# A Linguistics-based Stacking Approach to Disposable Domains Detection

Yuwei Zeng*†, Yongzheng Zhang*†, Tianning Zang*†, Xunxun Chen*‡, and Yipeng Wang*†
* Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
† School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
‡ National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China
Email: zangtianning@iie.ac.cn, xx-chen@139.com

***Abstract*—More Internet services tend to collect the one-time information from clients via DNS queries. Notably, the uncertainty of such transient information makes these domain names be queried only once in their lifetime. This type of domain is called disposable domain. Although they are not malicious, the efficiency of DNS infrastructures will still be affected by their ever-increasing number. In this paper, we propose Vogers, a linguistics-based stacking model, to detect the disposable domains. Our evaluation demonstrates that Vogers decreases the false positive rate by more than 19%, compared with the prior art, while maintaining the true positive rate above 98.9%**

## I. Introduction

This paper concentrates on disposable domain names, which are leveraged by some Internet services to convey provisional information from clients to servers [1]. These domain names are automatically generated by algorithms in bulk. But unlike some DNS-based covert channels [2], disposable domain names are not malicious. For instance, the domain name 'only-930918-111-207-250-35.nstool.netease.com' is generated by a DNS configuration check service of NetEase. This service compresses the client's DNS address, IP address, and current timestamp into the subdomain of 'nstool.netease.com', and actively queries the combined domain name to inspect the related configurations. However, the uncertainty of such system information makes every generated domain name quite unique, which means that these domain names will be queried only once in their lifetime. According to the existing literature [3], 27.6% of all queried domain names are disposable, resulting in a bloated DNS ecosystem. Obviously, such disposable domains can severely influence the efficiency of DNS infrastructures in two aspects. First, an influx of disposable domains will drain the available cache space of DNS resolvers, and even crowd out the resource records associated with popular domains, which inevitably degrades the response efficiency of recursive DNS resolvers. Second, disposable domains affect the efficiency of PDNS database. Redundant disposable domains will put unnecessary storage pressure on the database and affect its I/O speed. It is therefore necessary for researchers to detect and eliminate such disposable domains.

We propose a stacking system, called Vogers, to detect disposable domains. This system can determine whether the input domain name is disposable solely based on the domain name itself without any additional intelligence. Besides, Vogers does not require any scale information and can directly act on a single domain name. This system has three modules. The first module is feature extractor, which is to obtain the linguistic features of domain names. The second module is three parallel readability score assigners. This module assigns readability scores to domain names as a function of the linguistic features of three parts, namely the prefix, the longest label in the prefix, and the suffix. The last module, a binary classifier, determines the category of each domain name based on three readability scores output by the former.

978-1-7281-2700-2/19/$31.00 2019 © IEEE

## II. Data Analysis

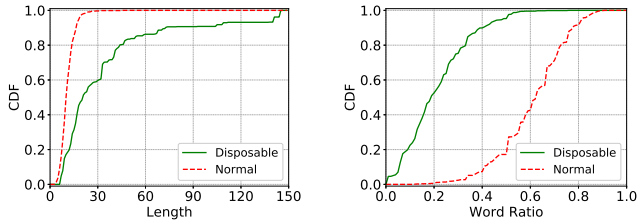
We enumerate six samples in Table I. Among them, three samples are disposable domains, and the remaining three are normal domains. It is clear that the subdomain names of disposable samples own obvious DGA-like appearances.

In addition to these typical disposable domains, we also find numerous normal domains that exhibit the identical traffic profile as disposable ones. As the fifth to eighth samples in Table I, these domains are mainly used for two services, namely blogs and e-commerce. In order to facilitate users to manage their homepages, some platforms provide each user with a unique subdomain name. However, the vast majority of homepages have fallen into the state of neglect, hence present the same query frequency as disposable domains.

Observing the subdomain name of each sample in Table I, we find that these two types differ greatly in character composition. The prefixes of disposable domains consist of arbitrary alphabetical and numerical characters, both verbose and unreadable. While the appearance of normal domains are exactly the opposite so that we can easily divide them into meaningful words. To verify the pervasiveness of such linguistic disparities between disposable and normal domains, we randomly select 900 samples from each type and calculate the linguistic metrics of their prefixes. Here we employ two widely-used metrics, length and word ratio. When calculating the word ratio, we apply an open source tokenizer that splits a string into tokens based on a preloaded dictionary [4]. Since a domain name includes not only alphabets but also numbers and hyphens, we further split all numbers and hyphens into separate tokens (e.g., 'hello-12' will be split into 'hello', '-', '1', '2'). Let $s$ be a subdomain, and $k_1, k_2, ..., k_n$ be the

TABLE I
REPRESENTATIVE DOMAIN NAME SAMPLES WITH ONLY ONE QUERY ON 04/27/2018.

| # | Subdomain Name | Domain Zone | IP Address | Word Ratio | Type |
|---|---|---|---|---|---|
| 1 | p4-aiee25cwbebug-nljfouh5mmh2zt5s-if-v6exp3-v4 | metric.gstatic.com | 216.58.200.* | 36.96% | Disposable |
| 2 | only-491874-116-5-86-20 | nstool.netease.com | 59.111.0.* | 34.78% | Disposable |
| 3 | a56e616288e6b61e33695d15ddeeb5a20.profile.jax1 | cloudfront.net | 13.32.241.* | 26.09% | Disposable |
| 4 | flowerqingse | lofter.com | 60.217.239.* | 75.00% | Normal |
| 5 | best5cooking | tumblr.com | 31.13.69.* | 75.00% | Normal |
| 6 | zhongjiliyu | fang.com | 124.251.87.* | 63.64% | Normal |




(a) Length Distribution. (b) Word Ratio Distribution.

Fig. 1. Linguistic metrics of disposable domains and normal domains.

token series of $s$. Denote $L(x)$ as the number of tokens in the sequence $x$. We define the word ratio $wr$ of a subdomain $s$ as $wr(s) = 1 - \frac{L(k_1,k_2,\ldots,k_n)}{L(s)}$. Naturally, a string with too many tokens is bound to have a lower word ratio. Fig. 1 presents the cumulative distributions of these metrics. For normal samples, almost all of their prefixes are less than 30 in length. But for disposable domains, only 60% of prefixes are shorter than 30. The difference in word ratio between these two types is more prominent. If we take 0.5 as a word ratio threshold, then more than 95% of disposable domains are below that, and more than 80% of normal domains are above that. These observations are easy to understand. To convey as much data as possible, disposable domains use certain compression algorithms, which makes the subdomains lengthy and unreadable. While normal domains are designed to provide direct Internet services. They are always short and easy to memorize, thus prompting users to frequently visit them.

Taken the above analysis together, although the large subdomain scale and the nearly zero cache hit rate are the most two obvious characteristics of disposable domains from the macro point of view, they neither can be the decisive criterion to identify disposable domains. The key insight here is such domains are algorithmically generated. Therefore, we should probe deeply into the more underlying features when dissecting the disposable domains. Just like the methodology used in mining malicious DGA domains, we can distinguish the disposable domains from the normal ones by leveraging a series of linguistic features.

## III. OUR APPROACH

According to the aforementioned analysis results, we propose a stacking model, called Vogers, to identify disposable domains from raw DNS traffic. The framework of Vogers is shown in Fig. 2. This system consists of three modules: feature extractor, readability score assigner, and the final binary classifier. We will discuss these three modules in depth throughout the remainder of this section.

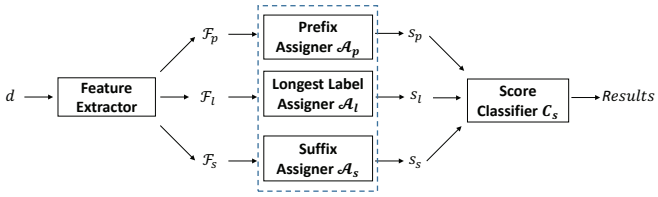



Fig. 2. Architecture of Vogers.

### A. Feature Extractor

This module takes the original domain names as input, outputs three feature vectors of such domain names. Before doing so, we should clarify what the three feature vectors represent for. It is known that the entire domain name of a malicious DGA domain is algorithmically generated. So previous works on the linguistic features of domain names all treat the entire domain name as a whole to analyze. While we find that, unlike malicious DGA domains, the auto-generation parts of disposable domains only exist in their prefixes, and the suffixes are the same as normal ones. Hence, there is a need to analyze the prefixes and suffixes of disposable domains separately. However, given a multilevel domain name, it is hard to tell which separate dot is the cut-off point between prefix and suffix. Thus, to simplify the process, we conservatively specify the 2LD as the suffix and the rest as the prefix.

Additionally, even if the prefixes of disposable domains are automatically generated, some meaningful words are explicitly introduced as identifiers in particular labels. Such identifiers dramatically increase the word ratio of the entire prefix. Thus, we should consciously amplify the impact of the auto-generated part on overall readability. Here, we choose to extract the longest label in the prefix as the 'attention signal' and put it on an equal footing with the prefix.

Given the above, we consider the readability of a domain name from three aspects, namely the prefix, the longest label in the prefix, and the suffix. Whereafter, we extract the linguistic features for each aspect which will be expounded in Section IV. For the convenience of presentation, we notate the prefix, the longest label in the prefix, and the suffix of a domain name $d$ as $dp$, $dl$, and $ds$, respectively. The feature vector is notated as $\mathcal{F}$. Finally, this module outputs three feature vectors, namely the feature vector $\mathcal{F}_p$ of $dp$, the feature vector $\mathcal{F}_l$ of $dl$, and the feature vector $\mathcal{F}_s$ of $ds$. Among them, $\mathcal{F}_p$ has 12 dimensions, $\mathcal{F}_l$ and $\mathcal{F}_s$ have 10 dimensions, respectively. These feature vectors are then fed into the next module.

### B. Readability Score Assigner

We construct three readability score assigners to learn the readability differences among different domain types in $dp$,

$dl$, and $ds$, which are notated as $\mathcal{A}_p$, $\mathcal{A}_l$, and $\mathcal{A}_s$, respectively. Each assigner takes a linguistic feature vector as input, and outputs a readability score between 0 and 1. The higher the score, the better the readability.

More specifically, each assigner is actually a binary classifier. In the training stage, we apply different training sets for each assigner in accordance with their distinct learning objectives. When training $\mathcal{A}_p$, the dataset is composed of the prefixes of disposable domains and normal domains. When training $\mathcal{A}_l$, we directly extract the longest label of each sample in prefix set to form the training set of this task. When training $\mathcal{A}_s$, we adopt the effective 2LDs from two authoritative public data sets. One is the Alexa Top List, which records the most popular benign domains. The other is the DGArchive [5], which provides the daily generated DGA samples. In each dataset, we construct 30,000 positive samples and 30,000 negative samples, respectively. The feature vectors associated with auto-generated parts are labeled as 0 (negative), while those associated with man-made parts are labeled as 1 (positive).

When the training reaches convergence, given a feature vector as input, the relative assigners will output the probabilities on both positive and negative categories. We take the probability value of the positive category as the final readability score. Finally, this module outputs three scores, namely $s_p$, $s_l$, and $s_s$, that correspond to the readability of $dp$, $dl$, and $ds$, respectively.

### C. Score Classifier

So far, we have obtained the readability scores of the three representative parts of domain names. To achieve the final detection target, we aggregate the three output readability scores into a 3-dimensional feature vector through stacking ensemble. Next, we construct a binary classifier, notated as $\mathcal{C}_s$, which takes the aggregated feature vectors as input, and outputs the final discrimination results. To train this classifier, we carefully select 10,000 disposable domains as negative samples (labeled as 0) and 10,000 normal domains as positive samples (labeled as 1). To avoid mistakenly classify malicious DGA domains as disposable domains, we additionally add 5,000 DGA domains into the positive sample set. All of these samples have been converted into 3-dimensional feature vectors through the first two modules. It is important to claim that the additional selection of malicious DGA domains as positive samples does not mean the readability of malicious DGA domains is similar to that of normal domains. What we actually aim to emphasize is the readability discrepancy between prefixes and suffixes of the disposable domains.

## IV. Features

In this section, we craft a series of linguistic features to quantify the so-called readability. Inspired by the ideas in [6], [7], we construct 12 basic linguistic features:

- **(F1) Length.** As exemplified in Fig. 1(a), to carry as much information as possible, the auto-generated parts are naturally longer than the man-made parts.
- **(F2) Label Level.** Since both $dl$ and $ds$ are one of the labels in domain $d$, we only employ this feature in $dp$. Besides, we consider both separate dots and hyphens as the division point of labels. As shown in Table III, the subdomain of disposable domains tend to have more division points, compared with the normal domains, and thus have higher label levels.
- **(F3) Average Label Length.** This feature is equal to the total length of labels divided by the label levels. Because both $dl$ and $ds$ have only one label, this feature exclusively serves for the prefix $dp$.
- **(F4 - F6) Vowel Ratio, Consonant Ratio, and Digit Ratio.** These three features reflect the ratio of vowels, consonants, and digits in a given string, respectively. The proportion and position of vowels and consonants in a word determines how it is pronounced. Only a domain with appropriate vowel-consonant ratio can be remembered more easily and queried continuously.
- **(F7 - F8): Consecutive Consonants Ratio, Consecutive Digit Ratio.** If a string of length greater than two is composed of pure consonants (digits), we consider that string to be consecutive consonants (digits). These two features calculate the proportion of consecutive consonants and consecutive digits in the given input, respectively.
- **F9: Character Cardinality.** This feature refers to the number of distinct characters in the string. The longer and more unreadable a string, the higher this feature is.
- **F10: Inner digits number.** Algorithmically generated strings are always alternately composed of alphabets and digits. None of the above features, however, can reveal this positional relationship between alphabets and digits. We introduce this feature to reflect the positional relationship in an auto-generated string. To this end, we first extract all the consecutive digits from a given string. Next, we judge whether the two characters adjacent to these substrings are alphabets or hyphens.
- **F11: Word ratio.** We employ this feature to explicitly reflect the proportion of meaningful words in a given string. The specific calculation method of word ratio has been introduced in Section II.
- **F12: Entropy.** This feature reflects the uncertainty degree of the given string. The specific calculation method of this feature can be referred to Shannon entropy.

## V. Evaluation

This section presents the experiments that we conduct to evaluate the performance of Vogers. First, we discuss the problem of classifier selection. Next, we compare Vogers with the prior art.

### A. Classifier Selection

There are two modules apply classifiers in Vogers. One is the readability score assigner, and another is the score classifier. Among them, the former uses three classifiers in parallel, and the latter needs one. In this subsection, we expound the most suitable classification model in each module.

TABLE II
PERFORMANCES OF THE CLASSIFICATION MODELS IN EACH SCENARIO.

| Module | RF | | | GBDT | | |
|---|---|---|---|---|---|---|
| | AUC | TPR | FPR | AUC | TPR | FPR |
| $\mathcal{A}_p$ | 0.998 | 0.983 | 0.020 | 0.999 | 0.981 | 0.012 |
| $\mathcal{A}_l$ | 0.997 | 0.972 | 0.028 | 0.998 | 0.975 | 0.018 |
| $\mathcal{A}_s$ | 0.996 | 0.995 | 0.032 | 0.998 | 0.996 | 0.034 |
| $\mathcal{C}_s$ | 0.999 | 0.995 | 0.003 | 0.999 | 0.998 | 0.001 |

*1) Readability Score Assigner:* As mentioned in Section III-B, we have prepared 30,000 positive samples and 30,000 negative samples for each assigner. For $\mathcal{A}_p$ and $\mathcal{A}_l$, we single out 100 disposable zones and 100 normal zones, then extract 300 domains from each of them. As for the 300 samples in each zone, 200 are used for training and 100 are used for test. With regard to $\mathcal{A}_s$, we skip the selection of domain zones and directly extract 30,000 samples from Alexa Top List and DGArchive, 20,000 for training and 10,000 for test.

We conduct experiments to select the best performing classification model for each assigner. Here we select three commonly used classification models as candidates, namely random forest (RF), gradient boosting decision tree (GBDT). The task of these assigners is to assign reasonable readability scores for related domains based on the input feature vectors. We therefore do not have to pursue excessive accuracy. We evaluate the AUC values of RF and GBDT, and their TPR values and FPR values at the threshold of 0.5. The evaluation results of them are listed in Table II. Compared with RF, GBDT always has higher AUC values and lower FPR values. Hence, we decide to employ GBDT as the classification model of the three readability score assigners.

*2) Score Classifier:* This module takes the three readability scores as input and outputs the final classification results. Compared with the readability score assigners, this module focuses more on accuracy. We extract 14,000 disposable samples and 14,000 normal samples from the training set used for $\mathcal{A}_p$. Moreover, we add 5,000 malicious DGA samples into this set. Here, the disposable domains are labeled as 1, and the others are labeled as 0. Next, we divided this set into two parts, 80% for training and 20% for test. Similarly, we use RF and GBDT as the candidate classification models for this module. The metrics in Table II show that, with the same AUC value of 0.999, the FPR value of GBDT is only 0.001 when the threshold is set to 0.5, which is 0.002 lower than that of RF. We therefore choose GBDT as the score classifier.

### B. Comparision with the Prior Art

We reconstruct the methods proposed in [3], and compare its performance with Vogers in the dataset used in Section V-A2. We use AUC, TPR, and FPR to evaluate the performance of these methods. The threshold is still set to 0.5 when calculating the values of TPR and FPR. The specific evaluation results are shown in the Initial column of Table III, from which we can clearly see that Vogers has an overwhelming superiority in detecting disposable domains. Compared with the prior method, Vogers improves the AUC value by at least 0.0588 and lowers the FPR value by more than 0.19.

TABLE III
COMPARISON WITH PRIOR ART.

| Data Source | AUC | TPR | FPR |
|---|---|---|---|
| Vogers | 0.9997 | 0.9898 | 0.0011 |
| [3] | 0.9409 | 0.9857 | 0.2001 |

Correlating the poor performance of Zone Miner and the analysis results in Section II, we can draw a tentative conclusion that neither cache hit rate nor subdomain scale can fundamentally characterize the disposable domains. The primary reason is that the vast majority of DNS traffic is monopolized by several hot domain zones. Although a disposable domain will only be queried once, the traffic profile it shows is almost the same as many unpopular domains, especially the aforementioned blog-like and e-commerce domains. What substantially dominates Vogers is it grasps the essence of disposable domains that the subdomains of disposable domains are auto-generated. Compressing the data into a fixed-length string via certain algorithms inevitably makes the string unable to satisfy regular lexical rules. For transmitting as much information as possible over a finite domain length, the producers have to compress the data in bulk, making the corresponding domain zones full of garbled subdomains.

## VI. CONCLUSION

We discuss the issue of disposable domains in this paper. These auto-generated domains are employed by some Internet services to gather transient information from clients. We propose a linguistics-based stacking model, Vogers, to identify such disposable domains. The evaluation results show that Vogers achieve the true positive rate of 98.9% in detecting disposable domains. In addition, compared with the prior art, it reduces the false positive rate by more than 19%.

## ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (2016YFB0801305, 2018YFB0804704) and Strategic Priority Research Program of the Chinese Academy of Sciences (XDC02030100). The corresponding authors are Tianning Zang and Xunxun Chen.

## REFERENCES

[1] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan, "On the responsiveness of dns-based network control," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. ACM, 2004.
[2] V. Paxson, M. Christodorescu *et al.*, "Practical comprehensive bounds on surreptitious communication over dns," in *22nd USENIX Security Symposium*, 2013.
[3] Y. Chen, M. Antonakakis, R. Perdisci, Y. Nadji, D. Dagon, and W. Lee, "Dns noise: Measuring the pervasiveness of disposable domains in modern dns traffic," in *44th DSN*. IEEE, 2014.
[4] "Jieba text segmentation," https://github.com/fxsjy/jieba, 2014.
[5] "DGArchive." https://dgarchive.caad.fkie.fraunhofer.de.
[6] S. Schüppen, D. Teubert, P. Herrmann, and U. Meyer, "Fanci: Feature-based automated nxdomain classification and intelligence," in *27th USENIX Security Symposium*, 2018.
[7] S. Yadav, A. K. K. Reddy, A. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010.