Pontus: A Linguistics-based DGA Detection System

Dingkui Yan[*†] Huilin Zhang[‡] Yipeng Wang[*†] Tianning Zang[*†] Xiaolin Xu[*‡] Yuwei Zeng[*†]
[*]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: xxl@cert.org.cn, {yandingkui, wangyipeng, zangtianning}@iie.ac.cn

Abstract—**Many botmasters use domain generation algorithms (DGA) to generate a host of malicious algorithmically-generated domains (mAGDs) and then choose several mAGDs for actual command and control (C2) communication. The botmasters use different seeds (e.g., timestamp) to generate different mAGDs, which makes the communication mechanism resilient to blacklisting. Thus the botmasters can hide C2 channels very well. If we can quickly detect these mAGDs from DNS traffic, we will effectively block the communication. In this paper, we propose a novel system, called Pontus, to detect mAGDs from DNS traffic. Pontus extract features exclusively from the individual domain names. We compare Pontus with the state-of-the-art system and find that Pontus improves the precision by at least 4.7%.**

Index Terms—**Domain Generation Algorithm, DNS traffic, Cybersecurity**

I. Introduction

A. Background and Motivation

In a botnet, an attacker (the botmaster) remotely controls a group of malware-compromised machines (bots) through a Command and Control (C2) communication channel [1]. To contact the botmaster, bots often access a hard-coded domain name or IP address of the C2 server to receive commands from the botmaster. This way has an apparent drawback that cybersecurity workforces can take down the entire botnet when they find the fixed domain name or IP address. To hide C2 servers effectively, many advanced botnets, such as Mirai [2], have adopted domain generation algorithms (DGAs) as a new communication mechanism. In this mechanism, bots and the botmaster periodically execute the same domain generation algorithm with the same seed (e.g., timestamp) to produce a list of candidate malicious domain names. The botmaster will choose several domain names to register. Bots query all these malicious algorithmically-generated domains (mAGDs) until one of the domain names can be resolved to a valid IP address of C2 server, and then they can establish connections with the botmaster. Soon the attacker abandons the domain names and IP addresses to use mAGDs generated by new seeds. Therefore it is very difficult to discover the mAGDs for actual C2 communication and block the botnets in time. That is why many kinds of botnets still exist on the Internet. In this paper, we devise a DGA detection system only based upon the string of a domain name, and the system can effectively identify mAGDs from domain name system (DNS) traffic.

B. Limitation of Prior Art

The deep understanding of DGA detection is a vital problem regarding cybersecurity. The research interests in this field have lasted over the last decade [3]–[6]. The most recent and relevant work is FANCI [3]. FANCI leverages supervised learning classifiers to detect mAGDs, and it yields a very high classification accuracy. However, we find FANCI has two essential defects.

First, FANCI has a high false positive in second-level domains(2LDs) and third-level domains(3LDs). A domain name is made up of a sequence of subdomains separated by dots. The right-most subdomain is called TLD and 2LD represents the two rightmost subdomains separated by a dot, and so on (e.g., *baidu.com* is a 2LD, *api.baidu.com* is a 3LD). FANCI adopts fully qualified domain names (FQDN) [12] and mAGDs which are 2LDs or 3LDs from DGArchive as training data. FANCI can easily differentiate them with the structural difference between them, such as the length of a domain name, the number of subdomains and so on. That way is effective in corporate or campus-grade networks because the quantity of 2LDs and 3LDs is small. But there are so many 2LDs and 3LDs in an ISP network that FANCI judges many benign 2LDs and 3LDs as mAGDs.

Second, FANCI has a poor performance on detecting wordlist-based mAGDs. We roughly divided all mAGDs into two categories: random-looking mAGDs and wordlist-based mAGDs. The character distributions of random-looking mAGDs look random because the mAGDs are generated by randomly selecting characters from an alphabet, hashing, or permuting an initial domain name. The wordlist-based mAGDs are concatenated by several words from a wordlist. Wordlist-based mAGDs follow the regular linguistic pattern because many benign domain names are concatenated by several words as well. For this reason, wordlist-based mAGDs are hard to be detected. The features of FANCI are not enough to reflect the difference between benign domain names and wordlist-based mAGDs so that FANCI cannot accurately distinguish wordlist-based mAGDs from benign domain names.

C. Proposed Approach

In this paper, we propose a DGA detection system, called Pontus, which is based upon powerful linguistics-based features. The features of Pontus are extracted exclusively from the individual domain name, Pontus still has a good classification performance.

Our system is based upon the key insight that benign domain names and mAGDs differ greatly in the linguistic aspect. Benign domain names often represent some specific meanings, such as a brand name, a person name. Those domain names usually adhere to the regular linguistic pattern for fluent reading or easy remembering. However, the random-looking mAGDs disobey regular linguistic pattern. Though wordlist-based mAGDs follow the regular linguistic pattern, they can be split into 2 or 3 words completely. Sometimes, the 2 or 3 words are separated by hyphens.

The input to Pontus is domain names, and the output is the label of each domain name, benign or mAGD. As shown in Fig.1, Pontus has two major phases: 1) Training phase and 2) Classification phase. The training phase uses labeled domain names to train a supervised classifier. The classification phase detects mAGDs from DNS traffic.

D. Key Contributions

We evaluate Pontus on datasets where benign domain names are collected on DNS recursive servers of two ISPs, and mAGDs are generated by 61 DGAs from the DGArchive [11]. The effectiveness of Pontus is measured with accuracy, precision, and recall. In sum, this paper has three key contributions:

1) We propose an ISP-scale DGA detection system. Our system only depends on linguistics features extracted from the individual domain name and doesn't require additional information, such as Whois. Our system still accurately detects mAGDs from massive and complex ISP-scale DNS traffic.
2) The features of Pontus precisely reflect the difference between benign domain names and wordlist-based mAGDs. Wordlist-based mAGDs can be split into several words. Thus we segment a wordlist-based AGD and then extract some features from words after segmentation. Thus Pontus has an excellent ability to detect wordlist-based DGAs.
3) Pontus has outstanding detection ability for mAGDs. It achieves high accuracy with 96.6%, precision with 96.7%, and recall with 96.5% while FANCI achieves accuracy with 92.4%, precision with 92%, and recall with 92.8% on the same datasets. Pontus outperforms FANCI significantly.

The rest of the paper is organized as follows. In Section II, we introduce the architecture of Pontus. In Section III, we elaborate on features employed in Pontus. We conduct the experiments to evaluate the performance of Pontus and compare the experimental results of Pontus to FANCI in Section IV. Finally, we conclude this paper in Section V.

II. Architecture of Pontus

Pontus is composed of the Training phase and the Classification phase. Fig.1 shows the architecture of Pontus.

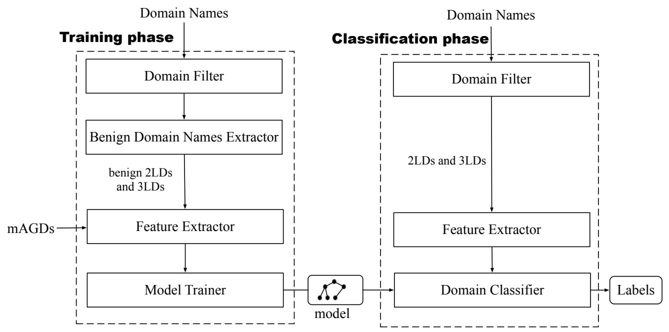


Fig. 1: Architecture of Pontus.

A. Training Phase

The Training phase is the foundation of this system. The phase consists of four modules, namely, Domain Filter, Benign Domain Names Extractor, Feature Extractor, and Model Trainer.

1) Domain Filter. Domain Filter aims to filter out the typical benign domain names and the invalid domain names. In this module, we filter out the domain names: a) domain names without invalid TLDs; b) domain names in the white list. The white list consists of domain names with Alexa top 100,000 domain suffixes [10], some CDN domain names, and reverse DNS domain names. c) existing mAGDs. d) nLDs (n>3). Because existing literature states that most of mAGDs are 2LDs and a few DGAs generate 3LDs which are usually dynamic domain names (e.g., ugrpuoo.dyndns.org) [7].
2) Benign Domain Names Extractor. Benign Domain Names Extractor obtains benign data for training classifier. First, we obtain the most frequently accessed 2LDs and 3LDs. Then we check these domain names on VirusTotal a reliable security analysis website [14]. VirusTotal collects many malicious domain names generated by malware. We remove domain names labeled malicious by VirusTotal to guarantee our benign data sets do not contain mAGDs as much as possible.
3) Feature Extractor. The input of the module is benign 2LDs and 3LDs extracted by the Benign Domain Names Extractor and mAGDs generated by 61 DGAs from DGArchive. The output is feature vectors and corresponding labels of the input domain names. Firstly, Pontus extracts features from the subdomains of a domain name without public suffix [9]. If the domain name has two subdomains, we concatenate the feature vectors of two subdomains. If the domain name has one subdomain, we will use a zero vector to replace the blank space. For instance, the feature of *api.baidu.com* is the combination of the feature vector of *api* and the feature vector of *baidu*, but the feature of *yahoo.com.cn* is made up of a zero vector and the feature vector of *yahoo*.
4) Model Trainer. Model Trainer uses the features and the labels from the Feature Extractor to train a model for the Classification phase.

B. Classification phase

The classification phase is the second phase of Pontus, which is mainly used to classify unlabeled domain names. The classification phase consists of three modules: Domain Filter, Feature Extractor, and Domain Classifier. In Domain Filter, we filter out domain names that have been identified as benign and mDGAs that have appeared. Then we extract features from the rest of 2LDs and 3LDs. Finally, we use the model mentioned in the Training Phase to detect mAGDs in the Domain Classifier modules.

III. Features

The features are extracted from a subdomain, not an FQDN. Simply, for *baidu.com*, Features are extracted from *baidu*, not *baidu.com*. We divide the features into three categories: distribution features, transition probability features, and words features. Distribution features focus on the character distribution of vowels, consonants, and digits. Transition probability features represent the average value of the transition probability between adjacent characters. Words features stand for the features extracted from words and intervals when a subdomain processes word segmentation. In Table I, we give an overview of all features with an example evaluation on the domain names d1 = ”grand-casino50.com” and d2 = ”trlekmynqihxxhy6k.com”, where d1 is a benign domain name and d2 is a mAGD.

TABLE I: Overview all features applied on d1 and d2

	#	Feature	d1	d2
A	1	The Length of Subdomain	14	17
	2	Number of Vowel	4	2
	3	Maximal Length of Consecutive Vowel	1	1
	4	Mean Length of Consecutive Vowel	1	1
	5	Ratio of Maximal Length of Consecutive Vowel to Number of Vowel	0.25	0.5
	6	Ratio of Vowel	0.286	0.118
	7	Number of Consonants	7	14
	8	Maximal Length of Consecutive Consonants	2	5
	9	Mean Length of Consecutive Consonants	1.4	3.5
	10	Ratio of Maximal Length of Consecutive Consonants to Number of Consonants	0.286	0.357
	11	Ratio of Consonants	0.5	0.824
	12	Number of Digits	2	1
	13	Maximal Length of Consecutive Digits	2	1
	14	Mean Length of Consecutive Digits	2	1
	15	Ratio of Maximal Length of Consecutive Digits to Number of Digits	1	1
	16	Ratio of Digits	0.143	0.059
	17	Ratio of Hyphen	0.071	0
	18	Is a Hexadecimal Digit	0	0
	19	Is a Decimal Digit	0	0
	20	Shannon Entropy	3.522	3.617
B	21	Average Transition Probability Based on Article	0.041	0.003
	22	Average Transition Probability Based on Alexa Top 100,000	0.055	0.015
C	23	Number of Words in AllWords	13	6
	24	Number of Maximum Matching Words	2	5
	25	Ratio of Length summation of Maximum Matching Words to Length of Subdomain	0.786	0.588
	26	Maximal Length of Words in Maximum Matching Words	6	2
	27	Mean Length of Words in Maximum Matching Words	5.5	2
	28	Number of IntervalWords	2	3
	29	Maximal Length of Word in IntervalWords	2	4
	30	Minimal Length of Word in IntervalWords	1	1
	31	Mean Length of IntervalWords	1.5	2.333
	32	Is Only Hyphen in IntervalWords	0	0

A. Distribution Features

Because many DGA generates domain names by randomly selecting characters from an alphabet, there will be much difference in character distribution between benign domain names and mAGDs. Hence, we can distinguish benign domain names from mAGDs by analyzing character distribution. All distribution features are shown in Table I-A. We only explain those features that are not easily understood.

(#18) Is a Hexadecimal Digit. If all character of the subdomain in **0-F**, we think the subdomain is a hexadecimal digit. In particular, In (**#12**, **#13**, **#14**, **#15**, **#16**, and **#19**), the digit refers to **0-9**.

(#21) Shannon Entropy. *S* is the 1-gram set of a domain name, *c* is a character in *S*, *P(c)* is the probability of *c*. *Shannon Entropy* = $-\sum_{c \in S} P(c) \times log_2(P(c))$.

B. Transition Probability Features

When a domain name follows the regular linguistic pattern, some character pairs of the domain name usually appear in other words or domain names because a character pair may be a syllable or a meaningful abbreviation (e.g., *Google, Yahoo, good* have the same character pair – *oo* because *oo* is a syllable.). According to existing works [8], we design the Transition Probability Features to state whether a domain name follows regular linguistic patterns. Firstly, we use a corpus to obtain the transition probability of a character appears when a character has appeared. Then we compute the average transition probability of all adjacent characters of a domain name. Algorithm 1 shows the details of extracting the Transition Probability Features. We execute the function named **getFeature** to get the value. That a character pair appears frequently in the corpus means there is a high transition probability from the former character to the later character. Therefore, the more frequently the character pairs of a domain name appear in the corpus, the value of Transition Probability Features is higher. All Transition Probability Features are shown in Table I-B.

(#21) Average Transition Probability Based on Article. We use an article [13] as the corpus to compute the transition probability of a domain name.

(#22) Average Transition Probability Based on Alexa Top 100,000. We use Alexa top 100,000 as the corpus to compute the transition probability of a domain name.

Algorithm 1 Transition Probability Features Extraction

```
Require:
 1: a text for statistics, corpus
 2: all characters will appear in a domain name, chars
 3: function BUILDMATRIX(corpus, chars)
 4:     $2\_gram\_list \leftarrow 2\text{-}gram(corpus)$
 5:     $matrix = array[chars.length][chars.length]$
 6:     for $[a, b]$ in $2\_gram\_list$ do
 7:         $matrix[a][b] += 1$
 8:     end for
 9:     for $a$ in chars do
10:         $rT \leftarrow 0$
11:         for $b$ in chars do $rT += matrix[a][b]$
12:         end for
13:         for $b$ in chars do
14:             $matrix[a][b] = \log(matrix[a][b]/rT)$
15:         end for
16:     end for
17:     return $matrix$
18: end function
19:
20: function GETFEATURE(corpus, chars, domain name)
21:     $matrix \leftarrow$ BUILDMATRIX(corpus, chars)
22:     $2gram\_list = 2\text{-}gram(domain\ name)$
23:     for $[a, b]$ in $2gram\_list$ do
24:         $log\_prob += matrix[a][b]$
25:     end for
26:     $result = e^{(log\_prob/2gram\_list.length)}$
27:     return $result$
28: end function
```

C. Words Features

The random-looking mAGDs often do not contain words and the wordlist-based mAGDs contain 2 or 3 words, which inspires us to extract features by segmentation and maximum match.

Firstly, we employ segmentation on the subdomain. We refer to all words contained in the subdomain as AllWords. Then we perform the maximum match operation. We partition the subdomain into words as long as possible in left-to-right order. We call these words as Maximum Matching Words. Meanwhile, we call all strings among the words in Maximum Matching Words as IntervalWords. All Words Features are shown in Table I-C.

Fig.2 clearly shows the segmentation and maximum match process applied to d1="grand-casino50.com". It is very intuitive for us to understand the meaning of definitions.

(#23) Number of Words in AllWords. The number of words hiding in the subdomain. For d1, the number is 13.

(#25) Ratio of Length summation of Maximum Matching Words to Length of Domain Name. The ratio of the total length of words in Maximum Matching Words to the length of the subdomain. For d1, the ratio=11/14=0.786.

(#26) Maximal Length of Words in Maximum Matching Words. The length of the longest word in Maximum Matching Words. For d1, the value is 6 (the length of 'casino').

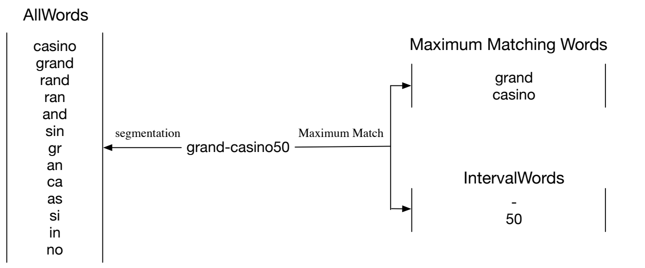


Fig. 2: segmentation process applied on d1=grand-casino50.com

(#29) Maximal Length of Word in IntervalWords. The feature is similar to **(#26)**. For d1, the length is 2(the length of '50').

(#32) Is Only Hyphen in IntervalWords. All strings in IntervalWords are hyphens. For d1, the value is 0 because '50' is not a hyphen.

IV. Evaluation

In this section, we present a series of experiments to evaluate the performance of Pontus. First, we introduce our data sets. Next, we state the process of classifier selection. Then, we design experiments on different data sets to test the generalization of Pontus. Finally, we compare our system with FANCI and point out our improvement.

A. Data Sets

In our data sets, we prudently consider 2LDs and 3LDs accessed most frequently and labeled benign in VirusTotal as benign domain names and find reliable and authoritative AGD source – DGArchive [11]. In short, our benign domain names come from the following sources:

- **Alexa Top List:** The list contains the most popular 1,000,000 domain names in the world. We need the top 100,000 as a corpus to extract features, so we just consider the domain names that rank from 100,001 to 1,000,000.
- **Raw Top List on China Telecom:** We count the frequency of domain names in the DNS traffic captured on a recursive DNS server of China Telecom that is an Internet service provider. After filtering out Alexa top 100,000 and some CDN domain names, We select the top 30,000 2LDs and 3LDs as benign domain names.
- **Raw Top List on China Mobile:** Similarity, the list contains the top 30,000 2LDs and 3LDs most queried in DNS traffic from the DNS servers of China Mobile. China Mobile is another Internet service provider.

Our malicious mAGDs come from DGArchive:

- **DGArchive:** So far, DGArchive collects 89 DGAs from now and provides mAGDs generated by these DGAs. Because some DGAs only generate a very small amount of mAGDs, we just select 61 DGAs. These DGAs either have more than 500 mAGDs or generate 3LDs.

We collect DNS traffic from ISP DNS servers during a week (from April 27, 2018 to May 3, 2018) and obtain the top list of every day. In other words, we have 7 data sets for each ISP. When we build the 7-day data sets, we randomly select 500 mAGDs from every mAGD and sample the same total number of benign domain names. If some DGAs don't generate 500 mAGDs, we select all mAGDs. We try to make mAGDs different for each dataset. Due to different user habits, only 40% of domain names are overlapping in the different ISP data sets.

B. Classifier Selection

We consider three classifiers as our candidate classifiers: gradient boosting decision tree (GBDT), support vector machine (SVM) and random forest (RF). Classification accuracy(**ACC**) and training time(**Time**) are two metrics that we mostly focus on. Accuracy means whether Pontus can find mAGDs to achieve our goal and training time reflects the usability in the real environment. In this section, we parallelly examine the performance of these classifiers on two data sets. The first data set is a one-day data set of China Mobile. Another data set comes from China Telecom. Table II presents the classification accuracy and training time of three classifiers. It is obvious that gradient boosting decision tree (GBDT) is the most suitable classifier no matter in classification capability or in training time. Hence, we use GBDT as our classifier in the next all experiments.

TABLE II: The Performance of Three classifiers

	GBDT		**SVM**		**RF**	
	ACC	**Time**	**ACC**	**Time**	**ACC**	**Time**
China Mobile	96.3%	88	95.8%	2832	96.2%	157
China Telecom	95.4%	87	94.6%	9482	95.2%	210

*The unit of training time is second.

C. Generalization

Next, we test the generalization of Pontus. On the one hand, we utilize the model trained in a data set to test the other data sets belonging to the same ISP. In this way, we can know whether Pontus generalizes well when DNS traffic changes with time. On another hand, we train a model in an ISP data sets and predict data sets of another ISP to verify generalization of our system when deployment environment changes.

Training and Prediction on the same ISP. We train a model on a data set of one day and then use the model to predict data sets of other days. We extract 80% domain names from a one-day data set. After filtering out the training domain names and balancing benign domain names and mAGDs in other data sets, we predict the rest of data to evaluate the generalization of Pontus. Table III shows the result of this experiment. In this experiment, Pontus still yields high classification accuracy. The mean accuracy is 96.6% in China Mobile and 95.5% in China Telecom. Besides, Pontus is robust because the standard deviation is very small in both networks.

TABLE III: The Generalization of Pontus in the Same ISP Data Sets

	acc_{max}	acc_{min}	$\overline{acc}$	σ
China Mobile	96.6%	95.9%	96.3%	0.001
China Telecom	95.8%	94.2%	94.8%	0.004

acc_{max}: The maximal classification accuracy.
acc_{min}: The minimal classification accuracy.
$\overline{acc}$: The average classification accuracy.
σ: The standard deviation classification accuracy.

Training and Prediction on different ISPs. In this experiment, we predict an ISP data sets by the model trained on different ISP data sets. We focus on whether Pontus generalizes well to different networks. Similar to the experiment above, we extract 80% domain names from a one-day data set then predict other data sets of another ISP. According to the result presented in Table IV, we find that no matter using the model trained in China Mobile to predict data sets of China Telecom or utilizing the model trained in China Telecom to predict data sets of China Mobile both have high mean accuracy with small standard deviations. Hence, Pontus can generalize well to different networks.

TABLE IV: The Generalization of Pontus in the Same ISP Data Sets

	acc_{max}	acc_{min}	$\overline{acc}$	σ
CM $\rightarrow$ **CT**	95.5%	94.5%	94.8%	0.003
CT $\rightarrow$ **CM**	96.5%	96.0%	96.2%	0.001

CM $\rightarrow$ **CT**: Training on China Mobile, Prediction on China Telecom
CT $\rightarrow$ **CM**: Training on China Telecom, Prediction on China Mobile

D. Comparisons With the Prior Art

Prior Art. FANCI is an outstanding DGA detection system, proposed by Schüppen *et al.* FANCI divides features into three categories: structural features (e.g., the length of a domain name), linguistic features (e.g., vowel ratio), and statistical features (2,3,4-gram distribution and Shannon entropy). The domain names in FANCI experiments vary greatly in structure. Thus the linguistic features of FANCI are not powerful but it is enough for FANCI. FANCI can't work well when faced with domain names that have similar structures with mAGDs. For the wordlist-based mAGDs, FANCI does not design features to reflect the specialty of this kind of mAGDs. In short, the features of FANCI need to be improved. For example, the benign domain name – *baiducdn.com* and the mAGD – *tytre9iu.ru* have the same number of vowel and their 2,3,4-gram distribution are similar, FANCI is hard to classify accurately. We will prove the preconception in the next comparison experiments.

Comparison. We design three experiments to compare the performance of Pontus and FANCI and point out our advancement.

- **Experiment I.** In the first experiment, we apply FANCI and Pontus to classify a data set of China Mobile and a data set of China Telecom. The accuracy, precision, recall of these two systems are shown in Table V. Table V shows that Pontus has superiority in detecting mAGDs. Though the proposed work only improves 4.7% precision than the previous work, Pontus will reduce the misjudgment of almost tens of thousands of domain names in an ISP network with massive traffic.

TABLE V: The Classification Results of Pontus And FANCI

	Accuracy		**Precision**		**Recall**	
	Pontus	**FANCI**	**Pontus**	**FANCI**	**Pontus**	**FANCI**
China Mobile	**96.6%**	92.4%	**96.7%**	92.0%	**96.5%**	92.8%
China Telecom	**95.4%**	91.1%	**95.0%**	90.3%	**95.9%**	92.0%

- **Experiment II.** In this experiment, we aim to verify the shortcoming of FANCI and the improvement of Pontus. We extract 20000 2LDs from Alexa top list and 20000 2LDs in mAGDs. These domain names are very similar because they have the near length, the same number of subdomain. In addition, a part of the domain name consists of abbreviations or is completely spliced by words. Therefore, distinguishing them is a difficult and core problem in the DGAs classification problem. Due to this samples have similar structures, if FANCI wants to distinguish these data, its features must be strong enough. In Table VI, FANCI has a poor performance on the data set, but Pontus still have over 96% accuracy when it is applied on the confusing domain names. The most critical aspect of identifying AGD is that the features can correctly reflect the generation schema of mAGDs. It is clear that FANCI did not grasp the essence of mAGDs but Pontus did it.

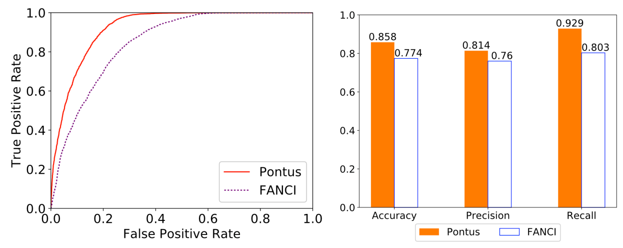
TABLE VI: The Classification Results of Two System on Alexa Data Set

	Accuracy	**Precision**	**Recall**
Pontus	**96.3%**	**96.7%**	**95.9%**
FANCI	89.4%	89.9%	88.6%

- **Experiment III.** This experiment is designed to demonstrate that our system has an outstanding ability to identify wordlist-based mAGDs. We choice *Gozi*, *SuppoBox*, *Matsnu* three wordlist-based DGA families to generate 30000 mAGDs and extract 30000 2LDs from Alexa top list as benign domain names. In the data set, we use 80% data to train models and left domain names for prediction. The ROC curves are shown in Fig.3(a) and the values of three metrics are shown in Fig.3(b). Obviously, the performance of Pontus is better than FANCI.

V. Conclusion

This paper proposes a novel DGA detection system, called Pontus, which determines whether a domain name is generated by a DGA. Further evaluations state that Pontus performances better than the state-of-the-art prior system,



(a) The ROC curves of Pontus and FANCI

(b) The Values of Three Metrics

Fig. 3: The Performance of Pontus and FANCI on Wordlist-based mAGDs

FANCI. Specifically, Pontus achieves on the accuracy of 96%, compared with 92% for FANCI on real DNS traffic. Meanwhile, Pontus has outstanding classification ability on wordlist-based mAGDs.

Acknowledgment

We thank the anonymous reviewers for their insightful comments. This work is supported by the National Key Research and Development Program of China under Grant No.2016YFB0801304, No.2016QY05X1002, No.2018YFB0804702, No.2018YFB0804704, and No.XDC02030100. The corresponding authors are Tianning Zang and Yipeng Wang.

References

[1] Antonakakis M, Perdisci R, Nadji Y, et al. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware[C]//USENIX security symposium. 2012, 12.
[2] Antonakakis M, April T, Bailey M, et al. Understanding the mirai botnet[C]//USENIX Security Symposium. 2017: 1092-1110.
[3] Schüppen S, Teubert D, Herrmann P, et al. FANCI: Feature-based Automated NXDomain Classification and Intelligence[C]//27th USENIX Security Symposium (USENIX Security 18). 2018: 1165-1181.
[4] Huang J, Wang P, Zang T, et al. Detecting Domain Generation Algorithms with Convolutional Neural Language Models[C]//2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). IEEE, 2018: 1360-1367.
[5] Wang T S, Lin C S, Lin H T. DGA botnet detection utilizing social network analysis[C]//Computer, Consumer and Control (IS3C), 2016 International Symposium on. IEEE, 2016: 333-336. s
[6] Schiavoni S, Maggi F , Cavallaro L , et al. Phoenix: DGA-Based Botnet Tracking and Intelligence[J]. 2014.
[7] Plohmann D, Yakdan K, Klatt M, et al. A Comprehensive Measurement Study of Domain Generating Malware[C]//USENIX Security Symposium. 2016: 263-278.
[8] Gibberish-Detector, Rob Neuhaus. 2015. [Online]. Available: https://github.com/rrenaud/Gibberish-Detector
[9] PUBLIC SUFFIX LIST,[Online]. Available: https://publicsuffix.org/list/
[10] The top 500 sites on the web, [Online]. Available: https://www.alexa.com/topsites
[11] DGArchive web site, [Online]. Available: https://dgarchive.caad.fkie.fraunhofer.de
[12] About fully qualified domain names, [Online]. Available: https://kb.iu.edu/d/aiuv
[13] How to Write a Spelling Corrector, [Online]. Available: http://norvig.com/spell-correct.html
[14] VirusTotal, [Online]. Available: www.virustotal.com