

# Detecting Algorithmically Generated Domains Using Data Visualization and N-Grams Methods

Tianyu Wang and Li-Chiou Chen

Seidenberg School of CSIS, Pace University, Pleasantville, New York

{*tianyu.wang, lchen*}@pace.edu

**Abstract**—Recent Botnets such as Kraken, Torpig and Nugache have used DNS based “domain fluxing” for command-and-control, where each bot queries for existence of a series of domain names and the owner has to register such domain name. Botmasters have begun employing domain generation algorithms (DGA) to dynamically produce a large number of random domains and select a small subset for actual use so that static domain lists ineffective. This article is to detect machine generated domain names; we tested common methods in classification on text strings of domain names has low accuracy. We introduced new features based on N-Grams in the classification methods and our experimental results show that the analysis of N-Gram methods can make a great progress in the accuracy of detection.

**Index Terms**— Classification Algorithms, Domain Name System, Network Security, Visualization

## I. INTRODUCTION

Many botnet detection systems use a blacklist of command-and-control (C&C) domains to detect bots and block their traffic. As a response, botmasters have begun employing domain generation algorithms (DGA) to dynamically produce a large number of random domains and select a small subset for actual use so that static domain lists ineffective. DGA is to be deterministic, yet generate a huge number of random domains so that bot maintainer only has to register one or few to enable the malware to work.

There is a trend that more recent botnets have used DNS based “domain fluxing” for command-and-control, where each bot queries for existence of a series of domain names, such as Conficker, Kraken and Torpig. This method is called DNS “domain fluxing”, which means each bot algorithmically generates a large set of domain names and queries each of them until one of them is resolved and then the bot contacts the corresponding IP-address obtained that is typically used to host the command-and-control (C&C) server [1] [2]. Besides, for command-and-control, spammers also routinely generate random domain names in order to avoid detection [3].

DGA stands for Domain Generating Algorithm and these algorithms are part of the evolution of malware communications. In the beginning, malware would be hardcoded with IP address or domain names and the botnet could be disrupted by going after whatever was hardcoded. The purpose of the DGA is to be deterministic, of which the bot maintainer only has to register one to enable the malware to phone home [4] [5]. If the domain or IP is taken down, the botnet maintainer with a new IP address can use a new name from the algorithm and the botnet maintained. Another major use case of detecting DGA is to protect non-authorized DNS servers, such as LDNS/ROOT-DNS.

The purpose of building a DGA classifier is not to take down botnets, but to discover and detect the use on our network or services. Furthermore, if we are able to have a list of domains resolved and accessed at one’s organization, it is possible to see which of those are potentially generated and used by malware.

This paper is organized as flows. In section 2, we discuss the background of domain names system and related security issues. We provide literature review in section 3. The DGA detection is presented in Section 4. We conclude the paper with our further research plan in section 5.

## II. BACKGROUND

### A. The Domain Name System

The Domain Name System (DNS) is a core component of Internet operation. It ensures the finding of any resource on the internet by just knowing the domain names of URL that is an easy way to remember.

### B. Domain Name Space

The naming system on which DNS is based is a hierarchical and logical tree structure called the domain namespace. Organizations can also create private networks that are not visible on the Internet, using their own domain namespaces.

As the following figure shows, the root of the domain name space is the “.” Node. The following figure shows a subtree of the domain name space and the path to the root. Every node is

This paper use the data from Alexa ranking list and DataDrivenSecurity dga dataset [20, 21].

Tianyu Wang is now a PhD candidate with the Department of Computer Science, Pace University, 861 Bedford Rd, Pleasantville, NY 10570 (e-mail: tianyu.wang@pace.edu).

Li-Chiou, Chen is the professor with the Department of Information System, School of Computer Science and Information Systems, Pace University, 861 Bedford Rd, Pleasantville, NY 10570 (e-mail: lchen@pace.edu).

called a level domain. Node at the base of the tree is called first level domains or Top Level Domains (TLD), for example, “edu”. Under the hierarchy, nodes are called second level domains (2LD), for example “email”, third level domains (3LD), etc.

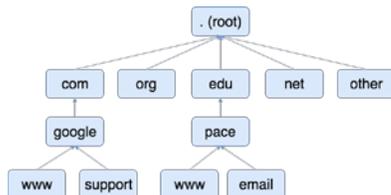


Figure 1. Domain Name Space Hierarchy.

### C. DNS Related Security Issues

DNS is often used to hide other kind of network traffic through the Internet. More specifically, there are many different DNS based misuse and malicious activities and related solving methods.

#### 1) DNS Fluxing

DNS fluxing is a series of activity that enhance the availability and resilience of malicious resources and contents by hiding the real location of a given resources within a network. The hidden resource is a server that delivers malware, phishing website or command and control server of a botnet (C&C).

Fast flux is one of the most common used DNS fluxing technique. It is used by botnets to hide phishing and malware delivery sites behind an ever-changing network of compromised hosts acting as proxies. It can also refer to the combination of peer-to-peer networking, distributed command and control, web-based load balancing and proxy redirection used to make malware networks more resistant to discovery and counter-measures. The Storm Worm (2007) is one of the first malware variants to make use of this technique [19].

The basic idea behind Fast flux is to have numerous IP addresses associated with a single fully qualified domain name, where the IP addresses are swapped in and out with extremely high frequency, through changing DNS records.

#### 2) Botnets

A botnet is a number of Internet-connected devices used by a botnet owner to perform various tasks. These botnets are groups of malware machines or bots that could be remotely controlled by botmasters. Botnets can be used to perform Distributed Denial of Service (DDoS) attack, steal data, send spam, and allow the attacker access to the device and its connection. The owner can control the botnet using command and control (C&C) software.

Botnets have become the main platform for cyber criminals to send spam, phishing and steal information, etc. Most of botnets rely on a centralized server (C&C). Bot could query a predefined C&C domain names that resolves IP address of server that malware commands will be received. Nowadays, in order to overcome the limitation that one single failure of C&C server is taken down, the botmaster would lose control over the botnet, C&C server have used P2P based structures in botnets, such as Storm, Zeus and Nugache [16, 17, 18]. To maintain a centralized P2P-based structure, attacker have developed a

number of botnet that locate their server through algorithms generated random domain names. The related algorithm is called domain generation algorithms (DGA).

#### 3) Domain Generation Algorithms (DGA)

Domain Generation Algorithms (DGA) is a series of algorithm that automatically generated domains names by given a random seed and then generate a list of candidate C&C domains. The botnet attempts to resolve these domains by sending DNS queries until one of the domains resolves to the IP address of a C&C server. This method introduces a convenient way to keep attacking resilience because if one domain names are identified and taken down, the bot will eventually get the valid IP address and using DNS queries to the next DGA domains. For example, Kraken and Conficker are some example of DGA-based botnets.

#### 4) DNS Monitoring

DNS service is widely used as a core service of the whole Internet. Monitoring the DNS traffic performs an important role. Globally the technique to identify flux networks and botnets using DNS analysis have been proved efficient. However, these techniques require previous know about fluxing domain names, since it rely on classification algorithms that need training on truth data. Another issue is these techniques require large amount of DNS replies from different locations so that to compute relevant features to train classification algorithms is not easy. The time taken by these methods to identify flux networks is too long. Finally, DNS based techniques for bot infected host detestation are involved with privacy concerns.

## III. RELATED WORK

Characteristics, such as IP addresses whose records and lexical features of phishing and non-phishing URLs have been analyzed by McGrath and Gupta [10]. They observed that the different URLs exhibited different alphabet distributions. Our work builds on this earlier work and develops techniques for identifying domains employing algorithmically generated names, potentially for “domain fluxing”. Ma, et al [9], employ statistical learning techniques based on lexical features (length of domain names, host names, number of dots in the URL etc.) and other features of URLs to automatically determine if a URL is malicious, i.e., used for phishing or advertising spam.

While they classify each URL independently, our work is focused on classifying a group of URLs as algorithmically generated or not, solely by making use of the set of alphanumeric characters used. In addition, we experimentally compare against their lexical features in Section V and show that our alphanumeric distribution based features can detect algorithmically generated domain names with lower false positives than lexical features. Overall, we consider our work as complimentary and synergistic to the approach in [8]. The authors [13] develop a machine learning technique to classify individual domain names based on their network features, domain-name string composition style and presence in known reference lists. Their technique, however, relies on successful resolution of DNS domain name query. Our technique instead,

can analyze groups of domain names, based only on alphanumeric character features.

With reference to the practice of “IP fast fluxing”, e.g., where the botnet owner constantly keeps changing the IP-addresses mapped to a C&C server, [12] implements a detection mechanism based on passive DNS traffic analysis. In our work, we present a methodology to detect cases where botnet owners may use a combination of both domain fluxing with IP fluxing, by having bots query a series of domain names and at the same time map a few of those domain names to an evolving set of IP-addresses. In addition, earlier papers [11], [8] have analyzed the inner working of IP fast flux networks for hiding spam and fraud infrastructure. With regards to botnet detection, [6], [7] perform correlation of network activity in time and space at campus network edges, and Xie et al in [14] focus on detecting spamming botnets by developing regular expression based signatures for spam URLs. M. Antonakakis present a new technique to detect randomly generated domains that most of the DGA-generated domains would result in Non-Existent Domain responses, and that bots from the same bot-net would generate similar NXDomain traffic [15].

#### IV. DGA DETECTION

##### A. Detection System

Classification in machine learning would help in DGA domains detection. The purpose of building a DGA classifier is not to remove botnets, but to discover and detect the use on our network or services. Furthermore, if we can have a list of domains resolved and accessed at one’s organization, it is possible to see whether there are potentially generated and used by malware.

Domain names are a series of text string, consisting of alphabet, numbers and dash sign. Therefore, it is common to use several supervised approaches to identify domains. Thus, the first step in any classifier is getting enough labeled training data. All we need is a list of legitimate domains and a list of domains generated by an algorithm.

##### B. Data Sets

###### 1) Alexa Domains

For legitimate domains, an obvious choice is the Alexa list of top web sites. The Alexa Top Sites web service provides access to lists of web sites ordered by Alexa Traffic Rank. Using the web service developers can understand traffic rankings from the largest to the smallest sites.

Alexa’s traffic estimates and ranks are based on the browsing behavior of people in our global data panel, which is a sample of all internet users. Alexa’s Traffic Ranks are based on the traffic data provided by users in Alexa’s global data panel over a rolling 3-month period. Traffic Ranks are updated daily. A site’s ranking is based on a combined measure of Unique Visitors and Page views. The number of unique Alexa users who visit a site on a given day determines unique Visitors. Page views are the total number of Alexa user URL requests for a site. However, multiple requests for the same URL on the same day by the same user are counted as a single Page view. The site with the highest combination of unique visitors and page views is ranked #1 [20].

However, the raw data grab from 1 Million Alexa domains are not ready for use. After we grab the top 1 Million Alexa domains (1,000,000 entries), we find that over 10 thousand are not domains but full URLs, and there are thousands of domains with subdomains that will not help. Therefore, after removing the invalid URL and subdomain and duplicated domains, we could have the clean Alexa data with 875,216 entries.

In this article, we only concentrate on the domains without top level. For example, www.google.com, we only use google as domain.

Table 1. First 5 Entries of Alexa data

	domain
0	google
1	facebook
2	youtube
3	yahoo
4	baidu

It is important to shuffle the data randomly for training/testing purpose and sample only 90% of total data. In addition, we put label for this Alexa dataset as ‘legit’. The number of Alexa domains: 787,694 out of the total Alexa domains 875,216.

###### 2) DGA Domains

On DataDrivenSecurity website, it provides file of domains and a high-level classification of “dga” or “legit” along with a subclass of either “legit”, “cryptolocker”, “goz” or “newgoz” [21]. These dga data are from recent botnets: “Cryptolocker”, two separate “Game-Over Zeus” algorithms, and an anonymous collection of algorithmically generated domains. Here we also resample 90% of the total data. Specifically, there are 47,398 out of 52,665 entries of algorithmically generated domains in our experiment. Here we also use domain names that without top-level parts.

Table 2. First 5 entries of dga domain

	domain	class
0	1002n0q11m17h017r1shexghfqf	dga
1	1002ra86698fjggqke1cdvbk5	dga
2	1008bnt1iekzdt1fqjb76pijxhr	dga
3	100f3a11ckgv438fpjz91idu2ag	dga
4	100fjpl1yk51751n4g9p01bgkmaf	dga

##### C. Basic Statistical Features

Now we need to implement some features to measure domain names. The domain field here means second-level domain only. In the following article, we use domains for abbreviation. The class field is binary category, either dga or legit. DGA stands for dynamic generated algorithms domain, and legit stands for legitimate domains.

###### 1) Length

First, we calculate the length of each domain. In the meantime, we drop those lengths that are less and equal to six, because for short domains, it is better use blacklist to filter out dga domains.

## 2) Entropy

Another feature is entropy of domain. In information theory, systems consist of a transmitter, channel, and receiver. The transmitter produces messages that are sent through the channel. The channel modifies the message in some way. The receiver attempts to infer which message was sent. In this context, entropy (more specifically, Shannon entropy) is the expected value (average) of the information contained in each message. This feature computes the entropy of character distribution and measure the randomness of each domain names.

The entropy can explicitly be written as

$$H(X) = \sum_{i=1}^n P(x_i) I(x_i) = - \sum_{i=1}^n P(x_i) \log_b P(x_i)$$

Table 3. Sampling first 5 entries with length and entropy

	domain	class	length	entropy
0	uchoten-anime	legit	13	3.392747
1	photoprostudio	legit	14	2.950212
5	andhraboxoffice	legit	15	3.506891
6	kodama-tec	legit	10	3.121928
7	pornubster	legit	11	3.095795

## D. Data Visualization

Before we begin our machine learning training, we plot scatter chart the check whether there is any correlation among the features.

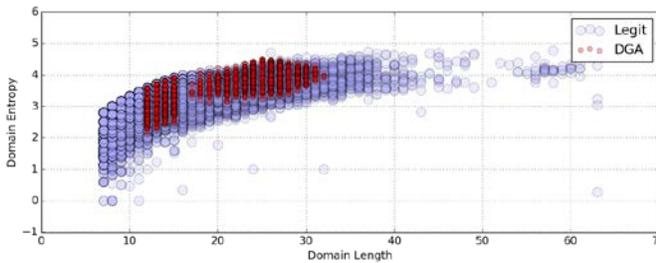


Figure 2. Scatter Plot: Domain Entropy vs Domain Length

In this figure, we found that legit domain and DGA domain are overlapped together. When domain length is approximately equal to four, DGA has a trend that has a higher entropy than Legit.

## E. Classification with Two Features

The next step is to run several classification methods use these two features (length, entropy). There are 787k legit and 47k DGA domains, so we use 80/20 split techniques for our training set and testing set. We choose to use three common supervised classification methods. Random Forest, Support Vector Machines (SVM) and Naïve Bayes.

Hypothesis:

- Positive: domain is dga
- Negative: domain is non-dga, in other words, legitimate domain

### 1) Using Random Forest Classifier

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks,

that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set

#### a) Random Forest Algorithms

A forest is the average of the predictions of its trees:

$$F(x) = \frac{1}{J} \sum_{j=1}^J f_j(x)$$

where  $J$  is the number of trees in the forest

For a forest, the prediction is simply the average of the bias terms plus the average contribution of each feature:

$$F(x) = \frac{1}{J} \sum_{j=1}^J c_j^{full} + \sum_{k=1}^K \left( \frac{1}{J} \sum_{j=1}^J contrib_j(x, k) \right)$$

#### b) Classifier Paramteres

Parameters	Values
The number of features (N)	2
The number of trees in the forest (n)	100
The number of features for the best split	$\sqrt{N}$
The minimum number of samples to split	2
The minimum number of samples at a leaf node	1

#### c) Classification Results

Predicted	dga	legit	All
True			
dga	2991	6379	9370
legit	427	127532	127959
All	3418	133911	137329

True Positive Rate (TPR) = 31.92%  
 False Negative Rate (FNR) = 68.08%  
 False Positive Rate (FPR) = 0.33%  
 True Negative Rate (TNR) = 99.67%  
 False Acceptance Rate (FAR) = 4.76%  
 False Rejection Rate (FRR) = 12.49%

The confusion matrix shows how our model predicts in classification using random forest classifier. The row is the true label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs not well. It identified dga domain as dga with only 31.92% accuracy (true positive rate). It misclassified dga domain as legit domain with 68.08% accuracy (false negative rate). Even it has a good prediction on true positive rate, which is 99.67%, the overall results in a biometric system is not good. False acceptance rate is 4.76% and false rejection rate is 12.48%. Therefore, the result of this method is not meet our requirement.

## 2) Using SVM Classifier

### a) SVM Algorithms

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm

builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

b) Classifier Parameters

Parameters	Value
Kernel	Linear
Penalty parameter C of the error term	1

c) Classification Result

Predicted	dga	legit	All
True			
dga	1160	8210	9370
legit	105	127854	127959
All	1265	136064	137329

TPR	FNR	FPR	TNR	FAR	FRR
12.38%	87.62%	0.08%	99.92%	6.03%	8.30%

The confusion matrix indicates how our model predicts in classification using SVM classifier. The row is the true label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs not well. It identified dga domain as dga with only 12.38% accuracy (true positive rate). It misclassified dga domain as legit domain with 87.62% accuracy (false negative rate). Even it has a good prediction on true positive rate, which is 99.67%, the overall results in a biometric system is not good. False acceptance rate is 6.03% and false rejection rate is 8.30%. Therefore, this method failed in classification.

3) Using Naïve Bayes Classifier

a) Naïve Bayes Algorithms

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

where  $P(c|X) = P(x_1) \times P(x_2) \dots P(x_n) \times P(c)$

- $P(c|X)$  is the posterior probability of class (c, target) given predictor (x, metric features)
- $P(c)$  is the prior probability of class
- $P(x|c)$  is the likelihood which is the probability of predictor given class
- $P(x)$  is the prior probability of predictor
- Naïve Bayes has no parameters to tune

b) Classification Result

Predicted	dga	legit	All
True			
dga	3332	6038	9370
legit	5061	122898	127959
All	8393	128936	137329

TPR	FNR	FPR	TNR	FAR	FRR
35.56%	64.44%	3.96%	96.04%	4.68%	60.30%

The confusion matrix indicates how our model predicts in classification using Naïve Bayes classifier. The row is the true

label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs not well. It identified dga domain as dga with only 35.56% accuracy (true positive rate). It misclassified dga domain as legit domain with 64.44% accuracy (false negative rate). Even it has a good prediction on true positive rate, which is 96.04%, the overall results in a biometric system is not good. False acceptance rate is 4.68% and false rejection rate is as high as 60.30%. Therefore, the classifier predicts unsuccessful.

Since these three models are not able to classify dga and legit domains successfully, we need to add more features to improve our model.

F. Model Improvement

We notice that dga domain either uses some random characters as text string or uses a dictionary to make up a new text string. Therefore, we build up our own corpus for these features.

1) N-Gram Features

If a domain is a legit domain, it more likely exists in the Alexa ranking list. Thus, it is necessary to find the similarity of legit domains. We could use some text analysis techniques. The first step is to build up a legit text corpus. Given a subsequence of domains, we summarize the frequency distribution of N-gram among the Alexa domain name string with  $n = [3, 5]$ . We called it Alexa\_grams matrix.

2) Alexa Gram

We calculate the similarity between every single domain and Alexa\_grams matrix. In order to calculate the similarity, we use some matrix transformation techniques to sum up the frequency. Furthermore, we normalize the frequency by log10 as a similarity score. (See Table 5.)

3) Dictionary Gram

We use a dictionary that contains 479,623 common used word terms [22]. The terms are combination of English vocabulary and common used words with mix of number and alphabet. We will use a words dictionary. After basic cleaning up work, the following is some basic discretions about the dictionary.

Similarly, we calculate the dictionary gram using N-gram,  $n = [3,5]$  and calculate the normalized similarity between words dictionary and every single domain. (See Table 5.) The reason why we choose  $n = 3, 4$  and  $5$  is because we have tested  $n = [1,10]$  and found  $n = 3, 4, 5$  have the best accuracy results.

Table 4. First 5 entries of words dictionary

	word
37	a
48	aa
51	aaa
53	aaaa
54	aaaaaa

Table 5. Sample of domain with Alexa grams and dictionary grams

domain	Alexa match	Dict match
google	23	14
facebook	42	27

pterodactylfarts	53	76
ptes9dro-dwacty2lfa5rrts	30	28

Now, we compute N-Gram matches for all the domains and add to our data frame.

Table 6. Calculated N-Gram for legit domains

domain	class	alexa_grams	word_grams
investmentsonthebeach	legit	144.721988	109.722683
infiniteskills	legit	81.379156	72.785882
dticash	legit	26.557931	23.710317
healthyliving	legit	76.710198	61.721689
asset-cache	legit	46.267887	31.690803

Table 7. Calculated N-Gram for dga domains

domain	class	alexa_grams	word_grams
wdqdreklqpp	dga	11.242176	6.367475
wdqjkpltirjhtho	dga	14.303602	16.554439
wdqxavemaedon	dga	28.468264	28.699800
wdraokbcnspexm	dga	25.935386	19.784933
wdsqfivqncbna	dga	4.597991	3.629002

#### 4) Data Visualization

Here we plot scatter about whether our new 'alexa\_grams' feature can help us differentiate between DGA and Legit domains.

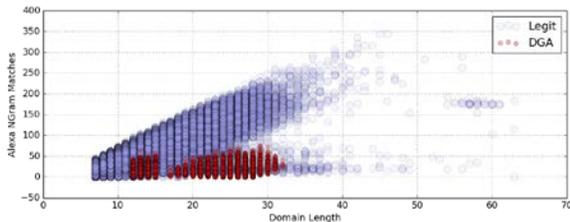


Figure 3. Scatter Plot: Alexa Gram vs Domain Length

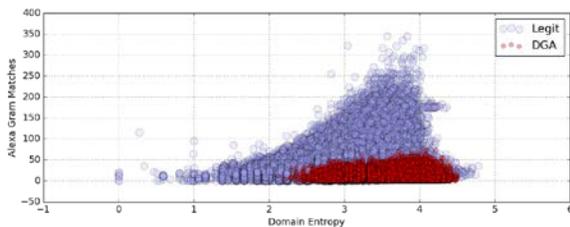


Figure 4. Scatter Plot: Alexa Gram vs Domain Entropy

Here we want to see whether our new 'word\_grams' feature can help us differentiate between Legit/DGA.

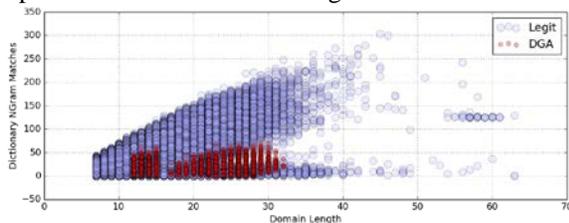


Figure 5. Scatter Plot: Dictionary Gram vs Domain Length

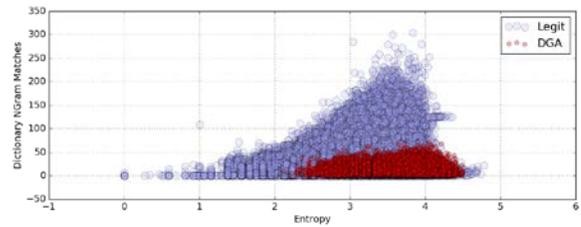


Figure 6. Scatter Plot: Dictionary Gram vs Entropy

After we add two extra features, the overlapped issue improved. We could have a clear view that legit, dga has their own clusters, and it is more reasonable to perform some classification methods once again.

#### 5) Classification with Four Feature

Now we have four features in our model: Length, Entropy, Alexa\_grams, and Dict\_grams. We could use the same parameters tuning our classification model.

##### a) Using Random Forest Classifier

Predicted	dga	legit	All
True			
dga	9139	231	9370
legit	254	127705	127959
All	9393	127936	137329

TPR	FNR	FPR	TNR	FAR	FRR
97.53%	2.47%	0.20%	99.80%	0.18%	2.70%

The confusion matrix indicates how our model predicts in classification using random forest classifier. The row is the true label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs pretty well. It identified dga domain as dga with 97.53% accuracy (true positive rate). It misclassified dga domain as legit domain as low as 2.47% (false negative rate). It has a good prediction on true positive rate, which is 99.80%, It also has low false positive rate which is 0.20%. The overall results in a biometric system is good as well. False acceptance rate is 0.18% and false rejection rate is 2.70%. Therefore, this method succeeds in classification.

##### b) Using SVM Classifier

Predicted	dga	legit	All
True			
dga	8623	747	9370
legit	534	127425	127959
All	9157	128172	137329

TPR	FNR	FPR	TNR	FAR	FRR
92.03%	7.97%	0.42%	99.58%	0.58%	5.83%

The confusion matrix indicates how our model predicts in classification using SVM classifier. The row is the true label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs pretty well. It identified dga domain as dga with 92.03% accuracy (true positive rate). It misclassified dga domain as legit domain as low as 7.97% (false negative rate). It has a good prediction on true positive rate,

which is 99.80%, It also has low false positive rate which is 0.42%. The overall results in a biometric system is good as well. False acceptance rate is 0.58% and false rejection rate is 5.83%. Therefore, this method succeeds in classification.

c) Using Naïve Bayes Classifier

Predicted	dga	legit	All
True			
dga	7203	2167	9370
legit	354	127605	127959
All	7557	129772	137329

TPR	FNR	FPR	TNR	FAR	FRR
76.87%	23.13%	0.28%	99.72%	1.67%	4.68%

The confusion matrix indicates how our model predicts in classification using Naïve Bayes classifier. The row is the true label, either dga or legit. The column is what our model predicted. Both the row and column has a total field indicate our sample size. The model performs pretty well. It identified dga domain as dga with only 76.87% accuracy (true positive rate). It misclassified dga domain as legit domain with 23.13% (false negative rate). It has a good prediction on true positive rate, which is 99.72%. It has low false positive rate, which is 0.28%. The overall results in a biometric system is not good. False acceptance rate is 1.67% and false rejection rate is 4.68%. Therefore, this method failed in classification.

6) Model Comparisons

Table 8. Model Comparisons

Performance Rate	Random Forest	SVM	Naïve Bayes
TPR	97.53%	92.03%	76.87%
FNR	2.47%	7.97%	23.13%
FPR	0.20%	0.42%	0.28%
TNR	99.80%	99.58%	99.72%
FAR	0.18%	0.58%	1.67%
FRR	2.70%	5.83%	4.68%

For true positive, true negative rate, the higher the better, because it means more accurate on our prediction. For false positive rate, true negative rate, false acceptance rate and false rejection rate, the lower the better, because it means the type I and type II error rates. Among all three models, Random Forest classifier outperforms the best. The reason that random forest performs the best is because random forest is a multi-layer decision tree. It will subgroup every details of features in a tree structure. The domain is a series of text string, and a tree structure classifier very easily captures the specific features of text string. However, linear SVM is trying to draw several straight line between the features of data. The scatter plot shows that we still have overlapped data among all the features so that the accuracy of SVM is not as good as random forest. The Naïve Bayes is a combination of conditional probabilities, and a single gram is not effective among text string.

We used this classifier as our prediction model. We also calculate the importance score on these four features. The importance of a feature is computed as normalized total reduction of the criterion brought by that feature.

Table 9. Importance Score on Random Forest

	Length	Entropy	Alexa_grams	Dict_grams
Score	0.2925341	0.21776668	0.36576691	0.1239323

We found that the most important feature in our model is Alexa\_grams. It indicates that Alexa ranking maintains a good contribution on dga classification. It proves our hypotheses that most of botnet masters are using dictionary or random characters to generate malicious domains. The second ranking is length of domain names followed by entropy and Dict\_grams. It indicates that more and more botnet masters are using some English words dictionary as their algorithms input. Our methods could also detect dga that using dictionary.

7) Misclassification

a) Educational Institution Domains

First, look at a piece of our prediction sample. The following table is an example of prediction using random forest as a classifier. It performs and predicts well except some university domain names. For example, tsinghua.edu.cn and sjtu.edu.cn are the domain names of university in China.

Table 10. Prediction sample

domain	prediction
google	legit
webmagnat.ro	legit
bikemastertool.com	legit
lcb8a5f36f	dga
pterodactylfarts	legit
pybmvodrcmkwq.biz	dga
abuliyen.com	legit
bey666on4ce	dga
sjtu.edu.cn	dga
tsinghua.edu.cn	dga

Table 11. Misclassification sample

domain	length	entropy	alexa_gram	word_gram	predict
duurzaamthuis	13	3.18083	20.353	17.785	legit
hutkuzwropgf	12	3.4183	14.240	10.431	legit
xn--ecki4eoz0157d hv1bosfom5c	28	4.28039	37.036	15.577	legit
nllcolooxrycoy	14	2.61058	31.160	26.914	dga
dk tazhqlzsnorer	15	3.64022	24.592	22.804	legit
eprqthyhoplu	12	3.25163	24.762	19.213	dga
domowe-wypieki	14	3.23593	28.051	24.537	legit
taesdijrndsaw	14	3.23593	30.930	21.647	dga
edarteprysytvhw	15	3.37356	36.684	29.358	dga
ukonehloneybmf	15	3.37356	39.44	36.303	dga
ekgzkawofkxzlq	14	3.32486	7.0389	5.4897	legit

For those legit domains but our model treat them as dga, some of legit domains come from foreigner countries. For example, domowe-wypieki comes from www.domowe-wypieki.com, which is a homemade pastries food website in polish. These countries use very different word and character system than those in English. In order to use English words in domain system, many of domains are adapted and made of some initial letters of approximately pronunciation of foreigner language. This is why some legit domain arise misclassification issue.

For those dga domains but our model regards them as legit, probably because Alexa ranking only summarize the unique visiting volume. Thus, there are still so many malicious and dga domain are among Alexa dataset.

#### b) Discussion

There are some potential ways to address those issues above and improve our model. First, we could set up a filter to sort the top-level domain (TLD) on those education and non-profit domains. In addition, for those foreign websites, we would try to figure out how these domains works and find a better legit dataset, except for Alexa. We could also use other dictionary such as Wiki keywords as our classifier features. At last, we plan to build up a self-adapted machine learning architecture that could learn from real-time DNS traffic, detect, and prevent those anomaly activities in our future research.

### V. CONCLUSION AND DISCUSSION

In this paper, we introduce the necessary about detection of DGA domains. In addition, we tested three common machine learning algorithms, random forest, SVM and Naïve Bayes, to classify legit and DGA domain names. We provide data visualization techniques with two new features, Alexa gram and Dictionary gram in classification experiment. At last, we found introducing NGram features would increase the accuracy of classification models and random forest classifier performs the best among all. We also found some issue using our methods and come up some ideas to solve the problem. We plan to improve our classification method and then setup our own DNS servers and build up two-engine network monitoring system. One is for machine learning training and model updating. The other one is for real-time monitoring for prevention.

### REFERENCES

[1] S. Yadav, A. K. K. Reddy, A. L. N. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," presented at the 10th annual conference, New York, New York, USA, 2010, pp. 48–61.

[2] S. Yadav, A. K. K. Reddy, A. L. N. Reddy, and S. Ranjan, "Detecting algorithmically generated domain-flux attacks with DNS traffic analysis," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, Oct. 2012.

[3] A. Reddy, "Detecting Networks Employing Algorithmically Generated Domain Names," 2010.

[4] Z. Wei-wei and G. Qian, "Detecting Machine Generated Domain Names Based on Morpheme Features," 2013.

[5] P. Barthakur, M. Dahal, and M. K. Ghose, "An Efficient Machine Learning Based Classification Scheme for Detecting Distributed Command & Control Traffic of P2P Botnets," *International Journal of Modern ...*, 2013.

[6] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-independent Botnet Detection. *Proceedings of the 17th USENIX Security Symposium (Security'08)*, 2008.

[7] G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. *Proc. of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, Feb. 2008.

[8] T. Holz, M. Steiner, F. Dahl, E. W. Biersack, and F. Freiling. Measurements and Mitigation of Peer-to-peer-based Botnets: A Case Study on Storm Worm. In *First Usenix Workshop on Large-scale Exploits and Emergent Threats (LEET)*, April 2008.

[9] S. S. J. Ma, L.K. Saul and G. Voelker. Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs. *Proc. of ACM KDD*, July 2009.

[10] D.K.McGrathandM.Gupta.BehindPhishing:AnExaminationofPhisher Modi Operandi. *Proc. of USENIX workshop on Large-scale Exploits and Emergent Threats (LEET)*, Apr. 2008.

[11] E. Passerini, R. Paleari, L. Martignoni, and D. Bruschi. Fluxor : Detecting and Monitoring Fast-flux Service Networks. *Detection of Intrusions and Malware, and Vulnerability Assessment*, 2008.

[12] R. Perdisci, I. Corona, D. Dagon, and W. Lee. Detecting Malicious Flux Service Networks Through Passive Analysis of Recursive DNS Traces. In *Annual Computer Society Security Applications Conference (ACSAC)*, dec 2009.

[13] M. Antonakakis, R. Perdisci, D. Dagon,W. Lee, and N. Feamster. Building a Dynamic Reputation System for DNS. In *USENIX Security Symposium*,2010.

[14] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulthen, and I. Osipkov. Spamming Botnets: Signatures and Characteristics. *ACM SIGCOMM Computer*.

[15] Manos Antonakakis, Roberto Perdisci, Yacin Nadjji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. 2012. From throw-away traffic to bots: detecting the rise of DGA-based malware. In *Proceedings of the 21st USENIX conference on Security symposium (Security'12)*. USENIX Association, Berkeley, CA, USA, 24–24.

[16] ZeuS Gets More Sophisticated Using P2P Techniques. <http://www.abuse.ch/?p=3499>, 2011

[17] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich. Analysis of the storm and nugache trojans: P2P is here. In *USENIX; login.*, vol. 32, no. 6, December 2007.

[18] Wikipedia. The storm botnet. [http://en.wikipedia.org/wiki/Storm\\_botnet](http://en.wikipedia.org/wiki/Storm_botnet).

[19] Prince, Brian (January 26, 2007). "'Storm Worm' Continues to Spread Around Globe". *FOXNews.com*. Retrieved 2007-01-27.

[20] Alexa ranking, <https://aws.amazon.com/alexa-top-sites/>

[21] Dataset collection, <http://datadrivensecurity.info/blog/pages/dds-dataset-collection.html>

[22] Data hacking, [http://clicksecurity.github.io/data\\_hacking/](http://clicksecurity.github.io/data_hacking/)