicious binaries being detected.

3.2. Attacker Capabilities

The attacker, who owns a domain d, can take the following actions in any order:

- Add/remove an A/AAAA record with IP address x
- Provision server with IP x on server hosting provider h
- Add an RR containing TTL value y
- Manipulate a popularity list, bringing domain to rank r
- Halt malicious activity for duration d
- Extend the domain registration length by l days

3.3. Attacker Knowledge

Real-world attacks are limited by their visibility into the data used to train the target's DNS reputation system. To address this, we examine two models of the attacker's knowledge: a gray-box model, where the attacker knows the model's architecture, training data, and testing data, and a black-box model where the attacker only has access to publicly available intelligence datasets (Table 1). In both cases, the attacker knows the model's features, which can be accomplished through analysis of prior work and

Model	Label Visibility		DNS Visibility		Practical Constraints		Model Stats	
	OSINT	SV1	aDNS	pDNS	Data Window	Data Cost	TPR	FPR
OSINT/aDNS	✓		✓		2 weeks	\$0	98.4%	2.5%
SV1/aDNS	✓	✓	✓		2 weeks	∼ \$500K	99.1%	1.3%
SV1/pDNS	✓	✓		✓	2 weeks	∼ 1M−10M	94.5%	3.7%

TABLE 1: **Reference Classifiers**—We constructed three models with increasing levels of data visibility.

by probing the model. Because DNS reputation systems heavily rely on features extracted from related DNS records, we also consider two settings for the attacker's DNS data access: the weakest attacker has access to a well-maintained aDNS dataset, and the strongest attacker has access to a large pDNS dataset. Obtaining access to these datasets varies drastically in terms of level of effort and financial cost, which we address below. Based on the performance and cost of each model, we take a middle-ground approach and perform subsequent reference model analysis using the SV1/aDNS model, which combines malicious domains from SV1's dataset and benign domains collected from OSINT with publicly accessible aDNS context.

3.4. Practical Constraints

Our attacks (Section 5) directly target the problem space of raw inputs in order to address the difficult inverse feature mapping problem of DNS reputation features. This yields attacks that are *possible*, but not necessarily *practical*, since they may still face external constraints that make them unrealistic. For example, a naive attack would be to point a malicious domain (and perform DNS resolution) at every possible IPv4 address; although this attack may evade malicious classification, it would be easy to detect and prohibitively expensive to perform. We evaluate each attack along two dimensions of practical constraints: time and financial cost. An attacker running a malicious campaign is also subject to means of detection beyond the scope of our reference model and lab evaluation, which we discuss in Section 6.4.

3.4.1. Financial Cost. When creating malicious infrastructure, an adversary may spend money on domain registration, servers, and cloud service providers. We estimate the financial cost of our attacks, which includes the cost of procuring a shadow model's ground truth data, to determine what level of adversary (e.g., low/high resource) can execute them. Table 5 provides a best-effort estimate of data costs calculated in coordination with our data partners. When considering the cost of an attack, we assume the attacker has already registered a domain for one year.

3.4.2. Time. Some features establish reputation by measuring the age of a domain or by analyzing resolution behavior over an extended period of time. An attacker's actions can be limited by the amount of time they need to invest before using a domain maliciously.

4. Methodology

Drawing on previous approaches for general malicious domain detection [3], [10], DGA detection [5], and botnet detection [66], we first build a classifier to detect malicious domains, by incorporating 43 features from 17 previous published works. Our classifier consists of four stages: data collection, feature extraction, model training, and evaluation. To promote reproducibility, we include our feature extraction code in our open-source repository.

4.1. Data Collection

Machine learning models for DNS reputation are highly data-dependent: different choices in DNS data and ground truth class labels lead to different classification results. We create a generic model for malicious domain classification with a training set consisting of spam, phishing, and malware C2 domains. We construct multiple models emulating different levels of DNS visibility possessed by operators. In particular, we compare the performance of freely available active DNS (aDNS) datasets with difficult-to-obtain passive DNS (pDNS) datasets. In either case, fully qualified domain names (FQDNs) are annotated with benign or malicious labels from a combination of popularity lists, public datasets, and public blocklists, as shown in Table 2.

4.1.1. Class Labels. Assigning malicious and benign labels to domains is a difficult task, and methodologies from prior work differ in their approaches. Individual approaches are difficult to reproduce due to their use of outdated blocklists (e.g., [3], [10] reference lists that no longer exist) or proprietary data sources (e.g., [36] collects malicious domains from a self-operated spam trap). Using a particular labeling heuristic introduces biases that alter a model's performance and attack surface. In this section, we present a methodology addressing two main challenges facing class labeling: inaccurate labels and dataset bias.

Benign data When building a set of benign domains, we must ensure that (1) the domains are truly benign and (2) the domains are representative of authentic Internet traffic. To solve (1), we apply aggressive data filtering, and to solve (2), we combine multiple domain popularity lists, which are commonly used in prior work, with a large set of unpopular domains, which are often neglected but important for a realistic evaluation of the false positives.

We source popular domains from the top 100K of the Tranco list [47], which ranks domains based on their average rank in multiple domain popularity lists and is more stable than using an individual list. To make our ground truth more robust to manipulation, we only use domains that appeared in the top 100K every day for 60 days preceding the training date (91.5K/100K domains). Additionally, we collect 118K unpopular domains related to websites listed in The Real Yellow Pages (YP) [89]. To associate a domain with a business listing, a user must verify their identity

	Data source	Date range	Size	Description
Training Labels	**Benign Ground Truth (OSINT)**			
	Tranco Popularity List [47]	8/22/2023 - 10/21/2023	100K domains	List of popular FQDNs
	Consistent 100K	8/22/2023 - 10/21/2023	91.5K domains	FQDNs consistently in Tranco 100K
	The Real Yellow Pages [89]	9/16/2023	118K domains	Source of unpopular domains
	Post-filtering	11/26/2023	84K domains	Unpopular domains after filtering
	Benign FQDNs (aDNS)	-	582K domains	Benign FQDNs appearing in aDNS
	Benign FQDNs (pDNS)	-	2B domains	FQDNs appearing in pDNS
	Malicious Ground Truth (OSINT)			
	StopForumSpam [82]	10/21/2023	40K domains	Known blog/forum spam (sampled)
	Prigent Malware List [2]	10/21/2023	40K domains	Aggregated malware (sampled)
	Phishing Army [63]	10/21/2023	40K domains	Aggregated phishing (sampled)
	Total Malicious FQDNs (aDNS)	-	120K domains	
	SV1 Ground Truth			
	Malicious FQDNs	10/21/2023	2.9M domains	
	Benign FQDNs	10/21/2023	10.5M domains	
Feature Extraction	BGP routing table [12]	10/1/2023	980K rows	Dataset of AS/BGP prefix
	AS ownership table [13]	10/1/2023	210K rows	Dataset of AS ownership relations
	WHOIS data	10/21/2023, 11/20/2023	700K records	Registration/expiration date for domains
	Firehol Level 1-3 [31]	10/21/2023	1M IPs	Public IP blocklist
	Tranco list [22]	10/21/2023	1M domains	Daily Tranco popularity list
	Public Suffix List [57]	10/02/2023	9.5K domains	List of known public suffixes
DNS Data	Passive DNS	10/7/2023 - 10/21/2023	546B RRs	pDNS data from large recursive resolver
	A Records	10/7/2023 - 10/21/2023	195B RRs	A records resolving to public IP space
	Active DNS [6]	10/7/2023 - 10/21/2023	102B RRs	Daily A record scans of public zone files
	Public IPs	10/7/2023 - 10/21/2023	2.6B RRs	Domains that resolve to public IP space

TABLE 2: **Data sources**—We combine large DNS datasets, both active (publicly shareable) and passive (proprietary), with training labels and supplemental data from primarily public/low-cost sources to build and evaluate our DNS reputation systems.

with the listing's current phone number. While YP does not mention the source of these phone numbers, they appear to be those listed in the Better Business Bureau's index [8]. As a result, these domains are unlikely to be malicious.

We found that some YP listings are stale (e.g., business closing, change of website). We further found numerous stale listings where low-reputation ad providers had purchased expired domain names to exploit residual trust [48]. We address this by removing domains with multiple historic registrars or gaps between the domain's expiration and creation dates. This data is limited to records appearing after VirusTotal began collecting WHOIS data in 2019. We then filter only domains that have *never* appeared on the Tranco 1M list. We also discard domains that do not resolve (`NXDOMAIN`/`SERVFAIL` DNS responses). To help ensure that domains are benign, we apply aggressive filtering on the domains using VirusTotal. We discard domains that have *any* malicious classification. The final list contains 84K domains. Code to generate each list is available in our open-source repository.

In both datasets, we exclude public suffixes that can be associated with both benign and malicious entities (e.g., CDNs, dynamic DNS, and free web hosting services). We determine these public suffixes using Mozilla's widely accepted Public Suffix List [57]. In total, the benign dataset contains 201K 2LDs. When annotating the DNS data, we label all the subdomains of a benign 2LD as benign, resulting in 582K benign FQDNs. We note that even if this dataset is not perfect, any mislabeled data will lead to non-optimistic suboptimal performance metrics when considering the attack's generalization to real-world models.

Malicious data Our malicious ground truth dataset consists of public domain name blocklists (PBLs) from (i) spam domains from StopForumSpam [82], (ii) malware domains from Prigent DBL [14], and (iii) social engineering domains from Phishing Army [63]. We remove domains that also appear in the benign ground truth and domains that are known to be dynamic DNS or shared hosting domains. Finally, we remove sinkhole domains from the malicious ground truth using the SinkDB dataset [1]. As an additional check, we validate that these domains are consistently labeled as malicious by more than 10 antivirus vendors on VirusTotal. After filtering, we downsampled each of these data sources so that the final malicious dataset contains balanced numbers of FQDNs in each subclass (spam, malware, and social engineering). We also evaluate a second source of malicious data: a commercially available set of malicious domains maintained by SV1.

4.1.2. DNS records. To understand our model's reliance on DNS datasets of varying visibility and financial cost, we train and evaluate performance separately against active DNS data and passive DNS data collected between October 7, 2023 and October 21, 2023.

Active DNS We use data from the Active DNS Project [44], which has continuously resolved domains from zone files, popular domain lists, and domain blocklists since 2016 [6]. This data is available to researchers at no charge [6]. As of October 23, 2023, Active DNS contains domains from 5,522 effective TLDs and complete simple 2LD coverage of 1,137 zone files. We remove A and AAAA records containing IP addresses designated as not globally reachable by IANA's special-purpose address registries ([38], [39]). Our open-source repository lists the 1,137 zones included in aDNS collection to assist in replicating our model.

Passive DNS We use 548B A/AAAA records collected passively from a North American Internet Service Provider (ISP). We operate on a daily set of non-timestamped resolution data with client IP addresses dropped. While the records in pDNS partially overlap with our aDNS dataset, our pDNS sensor does not collect our aDNS lookups.

Because pDNS is based on noisy internet traffic, the data requires additional cleaning. When cleaning the dataset, we must take care not to remove potentially malicious records. Like our aDNS filtering, we first remove A/AAAA records with IPs that are not globally reachable. This leaves over 1B reachable IP addresses, the majority of which are redundant. A record is considered redundant if many subdomains of a 2LD resolve to the same IP address. This causes a skew in the frequency of certain 2LDs in the ground truth, which biases the training and testing datasets. We filter out any FQDNs that share an IP address with more than 60 (99.5% quantile) other FQDNs under the same 2LD, and identify two main classes of redundant records: wildcard A records and benign DGA domains. A wildcard record for a domain responds with the same IP address for any subdomain queried. Automated subdomain brute-forcing results in a large number of wildcard records appearing. The most popular benign DGA, related to Microsoft Office user analytics, accounts for nearly 50% of all DNS records in our dataset.

To establish our pDNS dataset's coverage, we measure the intersection of our pDNS dataset with domains available in the Tranco list [47], which, as of October 31, 2023, aggregates DNS from globally distributed pDNS sources (Cisco Umbrella [22], Cloudflare Radar [23], and Farsight DB [28]), and non-DNS sources (Google Chrome User Experience Report [35] and Majestic [52]). We compare our pDNS dataset to the four available Tranco lists: the top 1M 2LD and FQDN lists, and the 'full' 2LD and FQDN lists, which include domains that appear in *any* of Tranco's sources. As Tranco uses simple 2LDs rather than effective 2LDs, we draw this comparison using simple 2LDs. For each list, we combine the domain names between October 7, 2023 – October 21, 2023. We find that over the same 14-day period, our pDNS dataset contains records for 96.4% of the combined top 1M 2LD list (1.05M domains), 94.1% of the combined full 2LD list (6M domains), 94.4% of the combined top 1M FQDN list (1.06M domains), and 82% of the combined full FQDN list (9.2M domains). Our pDNS dataset contains an additional 825M SLDs not in the full Tranco list.

4.2. Features

To build a generic malicious domain detector, we derive several common feature categories from prior work: lexical, popularity list, graph (RHDNs/RHIPs), evidence, registration, TTL, resource record, and temporal features. The features are listed in Table 3. Individual systems vary in feature inclusion and even implementations of the exact same feature. For example, prior work disagrees on whether to include the TLD in the computation of lexical features (e.g., [10] vs. [66]). To promote future replicability, we release Python 3.10 implementations for each feature in the camera-ready version of the paper. As explained in limitations Section 7.2, we exclude several popular feature categories, such as ones requiring active lookups of domains.

For the graph features, we model historic DNS records as a heterogeneous multipartite graph connecting domain vertices to vertices representing IPs, zones, or nameservers. Enrichment sources, such as BGP or IP blocklist datasets, add additional edges between IPs and these entities. We primarily derive features from the first and second neighborhoods of a domain.

Related historic IPs (RHIPs) A domain' set of RHIPs contains all IP addresses that a domain's A/AAAA records have pointed to since τ; in graph terms, the RHIP is the neighborhood for a given domain vertex. Each IP has a relationship to other network entities, such as ASN, BGP prefix, or AS name, from which we derive 19 features. We extract RHIP-based features from each domain's FQDN and effective 2LD and 3LD segments, as provided by the Mozilla Public Suffix List [57]. If a domain does not have a 3LD, we compute the feature based on its FQDN. *Example features: number of RHIPs, number of AS.*

IP-related historic domain names (IP-RHDNs) IP-RHDNs are the set of domains that have ever resolved to any IP in a domain's RHIP set; equivalently, this is the second (i.e., distance two) neighborhood of a target domain. When computing RHDN-based features that consider particular segments of a neighboring domain (e.g., number of distinct TLDs), we always consider the effective version of the segment, as provided by the Mozilla Public Suffix List. When computing the domain length and n-gram frequency of RHDNs, we do not consider the TLD segment (e.g., the computed length of `ieee.org` is 4). *Example features: average length of all domains sharing an IP with* d*.*

Although many count-based features can be modeled through purely graph-based techniques that consider a domain's neighborhood, others rely on node and edge features (e.g., domain age, time since first query). This disconnect implies that generic graph-based techniques [18] are an incomplete exploration of possible misclassifications; instead, we perform attacks with raw problem space inputs that realistically alter all features, since our model incorporates both graph-based and non-graph-based features.

Our popularity list features consider if a domain has achieved a certain popularity ranking. We implement this feature class using the stable Tranco list [47], which contains 2LDs aggregated from other popularity lists. When determining the rank of a FQDN, we use the rank of its 2LD.

We measure domain lifetime through the age and registration length of a domain. This data comes from WHOIS records, which are notoriously difficult to parse at scale due to their lack of schema [51]. By querying the VirusTotal API, we collected WHOIS registration/expiration dates for 80% of the 2LDs in the dataset. For the 20% of WHOIS records not in VT, we set the domain age to 0 and the registration length to 365 days, the minimum realizable values for these features.

Grouping	Feature	Used in
Lexical	Numerical char. ratio	[54], [74], [75]
	Longest word ratio	[20]
	Domain length	[36], [81], [66]
	# of trigrams	[36], [75]
	Contains "-"?	[36], [54]
	Level of subdomains	[54], [75]
	Entropy of domain	[54], [75]
Related historic IP addresses (RHIPs)	# of IPs (FQDN/3/2LD)	[3], [10], [20], [19], [61], [62], [50]
	# ASNs (FQDN/3/2LD)	[3], [19], [79]
	# BGP prefixes (FQDN/3/2LD)	[3], [19], [79]
	# AS organizations (FQDN)	[3], [19], [50]
	# AS countries (FQDN/3/2LD)	[3], [19], [50]
	# AS reg. dates (FQDN/3/2LD)	[3], [19]
	# AS registrars (FQDN/3/2LD)	[3], [19]
IP-related historic domain names (IP-RHDNs)	# of domains	[61], [62], [19]
	Domain length m/m/S	[74], [75], [19]
	{1–3}-gram freq. m/m/S	[5], [75], [19]
	# of TLDs	[74], [5], [19]
	TLD freq. m/m/S	[74], [19]
	`.com` TLD ratio	[5], [19]
Historic evidence	# of IPs in PBL	[3]
	# of BGP/AS IPs in PBL	[3], [70]
Resource records	Presence of DMARC/SPF record	
	PTR record ratio for IPs	[37], [10]
TTLs	# unique TTLs	[10]
	m/m/S of TTLs	[61], [62], [20], [37]
	Min. TTL	[79], [50]
Registration	Length of reg. period	[36], [81], [66]
	Days since reg. date	[66]
Temporal*	Time btwn first/last query	[50], [66]
	Time since last query	[50]
	# days queried	[70], [79]
	# consec. days queried	[70]
Popularity lists	2LD in Tranco Top 1M/500K/100K/10K/1K	[19], [66], [50]

*MW=malware, NS=nameserver, PBL=public blocklist, m/m/S=mean/median/STD, *only available in pDNS*

TABLE 3: **Reputation system features**—Features are computed over the DNS graph and attributes at several levels of aggregation. Exact or similar features are found in many related works.

4.3. Model Training & Evaluation

Our model is trained on feature vectors extracted from two weeks of DNS data. Our decision to use a two-week period reflects prior work (e.g., [3], [10], [50]) and balances the need for data richness with computational limitations. We split the data into training and test sets with an 80/20 ratio. To prevent data leakage, we ensure that all FQDNs that share 2LDs are in exactly one of these sets. We downsample FQDNs with overrepresented 2LDs in the training set while ensuring that the proportion of benign/malicious domains in the test set matches the proportion in the observed DNS traffic (0.8% malicious). We evaluated random forest (RF) [11], k-nearest-neighbors (KNN) [32], and logistic regression (LR) [25], ultimately choosing random forest for its superior performance to KNN and LR (Appendix B1). We select a final model based on each model's performance in minimizing the false positive rate (FPR) during a 10-fold cross-validation process and a grid search over hyperparameters. Training and evaluation code is available in our open-source repository.

Table 1 indicates that our SV1/aDNS reference model performs against the test set at a 99.1% true positive rate (TPR) and 1.7% false positive rate (FPR). This outperforms our OSINT model (98.4% TPR / 2.5% FPR) but comes at a significant data cost. More detailed metrics are reported in Appendix B1. Due to the low performance of the pDNS model, we primarily focus our evaluation on the aDNS model. While our attacks do succeed against the pDNS model, the high false negative rate makes it an easy target. While we are unable to directly compare our model to prior work due to differing training/testing data, we note that our model has a similar TPR and slightly worse FPR than prior work (curated in Appendix B2). This is likely a result of the practical assumptions in our data which lead to more difficult classification. For example, Segugio [70] filters domains with low numbers of queriers, and Exposure [10] removes domains that are older than one year. Additionally, prior work commonly sources benign domains exclusively from popularity lists, which we find leads to graph features that correlate with domain popularity and can thus cause models to overfit to it.

4.4. Attack Planning

We plan and execute attacks by applying *actions* performed on the problem space by the attacker, which subsequently induces changes to features. An attacker can apply actions directly to their domains (e.g., add a resource record at d) or external services (e.g., lookup d on resolver r). An attack is carried out against a malicious domain classifier, M, which may output a binary classification (malicious/benign) or a reputation score for the domain.

We model the problem space as a set of states representing the attacker's infrastructure. Given states G_t and G_{t+1}, an *action* $a(G_t) \mapsto G_{t+1}$ transitions between them. An *attack* is defined as a series of actions with an objective function. We consider a simplified model where the actions of an attack can be performed in any order and where all actions can be performed in parallel. We also enforce that actions cannot violate DNS specifications: for example, adding a CNAME record and an A record to the same node is prohibited by RFC-1034 [56]. Finally, actions within an attack cannot contradict each other: for example, an action that adds a certain RR cannot be paired with an action that removes the same RR. As a result of our simplifications, the world state after an attack is $G_n = \bigcup_{i=1}^{n} a_i(G_0)$.

An attack can be generated according to an optimization problem, where a domain's reputation score is either maximized or minimized and the attacker is bound by a set of constraints, C. An attack is defined formally as finding $a_1, a_2, \cdots, a_n$ applied to G_0 that optimizes:

$$f(d) = \max(\texttt{RepScore}(d, G_n))$$

subject to a financial budget, $\sum_{i=1}^{n} \texttt{Fin}(a_i) \leq C_F$, and a time budget, $\max(\texttt{Time}(a_i)) \leq C_T$.

While the objective function is easy to optimize under time constraints, maximizing the objective function subject to the sum of financial constraints is challenging. The actions do not contribute independently to the computation of $\texttt{RepScore}(d, G_n)$. Even under a simplifying assumption that the actions contribute to the reputation independently and linearly, the optimization is equivalent to the knapsack problem, making it NP-hard. The actions occur in a discrete problem space that is highly constrained and discontinuous, limiting the utility of gradient methods for approximation. To overcome these challenges, we use a hill-climbing algorithm [76] to approximate this optimization, as shown in Algorithm 1.

Algorithm 1 Altering the reputation of domain d by taking actions from $\mathcal{A}$. The attacker controls a domain in d_a.

```
Input: $d, n, \mathcal{A}$
  $r \leftarrow \texttt{RepScore}(d)$
  $A \leftarrow \{\}$
  while $n > 0$ do
      $a \leftarrow \texttt{RandomAction}(\mathcal{A})$
      if $\texttt{RepScore}(a(d)) > r$ then
          $r \leftarrow \texttt{RepScore}(a(d))$
          $d \leftarrow a(d)$
          $A \leftarrow A \cup \{a\}$
      end if
      $n \leftarrow n - 1$
  end while
  return $A$
```

An attack is *successful* when the reputation score of a solution exceeds a model's classification threshold function, δ. For evasion, this is $f(d) \geq \delta$; for poisoning, the threshold is $f(d) < \delta$.

We evaluate the aggregate effectiveness of our attacks by running N attacks, each on a different domain, and then by computing the *success rate* (S) and the *average change in reputation score* (ΔR) for each domain.

We define the success rate, $S(D)$, as the ratio of successful attacks to total attempts in A_D. For evasion,

$$S(A_D) = \frac{1}{|A_D|} \sum_{d \in A_D} \mathbf{1}\{\texttt{RepScore}(d, G_n) \geq \delta\}$$

In some cases, such as baselining, it is useful to find the average reputation for a set of domains. The average reputation of D in world state G_t is:

$$R(D, G_t) = \frac{1}{|D|} \sum_{d \in D} \texttt{RepScore}(d, G_t)$$

And the change in reputation due to a change in world state is:

$$\Delta R(G_n, G_0) = R(G_n) - R(G_0)$$

5. Attacks

In this section, we present practical attacks against DNS reputation systems. We begin by proposing attacks that domain name operators can use to target the most crucial feature groups used in prior work. While attacks against some feature groups, such as popularity list manipulation, are known, our mimicry attacks, which target graph-based features, are novel.

5.1. Attack Techniques

5.1.1. Popularity List Manipulation. Reputation systems in both academia and operational settings assign higher reputations to popular domains. This relationship can manifest as either a hard-coded allowlist for specific domains ([2]), a feature ([66], [19]), or by an implicit bias introduced when selecting data for the training set ([3], [10]). Popularity lists, however, are known to be unstable: Le Pochat et al. [47] find that an attacker can perform a small number of DNS lookups from many IP addresses to achieve a high rank on the Cisco Umbrella list [22]. As a result, many reputation systems have adopted the more stable Tranco list, which aggregates the data of other lists [47]. We will show that \$10 in VPN services is sufficient to achieve high rankings on the Cisco Umbrella List, the Cloudflare Radar List, and the downstream Tranco List in just two weeks.

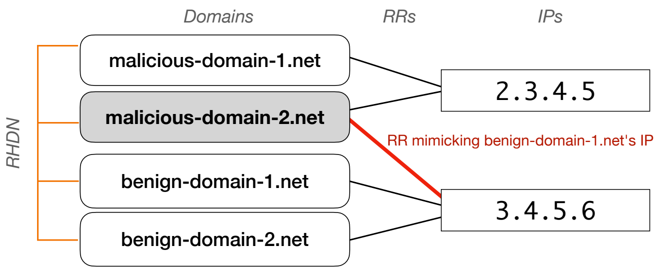


Figure 2: **IP mimicry overview**—Adversaries create new RRs for their malicious domain that resolve to IP addresses from benign RRs.

5.1.2. Mimicry Attacks. In a mimicry attack, adversaries temporarily copy the resource records (usually, A/AAAA, CNAME, NS, MX, or SPF) of a domain that they do not control. We focus on mimicry attacks as the first evasion of adversarial DNS reputation systems because they are easily accessible to adversaries and introduce low financial or operational burdens. They are also beneficial for multiple adversarial goals: evasion, sabotage, and data poisoning.

We first focus on A/AAAA record mimicry (Figure **??**) as domain-IP relationships are the only ones captured by our reference model, although we will later see that mimicry of NS records is effective against commercial models. The key insight behind these attacks is that an adversary only needs to control the domain and a single one of the IPs to which its type A RRs point. An adversary primes the reputation system with records pointing to *any* IP, including those belonging to benign domains, through *response overloading* or *record stuffing*. In *response overloading*, we create benign-mimicking type A RRs but always retain at least one type A record that points to malicious infrastructure. When a malware sample queries an evasive domain, it will iterate through the records and ultimately reach the desired malicious infrastructure. *Record stuffing* is performed either by poisoning a model's historical dataset before launching a campaign or by alternating between mimicked IPs and malicious IPs during the campaign. In the latter case, stuffing can be implemented by malware operators programmatically by rotating between multiple malicious domains.

Mimicry attacks exploit the assumption that a domain is similar to domains and IPs in its neighborhood. Using the intuition that many infrastructure-based features are summary statistics of entities related to the domain by its historical IPs (e.g., RHIPs, IP-RHDNs), we notice that IPs related to popular benign domains tend to introduce large numbers of these entities that dominate the summary statistics. Few malicious domains share these large values, which causes the model to associate large neighborhoods and second neighborhoods with benignness.

Group	Total Importance
Popularity	0.3625
Graph	0.164
Registration	0.1469
Lexical	0.0187
Evidence	0.0185
RR	0.0132

TABLE 4: **Feature importance per-feature group, SV1/aDNS model**—Feature importance is computed based on mean decrease in impurity on our random forest model. Appendix B3a contains a per-feature breakdown.

Attack	Success rate	ΔR	Time	$
IP mimicry	10%	0.39	1hr – 1wk	$0
Response overloading	-	-	<1 hour	$0
Record stuffing	-	-	<1 week	$0
Popularity list manipulation	-	-	-	$15
Top 1M	2.0%	0.33	<10 days	$10
Top 500K	100%	0.6	<2 weeks	$10
Long-term registration	-	-	-	-
>1 year	4.0%	0.16	<1 minute	$15
>5 years	6.0%	0.21	<1 minute	$60
1. Mimicry	10%	0.39	<1 day	$0
2. Mimicry/2yr reg.	60.0%	0.45	<1 day	$15
3. Pop 1M/mimicry	92.0%	0.55	<10 days	$10
4. Mimicry/Pop 500K	100.0%	0.33	<2 weeks	$10
5. Mimicry/500K/2yr reg.	100.0%	0.38	<2 weeks	$25

TABLE 5: **Practicality of evasion attacks against reference model**—We compare the success rates for each attack/attacker with their practicality.

6. Evaluation

We begin our evaluation by attacking individual feature groups, then combine the attacks and evaluate the performance of the reputation-boosting algorithm. We then determine the generalizability of our attacks against SV1 and evaluate their real-world practicality in terms of financial cost and time. Finally, we explore sabotage attacks against our reference model and SV1.

6.1. Attacking Individual Feature Groups

We begin our exploration of evasion attacks by attacking each feature group individually. Although an attack can be made more powerful by incorporating multiple feature groups, considering each feature group independently establishes a baseline and helps to illustrate the trade-off between ease of feature manipulation and evasion success. We consider manipulations of the most impactful feature groups: popularity-based features, graph-based (RHDN and RHIP) features, and registration-based features. Although the other feature groups (lexical features, resource records, TTL values, and temporal resolution patterns) affect the classification, they contribute just 5% of the total mean impurity, as indicated by Table 4. In each experiment, we randomly sample 200 malicious domains from the test set correctly labeled by the classifier. The overall results are shown in the upper half of Table 5.

6.1.1. IP Mimicry. To evaluate the strength of IP mimicry, we add A records resolving to IPs used by benign domains found in the consistent Tranco Top 100K. We assume a weak level of data visibility for the attacker: benign domains are sampled only from the testing set. We add $n = 1, 2, 3$ records *greedily*—in each iteration, we select the IP address that causes the greatest increase in reputation. If no such IP exists, the algorithm terminates. Figure 3b shows the results of this experiment.

6.1.2. Popularity. Our reference classifier uses five binary features related to popularity: existence in the Tranco 1M, 500K, 100K, 10K, and 1K. For our 200 malicious domains, we evaluate the effect on reputation by an attacker able to achieve each of these ranks. Figure 3a shows the relationship between rank and reputation. When the domain is in the top 500K, the model labels the domain as benign. When the domain is in the top 100K, the reputation approaches a perfect score.

6.1.3. Registration Length and Domain Age. Finally, we measure the effect of changing a domain's lifetime on its reputation. Figure 3c shows how registering a domain for at least two years increases its reputation regardless of the domain's age, but generally does not surpass the threshold for benign classification. Aging a domain, on the other hand, provides only a small increase in reputation, as shown in Figure 3d. In the next section, we will see that the effects of manipulating a domain's lifetime are amplified when combined with mimicry and popularity list attacks.

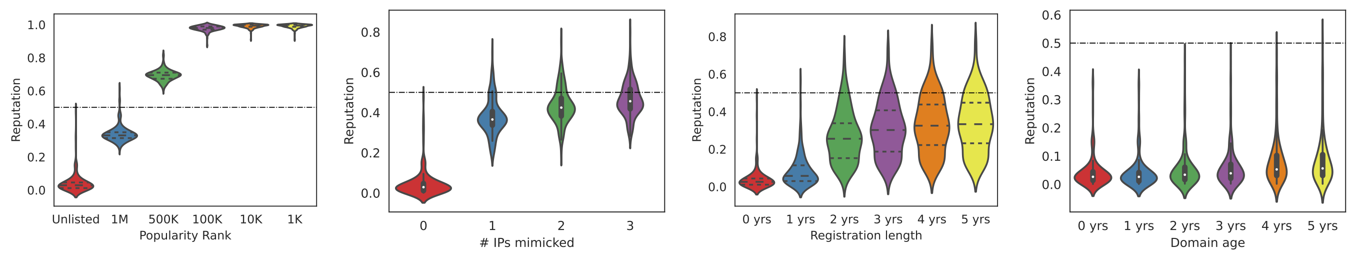
6.2. Reputation Boosting

Although some attacks do not cause the domain to cross the benign-malicious decision boundary, they do increase the domain's reputation. An attacker can combine such attacks to craft a hybrid attack that crosses the boundary. We greedily select problem-space actions that do not exceed the attacker's budget using the hill-climbing Algorithm 1. Table 5 shows the results for five different adversarial models.

Attacker 1: baseline *financial cost = $0, time = 1 day*. This attacker has no additional budget and has a short amount of time to perform their attack. In our action space, the only possible attack is a mimicry attack under these constraints, so this reduces to a single-feature group mimicry attack.

Attacker 2: mimicry + 2 year registration *financial cost = $15, time = 1 day.* This attacker has a small financial budget per domain and can extend their domain's registration period, which costs less than $15 for most domains and does not require aging. As shown in Table 5, the success rates of mimicry and 2-year registration individually are less than 5%, but when they are combined, attack success soars to 60%. The average change in reputation increases to 0.45 when combining the attacks, which is less than the sum of the individual changes after mimicry and 2-year registration attacks ($0.39 + 0.16 = 0.55$), implying some overlap between the two features.

Attacker 3: mimicry + popular 1M *financial cost = $10, time = 10 days*. In addition to mimicry, this attacker has the financial resources and patience to manipulate a



(a) Popularity rank (b) IP mimicry (c) Registration length (d) Domain age

Figure 3: **Reputation of malicious test set after manipulation of important feature groups**. We sweep across each attack and show the reputation for 200 malicious domains with that parameter.

stable popularity list. Similar to Attacker 2, the popular 1M feature and mimicry features both have under 5% success rate on their own, but together evade the model 92% of the time, which is a 50% improvement over the more time-constrained Attacker 2's success rate.

Attacker 4: mimicry + popular 500K *financial cost = $10, time = 2 weeks*. Unsurprisingly, pairing mimicry attacks with 500K popularity list manipulation does not reduce the effectiveness of reaching 500K on a popularity list.

Attacker 5: mimicry + 2 year registration + popular 500K *financial cost = $35, time = 10 days*. This is the upper bound of the financial costs and time for our attack space. The attacker continues to evade the model, and the change in reputation continues to increase as more feature groups are targeted.

6.3. Generalizability

Academic models provide insight into which feature classes are used in malicious domain detection and the impact of each feature group but often make assumptions that can increase false positive or false negative rates under real-world demands. In this section, we evaluate our attacks against a well-known security vendor (SV1). SV1 computes reputation scores based on features that are similar to prior work but uses a deterministic system engineered using expert heuristics to do so.

We introduce candidate domains into SV1's DNS reputation system by embedding them within malicious Windows executables generated by Metasploit's `msfvenom` tool [71]. In all of our experiments, the malicious binary is correctly classified as malicious by SV1's classifier. We arm the binary with defanged payloads that embed our domains in three different ways: (1) resolving the A record for the domain, then attempting to open a TCP connection to the IP over a non-standard, closed port, (2) running a Powershell command that performs a DNS lookup over a Cloudflare's DNS-over-HTTPS (DoH) resolver [24], or (3) storing the domain name in the binary, but not resolving it.

We found that the domain was only 'seen' by SV1 when the malware resolved the record, and did so in cleartext. This indicates that an attacker could evade the model for free if they can prevent it from being queried in a sandbox or look it up over DoH. For this reason, we primarily focus on malware that actually resolves the record and attempts to connect to it.

This requires executing our attacks in the real world, which can lead to unintended side effects. We leveraged our partnership with SV1 to ensure that our attacks remain isolated, which avoids poisoning the system against legitimate benign domains. Section 7.4 discusses our ethical considerations in detail.

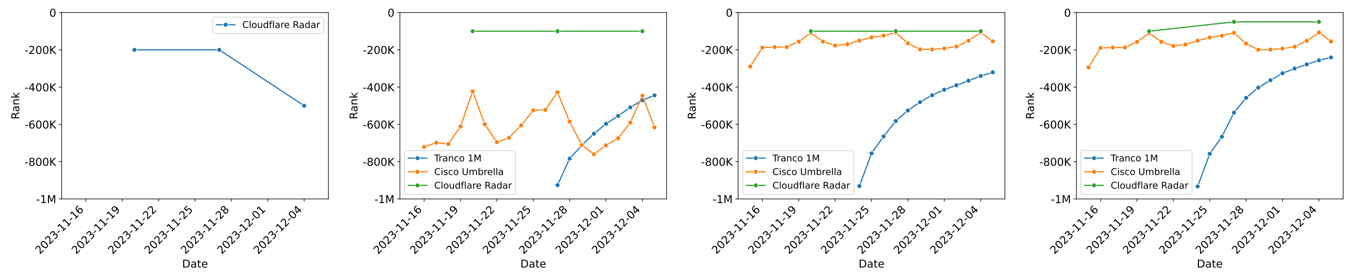
6.3.1. Domain Generation. To generate domains suitable for a real-world evaluation, we must ensure that the domains are representative of a previously undetected malicious campaign. We consider domains in two groups: generic domains (e.g., `california-news<.>com`) and DGA domains (e.g., `pnxp4<.>com`). While each group has different lexical characteristics, we did not find evidence that SV1 utilizes these features. For each group, we seed a large language model [59] with examples taken from a known campaign, then prompt it to generate five more. We then apply some steps to reduce potential noise. We separately consider domains that have been previously registered and domains that share their name with pre-existing domains. Then, we register the domains using Namecheap. We primarily register domains with low-cost TLDs (e.g., .xyz). While we did not find evidence that TLD affects classification on SV1, we note that a feature related to low-cost or abused TLDs would likely lower a domain's reputation, which would make our results non-optimistic for an attacker willing to spend more than $2/domain. We also assume that the operator has configured DMARC and DNSSEC. Many nameserver providers provide DNSSEC out of the box, including Namecheap. DMARC records are free to set up and have a negligible time cost. We use IPs from DigitalOcean to emulate a C2 server, ensuring that each IP address does not already appear on any blocklists.

SV1 lists all domains less than 24 hours old, so an attacker must wait a minimum of 24 hours before attempting to evade the model. We evaluate against SV1 when each domain has completed its attack, which closely bounds our proposed financial and time costs.

6.3.2. Popularity List Manipulation. We manipulate popularity lists by sending a small number of DNS queries to the Cisco OpenDNS resolver and Cloudflare's 1.1.1.1 resolver from a modest number ($N \leq 12,000$) of unique IPs between November 14, 2023 – December 1, 2023. We repeatedly grab new IPs from a $10/month subscription to Private Internet Access [68].

TLD	Tranco rank	Mw embedding	Age	Reg. length	IP mimicry	NS mimicry	ID'd as indicator	Evasion
.xyz	>1M	DNS lookup	3 days	1 yr	✓	✓	✓	
.network	>1M	DNS lookup	3 days	1 yr	✓		✓	
.xyz	>500K, <1M	DNS lookup	3 days	1 yr	✓	✓	✓	
.online	>1M	DNS lookup	3 days	2 yrs	✓	✓	✓	✓
.xyz	<500K	DNS lookup	15 days	1 yr	✓	✓	✓	✓
.online	<500K	DNS lookup	7 mo.	1 yr			✓	✓
.xyz	>1M	FQDN in strings	165 days	1 yr			✓	✓
.xyz	>1M	DoH lookup	165 days	1 yr				✓

TABLE 6: **Results of experiments against SV1**



(a) 3,000 queries/day (b) 9,000 queries/day (c) 3,000 queries/day (d) 12,000 queries/day

Figure 4: **Results from popularity list manipulation on Cisco Umbrella, Cloudflare Radar, and Tranco**—Daily popularity rankings after manipulation using a VPN service.

Figure 4 shows the results of sending DNS queries from various numbers of unique IP addresses using two VPN services. With a $10 subscription to PIA, we found that 9,000 lookups from different clients are sufficient to achieve high rankings in the Umbrella and Radar lists. We consistently achieved a ranking of under 250,000 on the Cisco Umbrella list. On the Cloudflare Radar list, which contains only 2LDs, we consistently achieved a ranking in the top 100,000. As a result, we achieved a ranking of 1 million on the Tranco list within 10 days and within 500K on the Tranco list within 14 days.

6.3.3. Reputation Boosting against SV1. By combining our domain generation and popularity list manipulation techniques, we can combine attacks. We take a grey-box approach to SV1. We have partial knowledge of their features, and also in some cases experiment directly with their malicious ground truth, which is available commercially for around $500k. We do not have access to the implementation details or feature enrichment data used at SV1, including the source of DNS traffic. We find that SV1 is similar to the same types of attacks as both reference classifiers. We find that SV1 can be evaded at either a $15/domain cost in around a day or at a $10 total cost in around two weeks.

The results are shown in Table 6. We began by running IP mimicry and popularity list attacks individually against SV1. We found that IP mimicry was not effective and that popularity list manipulation is only effective when boosted by another feature. We found that the registration length of a domain, which was effective in boosting IP mimicry against the reference classifier, also sufficiently boosts popularity manipulation, allowing it to succeed.

While our model does not model nameserver-based features, we learned that SV1 is vulnerable to nameserver mimicry attacks through manual testing. An attacker can create a false link between their domain and a reputable nameserver to increase their reputation. When overloading DNS responses with mimicked nameservers, round-robin nameserver selection introduces a m/N resolution error, where n is the number of mimicked nameservers and N is the total number of nameservers. When using the stronger record stuffing technique to poison the historical DNS dataset, this error disappears at a negligible time cost. We notified SV1 of this issue, who confirmed that it is a vulnerability. We propose two solutions: consider only current nameservers when assigning positive reputation, which mitigates response stuffing, and ensure that the domain resolves on each of its advertised nameservers, which mitigates response overloading. We found that this is a viable way to boost the reputation of the registration length feature, which reduces the time cost for an attack from two weeks to around one day for nameserver propagation.

6.4. Practicality

We now analyze our attacks through the lens of an attacker who is constrained by financial and temporal resources, as introduced in Section 3. We also qualitatively assess the detectability risk of each attack.

6.4.1. IP Mimicry. IP mimicry via response overloading is easy to perform—an adversary can simply add new type A RRs to the domain's DNS records. When implementing an attack with periodic record stuffing, however, the feasibility decreases as an attacker must coordinate benign periods and malicious periods with their malware code or phishing campaign, which takes additional time. In both cases, the detectability is high—mimicked IP addresses may result in

HTTP requests to the IP's legitimate owners, which often include metadata about the domain name used to initiate the request. The attack is difficult to attribute because the adversary is not required to register any additional infrastructure that could reveal their identity.

6.4.2. Popularity List Manipulation. Our VPN-based technique is an effective, low-cost solution to popularity list manipulation. VPN subscriptions range from free to around $10 and provide access to thousands of IPs around the world. Compared to prior work, which uses ephemeral IPs in cloud hosting providers [47] or IP spoofing [43] to manipulate the list, our technique can achieve higher rankings in less time and financial cost. Finding a hosting provider that allows IP spoofing is difficult, and abusing ephemeral cloud IPs comes at a financial cost and may lead to abuse complaints. Furthermore, our VPN-based method achieves a ranking of 500K on the Tranco list in under 14 days, which is more effective than prior methods.

7. Discussion

Our findings reveal the poor resistance of DNS reputation systems to adversarial attacks, even from relatively unsophisticated actors. To advance the robustness of these systems, we outline opportunities for hardening reputation models and ultimately translating these insights to production DNS reputation systems. We then discuss our study's limitations and ethical considerations.

7.1. Hardening DNS Reputation Systems

7.1.1. Remove Easily Evadable Features. Some of our evasion tactics are easier to deploy than others. Avoiding the use of certain features can help prevent evasion at the cost of overall model performance. We analyze the trade-off between evasion success and model performance and find that in some cases, the risk of potential evasion is higher than excluding the evadable feature. In Table 7, we show the change in TPR, FPR, and attack success rate against our reference model after removing certain feature classes. Across each experiment, we use identical training processes (train/test split, feature extraction, etc.). While removing the popularity features is detrimental to model performance (427% increase in FPR, 4.5% decrease in TPR), it reduces the efficacy of popularity manipulation, our most effective attack vector.

7.1.2. Adversarial Training. A popular technique for defending against adversarial attacks is for the defender to train their model on adversarial samples, thus increasing the model's robustness to the attacks [34]. DNS reputation systems commonly operate with a human-in-the-loop, which enables retraining on evasive samples. Over time, human analysts manually label popular, sophisticated, or unique threats. A defender could also generate hand-labeled adversarial samples by running the attacks outlined in our paper. If these hand-labeled samples are included in the model's training set, then the model could learn to correctly classify samples despite our attacks. Although this would provide a safeguard against mimicry attacks, large volumes of successful adversarial attacks could poison the retrained DNS reputation systems, thus increasing false positives.

7.2. Limitations

Our examination of practical attacks against DNS reputation systems faces several limitations. First, we cannot fully test every component of systems. If a malicious domain is used for spam rather than C2, systems could gather information related to the number of emails delivered and the timing of email deliveries. We cannot deliver spam as a part of an experiment as it is against the terms of service of most hosting providers. Additionally, emails traverse through many services en route to their destination, which makes it difficult to obtain informed consent from all parties that may be negatively impacted by sending large amounts of spam emails. Second, the practical constraints are best-effort approximations, sourced from publicly available market costs. In practice, there may be bulk-purchase discounts for pDNS data or domain registration, or alternate methods for popularity list manipulation. Fine-tuning these models is a topic for future work.

While we focus on attacks against the features used in DNS reputation systems, other methods of attack exist. Attackers could abuse blocklist removal requests or compromise a server hosted on a high-reputation domain. Such attacks are outside the scope of this work. Similarly, malware operators that use DNS services can program their malware to avoid or circumvent DNS reputation systems. For instance, malware authors could implement execution guardrails that alter behavior when the malware is run by someone other than their intended target [67], a technique commonly exploited by APT groups [46]. Malware can also leverage the DNS over HTTPS (DoH) protocol to bypass DNS reputation systems deployed by recursive resolvers, network firewalls, or dynamic analysis systems. We find that this technique is effective in evading SV1's DNS reputation system.

7.3. Sabotage Attacks

In addition to evading the model, an attacker can induce false positives for other domains by performing graph-based attacks or by submitting malware referencing these domains. This can cause a denial of service for the victim or cause alert fatigue for security analysts. A dual interpretation of our results in Table 6 is that by uploading malicious binaries to threat intelligence systems, attackers can induce false positives for certain benign domains. While the lower bound for defending against sabotage attacks is only $15 and three weeks, a benign site is unlikely to defend against the attack by proactively performing mimicry attacks or popularity list manipulation. In reality, unpopular domains (Tranco >500K) with short registration lengths are highly susceptible to sabotage attacks. As many threat intelligence platforms gather data from the same sources, their indicators often overlap [49]. As a result, attackers may be able to attack many security vendors using this technique.

7.4. Ethical Considerations

We took several precautions to ensure that our study remained ethical by following widely accepted ethical principles in computer security [7] and by strictly adhering

Removed feature group	TPR	FPR	IP Mimicry		Pop. (1M)		Pop. (500K)	
			S	ΔR	S	ΔR	S	ΔR
RHDNs	97.33%	3.27%	0.00%	0.01	2.00%	0.35	100.00%	0.61
Lexical	97.80%	2.82%	12.50%	0.03	50.00%	0.19	100.00%	0.39
Registration	97.91%	9.09%	0.00%	0.00	100.00%	0.12	100.00%	0.31
Network	97.95%	2.64%	20.00%	0.04	100.00%	0.27	100.00%	0.42
Popularity	94.66%	7.01%	2.22%	0.02	0.00%	0.00	0.00%	0.00
RRs	97.81%	2.87%	0.00%	0.00	100.00%	0.19	100.00%	0.34
None	99.12%	1.33%	10.00%	0.39	2.00%	0.33	100.00%	0.60

TABLE 7: **Effect of removing features on model performance and robustness (gray-box attack against aDNS model with SV1 ground truth)**

to our institute's data governance policies and contractual agreements with our vendors. First, when performing controlled attacks against the operational system at SV1, we worked closely with their team to ensure that we did not compromise the service's integrity or availability. We also coordinated with their team to isolate the reputation change caused by our experiments from real-world infrastructure. Further, we ensured that our experiments were not included in their historical dataset, preventing real infrastructure from being negatively affected in the future. We disclosed all vulnerabilities found to SV1. We notified both Cisco and Cloudflare of the potential for popularity list manipulation, who informed us that this is a known issue.

When operating on pDNS records from our data provider, we look at only aggregated daily resolution data, which does not reveal any information about a single client's behavior. This data is stored on a secure server compliant with our institution's computer security policy with limited user access.

8. Conclusion

DNS reputation systems are an interesting subclass of ML models where adversarial attacks are highly constrained by their realizability and practicality. By constructing a reference model, we are able to evaluate the success rate, financial cost, and time cost of attacks against these systems. These reference models help construct precise attacks against commercial DNS reputation systems. In both cases, operators face a trade-off between attack susceptibility and runtime/memory/classification performance. Our novel attacks against multiple DNS reputation feature classes show that current approaches in both academia and industry can be evaded with up to a 100% success rate by an attacker with a budget of \$10 and two weeks of aging. Our discussion of the attacks, their practicality, their root causes, and their mitigations will assist future researchers and commercial vendors. We open-source our system's code to improve reproducibility in this research area.

Acknowledgments

We thank the anonymous reviewers: their insight and attention to detail greatly improved this work. We also thank our industry collaborators, without whom this work would not be possible. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001123C0035. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Department of Defense (DoD).

References

[1] Abuse.ch. SinkDB. https://sinkdb.abuse.ch/.

[2] Abuse.ch. URLhaus. https://urlhaus.abuse.ch/.

[3] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a dynamic reputation system for DNS. In *USENIX Security Symposium*, 2010.

[4] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon. Detecting malware domains at the upper DNS hierarchy. In *USENIX Security Symposium*, 2011.

[5] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon. From Throw-Away traffic to bots: Detecting the rise of DGA-Based malware. In *USENIX Security Symposium*, 2012.

[6] G. I. o. T. Astrolavos. Active DNS Project. http://www.activednsproject.org/.

[7] M. Bailey, D. Dittrich, E. Kenneally, and D. Maughan. The menlo report. *IEEE Security & Privacy*, 10(2):71–75, 2012.

[8] Better Business Bureau. BBB Database. https://www.bbb.org/.

[9] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. *CoRR*, abs/1708.06131, 2017.

[10] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi. Exposure: Finding malicious domains using passive DNS analysis. In *Network & Distributed System Security Symposium*, 2011.

[11] L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[12] CAIDA. Routeviews Prefix-to-AS mappings (pfx2as) for IPv4 and IPv6. http://data.caida.org/datasets/routing/routeviews-prefix2as/, 2023.

[13] CAIDA. The CAIDA AS Organizations Dataset. https://publicdata.caida.org/datasets/as-organizations/, 2023.

[14] U. T. . Capitole. Prigent List. https://dsi.ut-capitole.fr/blacklists/index_en.php.

[15] F. Cara, M. Scalas, G. Giacinto, and D. Maiorca. On the feasibility of adversarial sample creation using the Android system API. *Information*, 11(9), 2020.

[16] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[17] X. Chen, C. Li, D. Wang, S. Wen, J. Zhang, S. Nepal, Y. Xiang, and K. Ren. Android HIV: A study of repackaging malware for evading machine-learning detection. *IEEE Transactions on Information Forensics and Security*, 15, 2020.

[18] Y. Chen, Y. Nadji, A. Kountouras, F. Monrose, R. Perdisci, M. Antonakakis, and N. Vasiloglou. Practical attacks against graph-based clustering. In *ACM SIGSAC Conference on Computer and Communications Security*, 2017.

[19] D. Chiba, T. Yagi, M. Akiyama, T. Shibahara, T. Yada, T. Mori, and S. Goto. Domainprofiler: Discovering domain names abused in future. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2016.

[20] H. Choi and H. Lee. Identifying botnets by capturing group activities in DNS traffic. *Computer Networks*, 56(1), 2012.

[21] Cisco. IP & Domain Reputation Center. https://talosintelligence.com/reputation_center.

[22] Cisco. Umbrella 1 Million. https://umbrella-static.s3-us-west-1.amazonaws.com/index.html.

[23] Cloudflare. Cloudflare Radar. https://radar.cloudflare.com/.

[24] Cloudflare. DNS over HTTPS. https://developers.cloudflare.com/1.1.1.1/encryption/dns-over-https/.

[25] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232, 1958.

[26] A. Demontis, M. Melis, B. Biggio, D. Maiorca, D. Arp, K. Rieck, I. Corona, G. Giacinto, and F. Roli. Yes, machine learning can be more secure! A case study on android malware detection. *IEEE Transactions on Dependable and Secure Computing*, 16(4), 2017.

[27] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[28] Farsight Security. Dnsdb 2.0. https://www.domaintools.com/products/farsight-dnsdb/.

[29] A. Faulkenberry, A. Avgetidis, Z. Ma, O. Alrawi, C. Lever, P. Kintis, F. Monrose, A. D. Keromytis, and M. Antonakakis. View from above: Exploring the malware ecosystem from the upper DNS hierarchy. In *Annual Computer Security Applications Conference (ACSAC)*, December 2022.

[30] S. Fernandez, M. Korczyński, and A. Duda. Early detection of spam domains with passive dns and spf. In *International Conference on Passive and Active Network Measurement*, pages 30–49. Springer, 2022.

[31] Firehol. Firehol IP Blocklist. https://firehol.org.

[32] E. Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3):238–247, 1989.

[33] Fraunhofer FKIE. DGArchive. https://dgarchive.caad.fkie.fraunhofer.de/welcome/.

[34] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples, 2015.

[35] Google. Chrome compliant CT logs. https://www.certificate-transparency.org/known-logs.

[36] S. Hao, A. Kantchelian, B. Miller, V. Paxson, and N. Feamster. Predator: Proactive recognition and elimination of domain abuse at time-of-registration. In *ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[37] X. Hu, M. Knysz, and K. G. Shin. Measurement and analysis of global IP-usage patterns of fast-flux botnets. In *IEEE INFOCOM*, 2011.

[38] IANA. Iana ipv6 special-purpose address registry. https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml, 2015.

[39] IANA. Iana ipv6 special-purpose address registry. https://www.iana.org/assignments/iana-ipv6-special-registry/iana-ipv6-special-registry.xhtml, 2015.

[40] M. Ikram, P. Beaume, and M. A. Kaafar. Dadidroid: an obfuscation resilient tool for detecting android malware via weighted directed call graph modelling. In *International Joint Conference on e-Business and Telecommunications*, 2019.

[41] Invaluement. Invaluement domain blocklist. https://www.invaluement.com/.

[42] I. Khalil, T. Yu, and B. Guan. Discovering malicious domains through passive DNS data graph analysis. In *ACM Asia Conference on Computer and Communications Security*, 2016.

[43] E. Kirda. Getting under alexa's umbrella: Infiltration attacks against internet top domain lists. In *Information Security: 22nd International Conference, ISC 2019, New York City, NY, USA, September 16–18, 2019, Proceedings*, volume 11723, page 255. Springer Nature, 2019.

[44] A. Kountouras, P. Kintis, C. Lever, Y. Chen, Y. Nadji, D. Dagon, M. Antonakakis, and R. Joffe. Enabling network security through active dns datasets. volume 9854, pages 188–208, 09 2016.

[45] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. 2018.

[46] P. Kálnai. Lazarus luring employees with trojanized coding challenges: The case of a spanish aerospace company, sep 2023.

[47] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Network and Distributed System Security Symposium*, 2019.

[48] C. Lever, R. Walls, Y. Nadji, D. Dagon, P. McDaniel, and M. Antonakakis. Domain-z: 28 registrations later measuring the exploitation of residual trust in domains. In *2016 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE, 2016.

[49] V. G. Li, M. Dunn, P. Pearce, D. McCoy, G. M. Voelker, and S. Savage. Reading the tea leaves: A comparative analysis of threat intelligence. In *28th USENIX security symposium (USENIX Security 19)*, pages 851–867, 2019.

[50] P. Lison and V. Mavroeidis. Neural reputation models learned from passive dns data. pages 3662–3671, 12 2017.

[51] S. Liu, I. Foster, S. Savage, G. M. Voelker, and L. K. Saul. Who is. com? learning to parse whois records. In *Proceedings of the 2015 Internet Measurement Conference*, pages 369–380, 2015.

[52] Majestic. The Majestic Million. https://majestic.com/reports/majestic-million.

[53] P. K. Manadhata, S. Yadav, P. Rao, and W. Horne. Detecting malicious domains via graph inference. In *European Symposium on Research in Computer Security*, 2014.

[54] S. Maroofi, M. Korczyński, C. Hesselman, B. Ampeau, and A. Duda. Comar: Classification of compromised versus maliciously registered domains. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020.

[55] I. Mishsky, N. Gal-Oz, and E. Gudes. A topology based flow model for computing domain reputation. In P. Samarati, editor, *Data and Applications Security and Privacy XXIX*, page 277–292, Cham, 2015. Springer International Publishing.

[56] P. V. Mockapetris. Rfc1034: Domain names-concepts and facilities. Technical report, 1987.

[57] Mozilla. Public suffix list. https://publicsuffix.org/.

[58] NordSpam. The nordspam project. https://www.nordspam.com/.

[59] OpenAI. Chatgpt. https://chat.openai.com/.

[60] Palo Alto Networks Blog. Domain reputation lookup. https://live.paloaltonetworks.com/t5/community-blogs/new-subdomain-reputation-detection-for-dns-security/ba-p/533484.

[61] R. Perdisci, I. Corona, D. Dagon, and W. Lee. Detecting malicious flux service networks through passive analysis of recursive DNS traces. In *Annual Computer Security Applications Conference*, 2009.

[62] R. Perdisci, I. Corona, and G. Giacinto. Early detection of malicious flux networks via large-scale passive dns traffic analysis. *IEEE Transactions on Dependable and Secure Computing*, 9(5), 2012.

[63] PhishingArmy. PhishingArmy Blocklist. https://phishing.army/.

[64] Y. Piao, J.-H. Jung, and J. H. Yi. Server-based code obfuscation scheme for APK tamper detection. *Sec. and Commun. Netw.*, 9(6), Apr 2016.

[65] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro. Intriguing properties of adversarial ML attacks in the problem space. In *IEEE Symposium on Security and Privacy (S&P)*, 2020.

[66] V. Pochat, T. Hamme, S. Maroofi, T. Van Goethem, D. Preuveneers, A. Duda, W. Joosen, and M. Korczynski. A practical approach for taking down avalanche botnets under real-world constraints. 01 2020.

[67] R.-M. Portase and A. Coleşa. Curator - a system for creating data sets for behavioral malware detection. In *2021 20th International Symposium on Parallel and Distributed Computing (ISPDC)*, pages 57–64, 2021.

[68] Private Internet Access. PIA VPN. https://www.privateinternetaccess.com/.

[69] Radix. Radix abuse report. https://www.radix.website/.

[70] B. Rahbarinia, R. Perdisci, and M. Antonakakis. Segugio: Efficient behavior-based tracking of malware-control domains in large isp networks. *IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015.

[71] Rapid7. Metasploit. https://www.metasploit.com/.

[72] V. Rastogi, Y. Chen, and X. Jiang. Droidchameleon: Evaluating android anti-malware against transformation attacks. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, page 329–334, New York, NY, USA, 2013. Association for Computing Machinery.

[73] I. Rosenberg, A. Shabtai, L. Rokach, and Y. Elovici. Generic black-box end-to-end attack against state of the art API call based malware classifiers. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, 2018.

[74] S. Schiavoni, F. Maggi, L. Cavallaro, and S. Zanero. Phoenix: DGA-Based Botnet Tracking and Intelligence. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, 2014.

[75] S. Schüppen, D. Teubert, P. Herrmann, and U. Meyer. FANCI : Feature-based automated NXDomain classification and intelligence. In *USENIX Security Symposium*, 2018.

[76] B. Selman and C. P. Gomes. Hill-climbing search. *Encyclopedia of cognitive science*, 81:82, 2006.

[77] R. Sheatsley, B. Hoak, E. Pauley, Y. Beugin, M. J. Weisman, and P. McDaniel. On the robustness of domain constraints. In *ACM SIGSAC Conference on Computer and Communications Security*, 2021.

[78] R. D. Silva, M. Nabeel, C. Elvitigala, I. Khalil, T. Yu, and C. Keppitiyagama. Compromised or attacker-owned: A large scale classification and study of hosting domains of malicious urls. page 19.

[79] M. R. Silveira, L. Marcos da Silva, A. M. Cansian, and H. K. Kobayashi. Detection of newly registered malicious domains through passive dns. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 3360–3369, 2021.

[80] Spamhaus. IP and domain reputation checker. https://check.spamhaus.org/.

[81] J. Spooren, T. Vissers, P. Janssen, W. Joosen, and L. Desmet. Premadoma: an operational solution for dns registries to prevent malicious domain registrations. pages 557–567, 12 2019.

[82] StopForumSpam. StopForumSpam Blocklist. https://www.stopforumspam.com/.

[83] O. Suciu, R. Marginean, Y. Kaya, H. D. III, and T. Dumitras. When does machine learning FAIL? generalized transferability for evasion and poisoning attacks. In *USENIX Security Symposium*, 2018.

[84] M. Sun and G. Tan. Nativeguard: Protecting Android applications from third-party native libraries. In *ACM Conference on Security and Privacy in Wireless & Mobile Networks*, 2014.

[85] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks, 2014.

[86] ThreatLog. Threatlog. https://www.threatlog.com/.

[87] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras. A high-performance, scalable infrastructure for large-scale active dns measurements. *IEEE Journal on Selected Areas in Communications*, 34(6):1877–1888, 2016.

[88] WhoisXML API. New subdomain reputation detection for DNS security. https://domain-reputation.whoisxmlapi.com/lookup.

[89] YP, LLC. The Real Yellow Pages. https://yp.com.

[90] Y. Zhauniarovich, I. M. Khalil, T. Yu, and M. Dacier. A survey on malicious domains detection through dns data analysis. *ACM Computing Surveys (CSUR)*, 51:1 – 36, 2018.

Meta Review

To be added in final camera-ready.

Paper	**DNS visibility**	**Specialization**	**Models**	**Deployment**	**Addt'l data sources**	**Feature types**	**Detection**
Perdisci09 [61]	Rec NS (below)	Fast flux	Clustering/Classifier	Telcom	IP probing	Clustering, Infra.	Operation
Antonakakis10 [3]	Rec NS (above)	Agnostic	Classifier	Telcom	Blocklists, IPWHOIS	Evidence, Infra., Zone	Operation
Antonakakis11 [4]	Auth/TLD NS (below)	Malware	Classifier	Zone authority	-	Infra., Pop., Resol.	Operation
Bilge11 [10]	Rec NS (above)	Agnostic	Classifier	Telcom	-	Lex., Resol.	Operation
Choi11 [20]	Rec NS (below)	Botnets	Clustering/Classifier	Telcom/Enterprise	-	Lex., Infra., Resol.	Operation
Antonakakis12 [5]	Rec NS (above)	DGA via NXDOMAIN	Clustering/Classifier	Telcom	-	Clustering, Lex., Infra.	Operation
Perdisci12 [62]	Rec NS (above)	Fast flux	Clustering/Classifier	Telcom	-	Infra.	Operation
Schiavoni14 [74]	Rec NS (above)	DGA	Clustering/Classifier	Telcom	-	Lex., Infra.	Operation
Mishsky15 [53]	Rec NS (above)	Malware	Graph	Telcom	Block/Allowlist, WHOIS	Infra.	Operation
Rahbarinia15 [70]	Rec NS (below)	Malware	Classifier	Telcom	Block/Allowlist	Infra., Pop., Resol.	Operation
Hao16 [36]	Registry logs	Agnostic	Classifier	Registry	WHOIS	Lex., Regis.	Regis.
Khalil16 [42]	Rec NS (above)	Agnostic	Graph	Telcom/Enterprise	-	Infra.	Operation
Lison17 [50]	Rec NS (below)	Agnostic	Classifier	Telcom	IPWHOIS, WHOIS	Infra., Lex., Pop., Resol.	Operation
Chiba18 [19]	Rec NS (above)	Agnostic	Classifier	Telcom/Enterprise	WHOIS, Block/Allowlist	Infra., Lex., Pop., Regis.	Operation
Schüppen18 [75]	Rec NS (above)	DGA via NXDOMAIN	Classifier	Enterprise	-	Lex.	Operation
LePochat20 [66]	Rec NS (above)	Avalanche botnet (DGA)	Classifier	Law Enforcement	aDNS, WHOIS, CT, WM	Infra., Lex., Pop., Resol.	Operation
Maroofi20 [54]	Blocklist+aDNS	Compromised vs. Malicious	Classifier	Registry	aDNS, WHOIS, HTTP req.	Infra., Lex., Pop., Regis.	Operation
Desmet21 [81]	Registry logs	Agnostic	Classifier, Clustering	Registry	WHOIS	Lex., Regis.	Regis.
Silveira21 [78]	TLD NS (above)	Agnostic	Classifier	Registry	IPWHOIS, registry logs	Infra., Regis., Resol.	Operation
Fernandez22 [30]	Rec NS (above)	Email spam	Classifier	Mail Servers	-	Evidence, Infra, Resol., SPF	Operation

TABLE A1: **Related work in DNS reputation systems**—Prior work varies in DNS visibility, feature sets, specialization, deployment scenarios, and machine learning models. Generic, network-deployed, operation-time detecting reputation systems that are targeted by our reference model are bolded. Terminology borrowed from Zhauniarovich et al.'s survey [90]. *CT=Certificate Transparency, Lex=Lexical, Pop=Popularity, Regis=Registration, Resol=Resolution*

Classifier	Malicious Ground Truth	Benign Ground Truth	DNS data	Date Range	F1	Precision	Recall	AUC
RF	SV1	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	**98.27%**	**99.05%**	**98.66%**	**98.76%**
KNN	SV1	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	96.16%	97.25%	96.70%	96.91%
LR	SV1	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	86.75%	97.43%	91.78%	92.14%
RF	OSINT	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	**95.92%**	**97.50%**	**96.70%**	**97.50%**
KNN	OSINT	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	94.20%	96.62%	95.39%	96.51%
LR	OSINT	Tranco + YP.com	aDNS	2023-10-07 - 2023-10-21	87.62%	97.53%	92.31%	94.60%

TABLE B1: **Model performance**—*RF = random forest, KNN = k-nearest-neighbors, LR = logistic regression*

Paper	Malicious Ground Truth	Benign Ground Truth	DNS data	TPR	FPR
Antonakakis10 [3]	OSINT	Alexa Top 500	pDNS	96.8%	0.38%
Antonakakis10 [3]	OSINT	Alexa Top 10K	pDNS	93.6%	0.4%
Antonakakis10 [3]	OSINT	Alexa Top 100K	pDNS	80.6%	0.6%
Bilge11 [10]	OSINT	Alexa 1K + all domains aged >1 yr	pDNS	98.4%	1.1%
Lison17 [50]	OSINT/private	Alexa Top 1M/OSINT/private	pDNS	95%	0.1%
Chiba18 [19]	Sandbox/paid lists/OSINT	Alexa Top 1M	aDNS	98.5%	$\sim 1\%$

TABLE B2: **Model performance of prior work in generic malicious domain detectors.**

TABLE B3: **Feature importance for SV1/aDNS and OSINT/aDNS models**—Feature importance is calculated using mean impurity of the random forest model (cutoff: 0.001)

(a) SV1/aDNS Model

Feature	Importance
days_created_expires	0.2883
days_created_now	0.1469
pop_1m	0.1292
pop_500k	0.117
pop_100k	0.1131
com_other_ratio	0.0217
distinct_ips_2ld_zone	0.0177
ip_blocklist_bgp	0.0153
number_ratio	0.0108
rhdn_length_mean	0.0099
distinct_prefixes_2ld_zone	0.0093
rhdn_length_std	0.0091
rhdn_1gram_std	0.0086
rhdn_length_median	0.0073
stddev_tld_freq	0.007
distinct_ips_3ld_zone	0.0068
distinct_ips	0.0067
rhdn_1gram_mean	0.0059
rhdn_count	0.0057
rhdn_1gram_median	0.0051
median_tld_freq	0.0049
rhdn_2gram_std	0.0048
rhdn_2gram_mean	0.0046
rhdn_2gram_median	0.0038
pop_10k	0.0032
ip_blocklist_asn	0.0032
distinct_tld_count	0.0032
rhdn_3gram_std	0.0031
avg_tld_freq	0.0029
domain_length	0.0027
rhdn_3gram_mean	0.0026
longest_human_readable_substring	0.0026
num_of_trigrams	0.0025
distinct_prefixes_3ld_zone	0.0024
distinct_asn_2ld_zone	0.0016
distinct_bgp_orgs	0.0016
distinct_as_names_2ld_zone	0.0013
entropy	0.0012
num_ip_reg_dates_fqdn	0.001

(b) OSINT/aDNS Model

Feature	Importance
days_created_expires	0.5145
days_created_now	0.2054
com_other_ratio	0.0334
rhdn_length_mean	0.0321
rhdn_length_median	0.0285
domain_length	0.0211
num_of_trigrams	0.0155
ip_blocklist_asn	0.0147
avg_tld_freq	0.0134
longest_human_readable_substring	0.0117
distinct_tld_count	0.0116
ip_blocklist_bgp	0.0107
stddev_tld_freq	0.0078
entropy	0.0073
rhdn_1gram_mean	0.006
rhdn_count	0.006
rhdn_3gram_std	0.0055
rhdn_1gram_std	0.0055
median_tld_freq	0.0053
rhdn_2gram_std	0.005
rhdn_1gram_median	0.0044
rhdn_3gram_mean	0.0041
rhdn_length_std	0.004
rhdn_2gram_mean	0.0034
number_ratio	0.0028
pop_500k	0.0027
pop_1m	0.0021
pop_100k	0.002
rhdn_2gram_median	0.002
distinct_ips_2ld_zone	0.0018
distinct_prefixes_2ld_zone	0.0014