Detecting Malicious Fast-Flux Domains Using Feature-based Classification Techniques

Dinh-Tu Truong[1], Dac-Tot Tran[2], Bao Huynh[3]

[1] Natural Language Processing and Knowledge Discovery Laboratory, Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Vietnam

[2] The Faculty of Information Technology, Ho Chi Minh City University of Food Industry, Ho Chi Minh City, Vietnam

[3] Faculty of Information Technology, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Vietnam

truongdinhtu@tdtu.edu.vn, tottd@hufi.edu.vn, hq.bao@hutech.edu.vn

Abstract

In recent years, new generation botnets tend to use an evasion technique based on Domain Name System (DNS) called Fast-Flux Service Network (FFSN) to hide the actual location of their malicious servers. Detection of FFSN continues to be a challenging issue because of the similar behavior between FFSN and other legitimate infrastructures, such as Content Delivery Networks (CDNs) and Round Robin Domain Name System (RRDNS). In this paper, we present a novel approach based on analyzing the passive DNS traffic traces to detect malicious FFSNs. By analyzing DNS traces, we extracted ten key features and employed on the popular machine learning algorithms to build classifiers aim to classify a domain as either malicious flux service or legitimate. The seven among the ten features are first introduced in this study. The effectiveness of selected features is illustrated by comparing the distribution of 95% confidence interval for the mean and standard errors between legit, malware and fast-flux domain names on each feature. The statistical results show that there are discernible biases in the distribution of the feature values between benign and malicious domain names. The experimental results show that our proposed approach achieves the higher detection accuracy and lower false positive rate than the previous methods.

Keywords: Domain-flux, DGA-based botnet, Malicious domains, Botnet detection

1 Introduction

One of the most serious threats currently on the Internet is Botnet. In most large-scale coordinated cyber-attacks, botnets are recognized as the platform serving the attacks, such as it can be used to perform distributed denial of service (DDoS), send spam or steal data, etc. Detecting botnet is therefore of great importance and some security researchers have concerned about this threat and proposed many effective botnet detection approaches.

However, botnet developers are constantly developing new techniques in order to improve their bot and evade detection from security researchers. In recent years, new generation botnets tend to use a technique called Fast-Flux Service Network (FFSN) to hide the true location of their botnet servers. The main idea of FFSN is one or more domain names that are resolved to multiple (hundreds or even thousands) different IP addresses with a short Time-To-Live (TTL), and the rapidly change in DNS answers [1-2]. These IP addresses are chosen in a round-robin fashion from a pool of thousands addresses of the infected machines in the botnet, called flux agents, which the user host can connect to [3]. Each time the user host requests to visit a site (e.g., flux.example.com), it will reach one of the flux agents to process its requests. Instead of processing the request itself, it continues redirects the request to a different server, called mothership [4, 10]. The responses from mothership will be sent back to the flux agent. Finally, the user host will receive the result from the flux agent (Figure 1). It is very difficult to find mothership, because they are often hidden behind the flux agents [10]. Furthermore, the flux agents are often distributed in may different places on the networks [5].

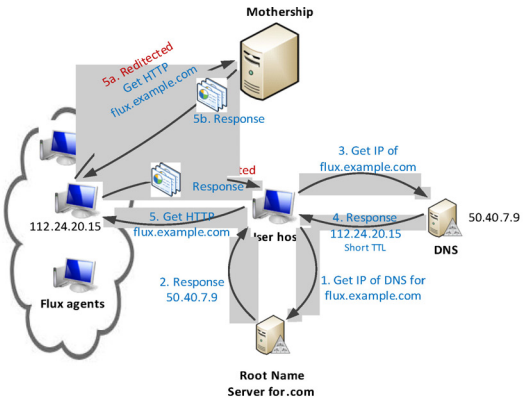


Figure 1. The basic idea of fast flux service networks

*Corresponding Author: Bao Huynh; E-mail: hq.bao@hutech.edu.vn
DOI: 10.3966/160792642020072104015

There are some approaches have been recently proposed to detect FFSN [5, 6-8, 9-22]. As previous mentioned, the characteristics of FFSN is one or more domain names are resolved to hundreds or even thousands different IP addresses (flux agents) and the rapidly change in DNS answers. The approaches that are proposed by [4, 13-15, 18, 21-22] use a small amount of active DNS traffic traces, so it cannot obtain multiple IP addresses associated with malicious fast flux networks. This disadvantage may enhance false positive and false negative rates. However, if the passive DNS replication method is installed, this limit can be overcome. We can reconstruct a partial view of the avaiable data in DNS by implementing DNS replication into DNS log databases. Based on the DNS Log databases we can answer questions such as where did this domain name point to in the past? What domain names point into a given IP network? What domain names are hosted by a given name-server? What subdomains exist below a certain domain name? etc.

By analyzing passive DNS traffic traces, we extract ten key features and employ on the popular machine-learning algorithms to build classifiers aim to classify a domain as either malicious flux service or legitimate/non-flux service.

To summarize, in this work we make the following contributions:

- We develop a tool based on Passive DNS [23] to collect DNS records passively from a network interface. From a large the number of the raw DNS data obtained, we define a DNS data aggregator aim to facilitate tracking and management of the DNS query/response information related to each domain.
- We introduce to extract ten key features based on the obtained DNS traffic data to train classifiers that be able to distinguish legitimate domains from malicious fast-flux ones. The seven features among the ten introduced features are first introduced in this study. The effectiveness of selected features is illustrated by comparing the distribution of 95% confidence interval for the mean and standard errors between legit, malware and fast-flux domain names on each feature. The statistical results show that there are discernible biases in the distribution of the feature values between benign and malicious domain names.
- We experiment on various MLAs and show that Random Forest is the best classifier to determine whether a domain name is malicious or legitimate. We also evaluate the effectiveness on the Recorded Dataset, the detection rate that we observed is similar to the detection rate estimated by the percentage split and cross-validate evaluations on the training set.

The remaining of this paper is organized as follows: Section 2 reviews a number of related works. The proposed approach, system architecture and the methods to extract features are presented in Section 3. The experimental results and evaluation are presented in Section 4, and finally, we conclude this paper in Section 5.

2 Related Works

The issue of fast flux networks was reported for the first time by the Honeynet project [1]. Although Fast Flux attacks certainly cause more difficulty to be detected, there have been also many related literatures about detecting Fast Flux.

To detect malicious Fast Flux domains, Holz et al [4] proposed to use features: number of unique NS records, number of unique A records and number of unique autonomous numbers (ASN). These features derived from several active DNS queries. They proposed a method of weighed linear regression to assign a flux-score to a domain for classifying Fast-flux domains.

Ammar Almomani [8] developed a system called the fast-flux hunter to detect unknown “zero-day” online fast-flux botnets. This system supported an evolving fuzzy neural network algorithm to enhance learning from inherent features of the FFSNs to distinguish the fast-flux botnet domain from legitimate domains. All the features are converted into digit numbers and ranked it to identify the most effective features in FFSN.

Passerini et al. [22] pointed out that TTL is one of important features in detecting a fast-flux domain. Personal registration information which is detected from a fast-flux domain can be used in Passerini et al.’ classification to find malicious domains, because the victim’s personal information or randomly generated domain names are often used by hackers to register malicious domain names.

Chen et al. [6] proposed a method to detect malicious FFSN based on LSTM network. They studied to use three characteristics as input of LSTM model. First characteristic is related to the conversion of domain name characters. Second characteristic is related to empirical information. Third characteristic is related to geographical and time. CDN servers in a content distribution system are often deployed in many different geographical areas. When users in different geographic areas query the domain name, the nearest server will handle it. The FFSN system has a quite small number of IPs dedicated to C&C server, it does not have distributed deployment capability [6]. Therefore, the FFSN system does not have the geographical discrimination characteristic of the resolution result, while the CDN has the geographical characteristic of the resolution result [6].

In addition to some of our proposed features are F2, F7, F9, F10, which are a bit similar to those suggested by Chen et al. [6], we also suggest some other features that can clearly distinguish between FF and CDN domain name. Because it is difficult to obtain HTTP

traffic due to privacy policy, our work does not access to HTTP traffic generated by individual customers. We only focus on privacy-preserving passive analysis of DNS traffic. This study proposes using ten features from DNS responses to develop a detection model.

3 Proposed Methods

3.1 System Overview

Figure 2 shows an overview of our detection system. The system's input is a stream of DNS traffic that is collected through Security Information Exchange (SIE) [24]. For each passive DNS sensor, the DNS queries/responses from/to the users' machines are monitored in a predefined period (e.g. One day), and store raw DNS traffic information in a database (DNS logs). In order to facilitate storage and access from the DNS logs, it is essential to build a data structure that is able to easily access to the fields of raw DNS data.

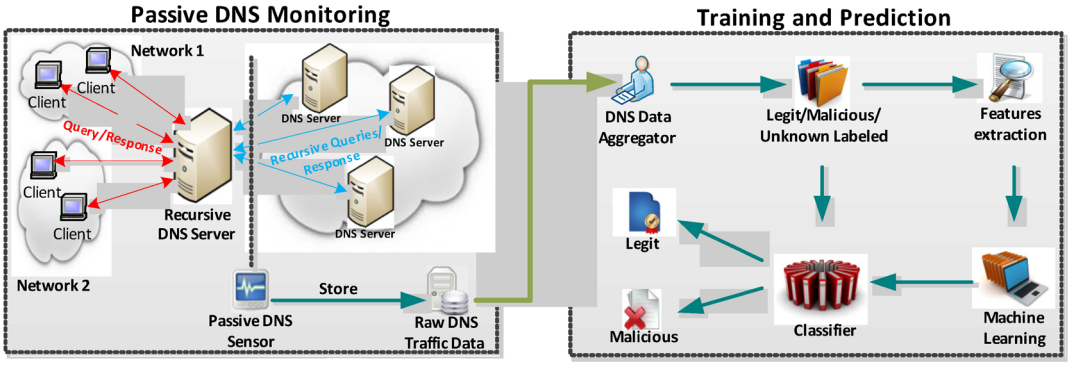


Figure 2. Overview of the proposed detection system

However, the amount of DNS traffic has been often overwhelming. Therefore, in order to reduce the analyzed traffic, we apply some filtering rules. All DNS messages related to d domain are assemblaged into a higher-level DNS message named *Q(d)* (more details will be described in Section 3.2).

The *Data Pre-filtering* module: This module is responsible for filtering out messages that are not alike flux domains. The main function of this module is to reduce the computational cost of the system and reduce the volume of the data (more details will be described in Section 3.3).

The feature extraction module performs extracting DNS features. By examining a large amount of DNS data, we identified 10 different features, four of which were used in previous research [25], the remaining six features is first proposed in this study. Our principle for selecting these features are explained in detail in Section 3.4.

Finally, some popular machine learning algorithms are applied to train the models and automatically classify domain names as legitimate or malicious (more details are described in Section 4).

3.2 Data Aggregator

In order to facilitate the description of the statistical features that our system extracts from DNS traffic data, the essential information must be gathered and expressed in a summary form. First, how DNS queries and related responses work should be defined. Suppose $D = \{d_1, d_2, \ldots, d_n\}$ is a set of n domains. Let $Q(d_j)$ is an aggregate of DNS queries/responses related to the domain d_j, which is performed by users during a given interval of time Δ. The information in the queries and its related response is aggregated by formula 1:

$$Q(d_j) = (d_j, c_j, I_j, T_j, S_j, t_1, t_2) \quad \textbf{(1)}$$

where d_j is the queried domain name; c_j is the number of DNS queries/responses for domain d_j; T_j is the set of Time-To-Live (TTL) of the DNS responses. I_j is the cumulative set of distinct resolved IP addresses as request/response to domain d_j; t_1 is timestamp of the first seen and t_2 represent timestamp of the last seen regarding d_j. S_j is the set of all subdomains of the domain d_j. In other words, $Q(d_j)$ includes information such as the number of DNS queries to the domain d_j during Δ time; the number of the IP addresses mapped to d_j; *time-to-live* (TTL), etc.

Summing up, after the data aggregating procedure, we have a dictionary of domains, where domain d_j is key and c_j, I_j, T_j, S_j, t_1, t_2 are values.

3.3 Data Pre-filtering

In large networks, recursive DNS traffic is often very large. Therefore, we must first apply some

filtering rules to remove DNS that is clearly related to non-flux domain so that can reduce the volume DNS traffic.

The fast-flux domains often have the following main features: (1) the time-to-live of DNS resources records is very short; (2) The high rate of changes of IP sets that is responded by each query; (3) There are a large number of IP addresses that can be resolved by querying the same domain name over time; and (4) the resolved IP addresses are usually scattered across many different networks [1].

We use the filtering policy as suggested by Perdisci et al [25] to select candidate flux domains with the following conditions:

- Time-To-Live of a domain must be less than 3 hours (i.e., TTL<=10800 s).
- Minimum number of distinct resolved IP addresses for the domain (i.e., $|I_j| >= 3$, or if a set of resolved IPs $|I_j| < 3$, must have a very low TTL <=30 seconds).
- Diversity of networks for a domain (i.e., *entropy(prefix(-I_j))*>=0.333). The resolved IP addresses with the same prefix /16 are likely to belong to the same network. Diversity is calculated as entropy of /16 prefix networks in I_j for a given domain d_j.

However, not all these candidate flux domains are actually malicious flux domains. There are also some legitimate services, such as CDNs or RRDNS are served through sets of domain names that share some similar properties with flux domains.

The domains related to legitimate CDNs often have a very low TTL and resolve to multiple IP addresses located in many different networks [1]. For instance, *weibo.com*, a legitimate CDN domain has a short TTL (60 seconds) and constantly changes its A record IPs, resulting in the accumulation of almost 117 IP addresses during our monitoring period. It is difficult to identify and distinguish malicious fast-flux domains if only using each individual feature, but if combined with many features it will allow to detect malicious fast-flux domains more exactly.

Summing up, a list of candidate flux domains and their related DNS information is considered as the output of this module. At the end of each candidate flux domain, we measure the features described in Section 3.4 to train classifier aim to automatically classify a domain as either legitimate or malicious flux.

3.4 Feature Extraction

Selection of discriminative features plays a critical role for machine learning based approaches. Therefore, we collected the DNS usage of several thousand well-known benign and malicious samples. After analysis period, we extracted ten important features that can be included in machine learning algorithms to train detection models.

- ***NumIPs (F1):*** Conventional benign domains usually have either one or few IP addresses associated with them. On fast-flux networks, a domain name is mapped to multiple IP addresses rather than a single IP address, with the goal of providing high availability and greater performance to the end user.
- ***MinTTL (F2):*** The Time-To-Live (TTL) of each DNS query is stored in the DNS record. Flux hosts typically use a shorter TTL than legitimate hosts, which is usually in the order of a few minutes [1]. The shorter the TTL, the faster a host can change its A records. Therefore, the MinTTL feature is chose to extract in our detection system.
- ***PreEntro (F3):*** The fast-flux agents are often distributed scattered across many different networks [1]. Most legitimate domain names are usually resolved to one or few separate network addresses. Domains related to malicious flux services often resolve to multiple IP addresses located in many different networks [1]. Based on matching IP addresses by prefix /16, we can distinguish them from the same network or different networks. Therefore, we choose this feature to estimate the degree of scattering of IP addresses among different networks, computed as formula 2:

$$PreEntro = \frac{-\Sigma_x p(x) \cdot \log_2 p(x)}{\log_2 |P|} \quad \textbf{(2)}$$

where P is the set of all the /16 IP prefixes in I. (e.g., the /16 prefixes of [*78.46.45.16; 80.93.217.196]* is *[78.46; 80.93]*); and $p(x) = count(x)/|P|$ is the relative frequency of the /16 prefix.

If the value of *PreEntro* is equal zero that means that either all distinct resolved IP addresses in I pointed to domain d belonging to the same network, (the same /16 network prefixes) or only a distinct IP address pointed to a domain d. In contrast, if all distinct resolved IP addresses are completely different (not the same /16 prefixes), the value of *PreEntro* will be equal 1. The *PreEntro* value is closer to 1, that means there are multiple IP addresses belong to different networks or organizations pointed to domain d; conversely, the *PreEntro* value is closer to zero.

- ***NumSub (F4):*** Most normal users when want to add new subcategories into their existing site, they rarely use subdomains. For example, a legitimate company owns example.com domain, if the company wants to add subcategories to their website, they often extend the URL (e.g. example.com/products) instead of using a third-level domain (3LD) (e.g. product. example.com). This allows web developers to avoid complicated DNS updates when adding new content to the site.

In contrast, bot-masters often use 3LD domains instead of subdirectories for their communication.

They may purchase for second-level domain (2LD), e.g., *example.com* from a registrar. In order to avoid increased costs and additional risks, bot-masters tend to create botnets within third-level subdomains (3LD), all under a common 2LD. For example, *botnet1. example.com, botnet2.example.com*, etc., Bot-master sees advantages in using subdomains. If service to a 3LD is suspended, service to other 3LDs within the same 2LD is usually not disrupted (i.e., if *botnet1. example.com* is blocked, traffic to *botnet2.example. com* is not disrupted). This lets bot-masters create multiple redundant DDNS services for their networks, all using the same 2LD.

- ***MEntrSub (F5) and SDEntrSub (F6):*** Given the subdomain X with letters $x_1, x_2, \ldots, x_k$ and respective probability values of $p(x_1), p(x_2), \ldots, p(x_k)$, the Shannon Entropy of X is calculated by using following formula (3):

$$H(X) = -\sum_{i=1}^{n} p(x_i) \cdot \log_2 p(x_i) \quad \textbf{(3)}$$

This feature computes the entropy of the character distribution for each subdomain in *S*. Then, we compute the *Mean (F5 feature)* and *Standard Deviation (F6 feature)* of the set of values $\{H(X)_j\}_{j=1...|S|}$.

- ***StabIPs (F7):*** Fast-flux service networks often belong to private organizations and are scattered across on the world [1]. Therefore, a malicious domain name can be resolved to different IP addresses belonging to different networks. On the other hand, to ensure the purpose of load balancing, legitimate domain names are often mapped to IP addresses belonging to the same network because they are owned in the same company. If the same domain name is resolved to many different networks, the server is likely compromised and used as a flux agent. Therefore, in order to distinguish fast-flux service networks are legitimate or malicious, we define a formula (4) as follows:

$$StabIPs = \frac{\sum_{j=1}^{|D|} unique(prefix(I_j))}{|D|} \quad \textbf{(4)}$$

where I_j is the set of IP addresses associated with domain d_j; $pref\,ix(I_j)$ is an extraction /16 prefix network of I_j; $|D|$ is the total number of relative domains.

- ***EntrSubL (F8)***: Given a domain X and the list of related subdomains S with the length of $l_1, l_2, \ldots, l_k$ and respective probability values are $p(l_1), p(l_2), \ldots, p(l_k)$, the Shannon Entropy of S is calculated by using following formula (5):

$$E(X) = \frac{-\sum_{i=1}^{k} p(l_i) \cdot \log_2 p(l_i)}{\log_2 |S|} \quad \textbf{(5)}$$

where $p(l_i) = count(l_i)/|S|$. This feature checks the information certainty on the length of the subdomains on each domain.

- ***The Expected Score of Domain names (ESOD) (F9):*** This feature aims to measure the expected score for a domain which can distinguish bot-generated domain names and human-generated ones. We first calculate frequency of occurrence of each Ngrams (with N = 3, 4, 5) across the domain name strings of Alexa Top 100,000 sites [26]. We then assign score for each *i*-th Ngrams in the following way:

Table 1. List of selected features

Features	Feature Name	Features Description
F1	*NumIPs*	The number of resolved IP addresses per each of domain
F2	*MinTTL*	The minimum value of Time-To-Live
F3	*PreEntro*	The Entropy of /16 network prefixes
F4	*NumSub*	The Number of Subdomains per each domain
F5	*MEntrSub*	The Mean of Entropy of Subdomains
F6	*SDEntrSub*	The Standard Deviation of Entropy of Subdomains
F7	*StabIPs*	The Stability of IP addresses pool
F8	*EntrSubL*	The Entropy of Subdomain Length
F9	*ESOD*	The Expected Score of Domain name
F10	SeenTime	The Time between the first Seen and the last Seen

$$S_{Ngrams(i)} = \log_{10}(count_{Ngrams(i)}) \quad \textbf{(6)}$$

We extract 23,613 Ngrams and compute score for each of their Ngrams (see Table 2). These Ngrams scores are referred to calculate the expected score for each observed domain.

Given a domain name X, we extract k Ngrams of X as $Ngrams_1, Ngrams_2, \ldots, Ngrams_k$ (with N=3, 4, 5). The expected score of domains (ESOD) of X is computed by the following formula:

Table 2. Excerpt of some Ngrams scores (N=3, 4, 5) in legitimate domain names from Alexa Top 100,000 sites

Ngrams	Counts	S_{Ngrams}
ame	1314	3.118595
res	1305	3.115611
onl	1303	3.114944
for	1293	3.111599
sta	1286	3.109241
nlin	1237	3.092370
new	1222	3.087071
onli	1212	3.083503
nline	1210	3.082785
onlin	1195	3.077368
…	…	…

$$ESOD(X) = \sum_{i=1}^{k} S_{Ngrams(i)} \quad \textbf{(7)}$$

where k is the number of Ngrams of domain X, $S_{grams}(i)$ is the score of *i*-th Ngrams which is referenced from 23,613 Ngrams scores (as shown in Table 1).

For example, for a given two domains *google.com* and *jvxzqy.net,* we first extract the *second-level* domain (2LD) of each domain, and for each 2LD domain we compute its ESOD by applying the formula (6) and the reference N-grams scores as described in formula (5), we have:

$$\begin{aligned} ESON\ (\text{`}google\text{'}) &= S_{goo} + S_{oog} + S_{ogl} + S_{gle} + S_{goog} + S_{oogl} \\ &\quad + S_{ogle} + S_{googl} + S_{oogle} \\ &= 2.418 + 1.939 + 1.986 + 2.298 + \\ &\quad 1.763 + 1.799 + 1.799 + 1.724 + \\ &\quad 1.770 = 17.496 \end{aligned}$$

$$\begin{aligned} ESOD\ (\text{`}jvxzqy\text{'}) &= S_{jvx} + S_{vxz} + S_{xzq} + S_{zqy} + S_{jvxz} + S_{vxzq} \\ &\quad S_{xzqy} + S_{jvxzq} + S_{vxzqy} \\ &= 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 0 \end{aligned}$$

ESOD(*'jvxzqy'*) = 0, this shows that Ngrams (N=3,4,5) of domain *'jvxzqy'* rarely appears or matches Ngrams of legitimate domains. In contrary, ESOD ('google') = 17.496 implies that Ngrams of *'google'* appear frequently in legitimate domains. Therefore, the higher ESOD value is, the higher ability of legitimate domain is.

- ***SeenTime (F10):*** This feature measures the period life between the first seen (*timestamp* t_1) and the last seen (*timestamp* t_2) of a domain d_j.

$$SeenTime(d_j) = (t_2 - t_1)/3600 \quad \textbf{(8)}$$

We use five popular machine learning algorithms along with all the proposed features to train the classifier aim to detect a domain as malicious flux service or legitimate/non-flux service. Afterward, the best classifier will be chosen to classify domains.

4 Experimental and Evaluation

All the experiments related to the collecting DNS traffic data and preprocessing data are conducted on a 4-core 2.13GHz machine with 16GB of RAM for our experiments.

4.1 Data Set

In our experiment, all authoritative DNS queries/ responses in the live DNS traffic were recorded. In addition, in order to increase the number and variety of flux domains in traffic, we also manually retrieved the list of domain names on ATLAS [27], hphosts [14] and DNS-BH [15] websites. During the period of three months monitoring (from Jan 01 to Apr 01, 2016), we have recorded more than 44.1 million DNS queries that aggregated about 5.4 million distinct domain names. We refer to this data as DAT dataset. In order to obtain the labeled data set (LDS), we used semi-manual processes as following describes.

- ***For Legit Domain Names (LDN):*** We derived a large white-list of benign domain names from the Alexa Top global domains list [26] with ranking from1 to 10000. From these benign domains, we check and examine in DAT dataset to get out legitimate DNS traffic that is queried by users during period from Jan 01 to Apr 01, 2016. Overall, we obtained the number of popular legitimate domain names that often appears in DAT dataset. We label this legitimate data set as LDN (see Table 3).

Table 3. Labeled DAT dataset based on known flux-domain, known malware domains, and legitimate domains

Date	# DNS Queries	# Distinct Domains	# LDN	# KMD	# KFFD	# Unknown
Jan-01, 2016	2,668,445	261,484	4,569	2,909	772	253,234
Jan-02, 2016	2,793,057	283,918	4,792	2,874	763	275,489
Jan-03, 2016	2,591,310	260,824	4,732	2,869	736	252,487
Jan-04, 2016	2,433,658	241,388	4,681	2,800	722	233,185
Jan-05, 2016	2,405,839	248,518	4,710	2,741	736	240,331
...	...	...	...	...	...	...
Jan-14, 2016	4,038,918	775,365	4,808	3,629	845	766,083
Jan-15, 2016	3,457,537	455,928	4,723	2,860	758	447,587
...	...	...	...	...	...	...

- ***For Known Fast Flux Domains (KFFD):*** We gathered a dataset of known fast-flux domains from some known botnets such as Zeus, SpyEye, Palevo, and Feodo, which are classified and reported by *abuse.ch* website. The collected dataset contains domain names classified as flux. We will refer to these domains as the KFFD dataset (see Table 3).
- ***For Known Malware Domain names (KMD):*** We first collect the malicious domains from multiple sources such as *Malware Domain List* [28], *Malware Domain* [29], *PhishTank* [30], *hpHosts* [31], and *CyberCrime Tracker* [32]. Overall, we obtained 242,498 of malicious domains. Then, we examine and match these known malware domains in DAT dataset to get out malicious domain names, and label them as **KMD**. The reason we using the KMDs are because the number of KFFD is often very small, and it rarely appears in DAT dataset. In addition, not all KMDs are related to flux networks, but if a malware domain *m* exhibits characteristics of a flux network (see Section 3.3) it will be flagged as a Known Fast Flux Domain (KFFD). In these cases, we identify them manually. Overall, we obtained the number of KFFDs as shown in Table 3. There are also some domains that we cannot find enough reliable information to label them, we marked those domains as **"Unknown"** (see Table 3).

4.2 Experimental Results

We evaluate the results of our experiment based on the labeled data set. We use labeled data set for two purposes: (1) for training classifier models, and (2) for estimating the accuracy of the classifier models.

4.2.1 Comparing the Feature Values

Figure 4 shows the distribution of 95% Confidence Interval (CI) for mean and Standard Errors (SE) between legitimate, malware and fast flux domains on each the extracted feature. The statistical results show that there are discernible biases in the distribution of the selected feature values between benign and malicious domain names. In other words, these selected features exhibit important characteristics that are useful for training classifiers to be able to classify domains as malicious or legitimate.

4.2.2 Training and Evaluation of the Classifiers

We use the labeled data set (LDS) described in Section 4.1 to train classifier models. In this study, we use five machine learning algorithms (Naïve Bayes (NB) [33], Nearest neighbors (KNN) [34], Assistive Vector Machines (SVM) [35], Decision Trees (J48) [36] and Random Forest (RF) [36]) to train and build classification models. The most performance model is selected to be implemented in our detection system.

In addtion, we use two validation methods known as tenfold cross-validation and percentage split to evaluate the accuracy of the classifier algorithms.

- ***Tenfold Cross-Validation (10-fold CV):*** We performed tenfold cross validation on labeled data set by employing various five Machine Learning Algorithms (MLAs) to train and build classification models. We split the data set into 10 smaller random subset, of which nine subsets were used for training, the remaining one was used for evaluation. This process was repeated 10 times to ensure that all samples were tested. The 10-fold CV estimate is the average of these 10 measures [38].
- ***Percentage split:*** To ensure unbiased results in training phase, the dataset is divided into two portions. The first one is 75 percent for training and the rest 25 percent is used to check the correctness, so that the classifier can be evaluated on data that had not been seen previously.

The following measure parameters are used to evaluate predictive performance of classifiers:

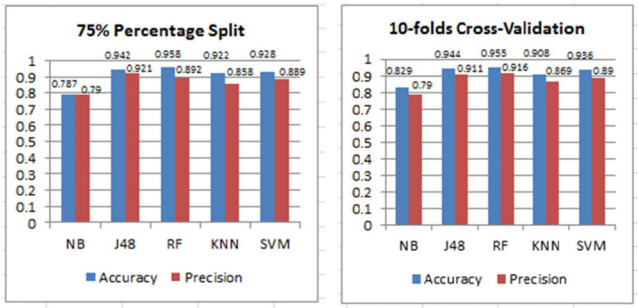
- ***Accuracy:*** is the total number of correctly classified domains divided by the total number of the classified domains (formula 7).

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad \textbf{(7)}$$

- ***Precision:*** is the number of the correctly predicted malicious domains divided by the total number of the domains that are predicted as malicious (formula 8).

$$\Pr ecision = \frac{TP}{TP + FP} \quad \textbf{(8)}$$

The results in Figure 3 show that the detection rate of the Random Forest (RF) classifier is the best with Accuracy around 95.5% and Precision around 91.6%.

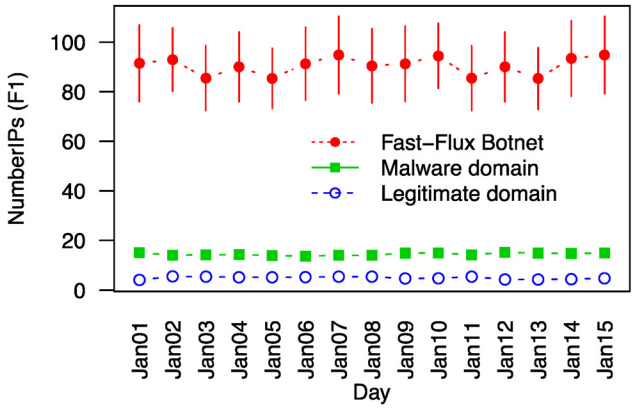


(a) Use Percentage Split (b) Use 10-fold Cross

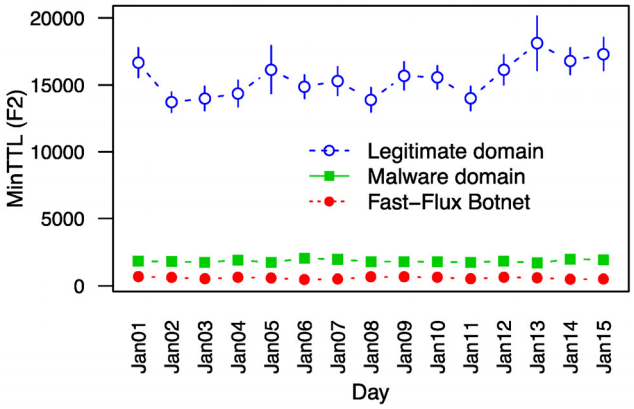
Figure 3. Classification accuracy and precision

Moreover, we also use the Receiver Operating Characteristic (ROC) curve [37] to compare predictions efficiency between classifiers, line in the plot is the closest to the left-hand border and the top border compared to other lines, indicating that it offers the better prediction result among other methods. From the training results of the two experiments are shown in

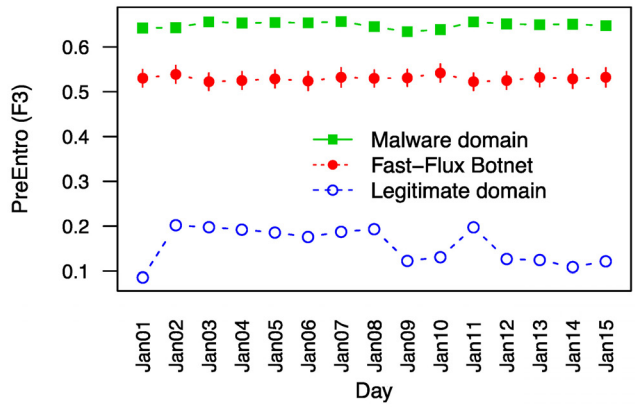
Figure 5, we found that the 10-fold CV usually produces better results. Herein, the Random Forest (RF) classifier achieves the best prediction result with an area under the ROC curve (AUC) of 0.99 for both the 10-fold CV and percentage split (see Figure 5). These results confirm that our proposed approach can detect malicious flux domain with high accuracy. Therefore, the RF classification algorithm is the best choice to train and build a model for detecting unknown malicious domains in our detection system.



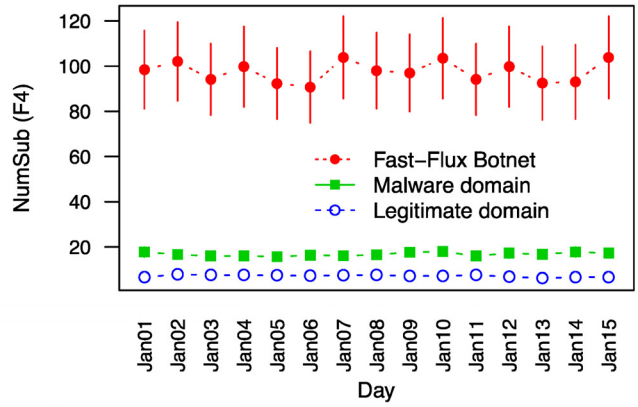
(a) The distribution of 95% CI of Mean and SE for NumIPs feature



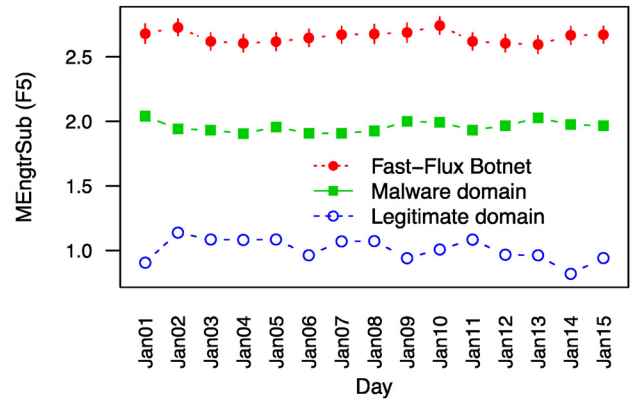
(b) The distribution of 95% CI of Mean and SE for MinTTL feature



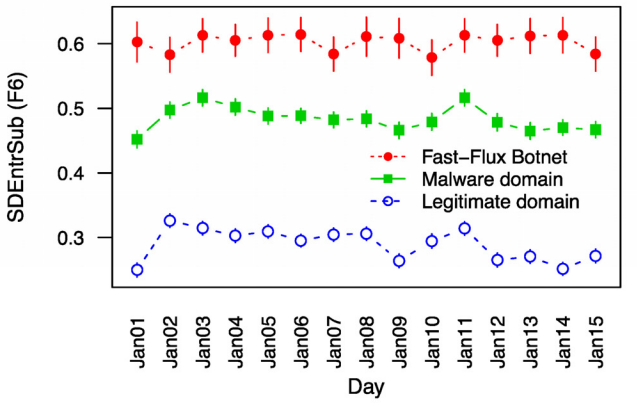
(c) The distribution of 95% CI of Mean and SE for PreEntro feature



(d) The distribution of 95% CI of Mean and SE for NumSub feature

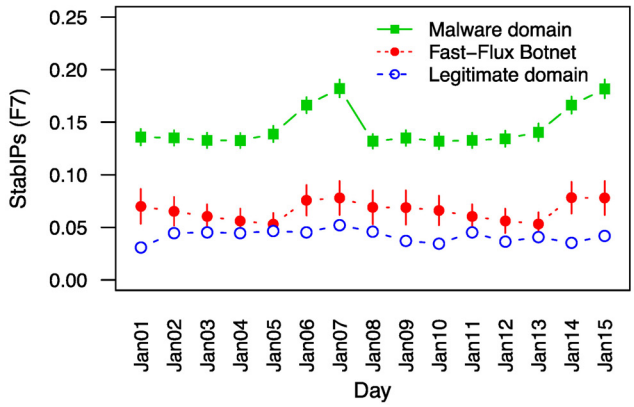


(e) The distribution of 95% CI of Mean and SE for MEntrSub feature

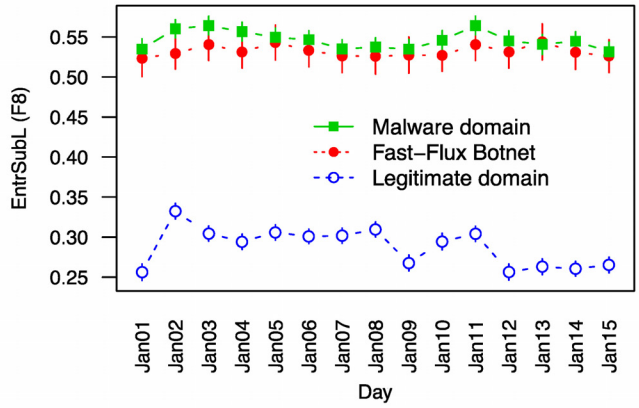


(f) The distribution of 95% CI of Mean and SE for SDEntrSub feature

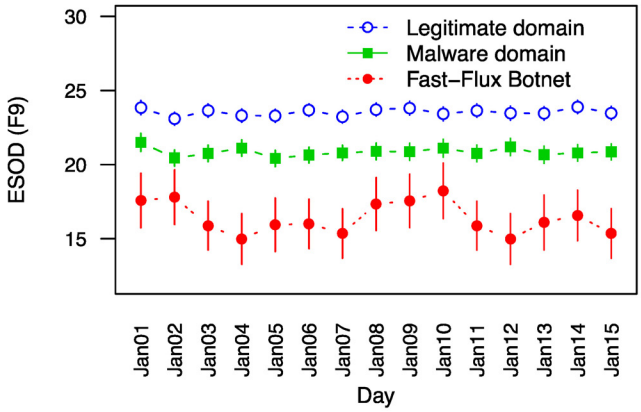
Figure 4. Comparing the distribution of 95% Confidence Interval (CI) for Mean and Standard Error (SE) between benign, malware and fast-flux domains on each the extracted-feature



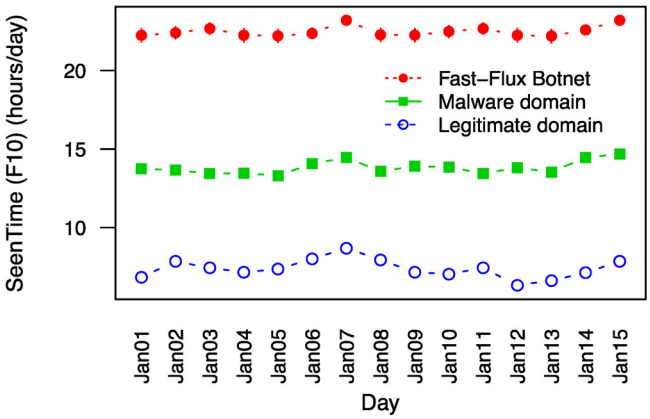
(g) The distribution of 95% CI of Mean and SE for StabIPs feature



(h) The distribution of 95% CI of Mean and SE for EntrSubL feature

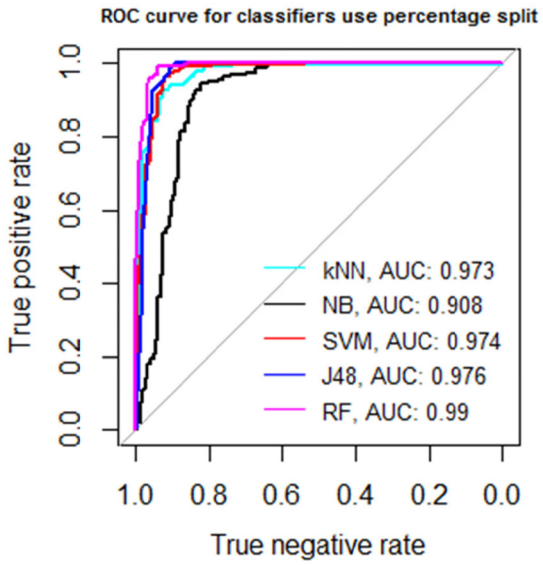


(i) The distribution of 95% CI of Mean and SE for ESOD feature

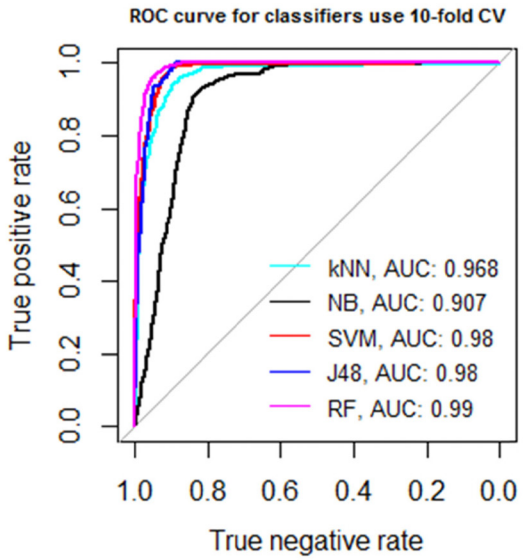


(j) The distribution of 95% CI of Mean and SE for SeenTime feature

Figure 4. Comparing the distribution of 95% Confidence Interval (CI) for Mean and Standard Error (SE) between benign, malware and fast-flux domains on each the extracted-feature (continue)



(a) use percentage split



(b) use 10-fold CV

Figure 5. ROC Curve compares the prediction performance of classifiers by signifying the tradeoff between True Negative (TN) rate and True Positive (TP) rate

4.2.3 Evaluation on the Recorded Dataset

- ***Evaluation of True Positives (TP):*** The experimental results indicate that the accuracy rate of the RF classifier is approximately 95.5%. In order to detect malicious domains that have not been seen previously in our training set, we collected the malware domains from various sources such as *Malware Domain List* [28], *Malware Domain* [29], which we have never used in our training.

During the period performed the experiments, the *Malware Domain List* [28], *Malware Domain* [29] reported 242,498 domains as being malicious. Out of these 242,498 domains, we examined and obtained 7,006 malware domains appear in the recorded dataset from Jan 09 to Jan 15, 2016. The remaining 235,948 domains were not requested in our recorded dataset. Therefore, we used 7,006 malware domains to evaluate our detection rate (True positive rate).

We apply filtering stage proposed by Perdisci et al. [25] to filter out candidate fast-flux domains. If a malware domain *m* exhibit characteristic of a flux network (e.g., NUM-IPS ≥30 and MINTTL ≤ 10800 seconds and *PreEntro* ≥ 0.333), domain *m* will be flagged as a fast-flux domain. Overall, we manually confirmed and marked 456 of 7,006 malicious domains as fast-flux. Finally, we fed 7,006 malicious domains (includes 6,550 malware domains and 456 fast-flux domains that were previously unknown in the training set) to our system, the experimental results showed that 6,289 of 6,550 malware domains were correctly classified as malware (TP = 96%) by our detection system, and 431 of 456 fast-flux domains were correctly classified as fast-flux (TP = 94.52%) (see Table 4).

Table 4. The detection results on dataset that was not seen previously

		Prediction			
		Legitimate	Malicious		
			Malware	Fast-Flux	
Actual class	Legitimate	10,583	471	14	**11,068**
	Malware	246	6,289	15	**6,550**
	Fast-Flux	10	15	431	**456**
		10,839	**6,775**	**460**	

- ***Evaluation of False Positive (FP):*** To measure the false positive rate, we collected the benign domains from Alexa Top global domains list [26] with ranking from 10000 to 30000 (which were not seen in training dataset). We examined and obtained 11,068 of popular legit domains appear in the recorded DNS data from Jan 09 to Jan 15, 2016. To evaluate false positive rate, we checked how many domains in legitimate dataset (which were previously unknown in training set) were misclassified as malicious by our system. Through experiments, we found that only 14 of 11,068 legitimate domains were misclassified as Fast-Flux domains (FP = 0.13%), and 471 of 11,068 legitimate domains were misclassified as malware domains (FP = 4.25 %) (see Table 4).
- ***Evaluation of True Negative (TN)****:* To measure the true negative rate, we checked the 11,068 legitimate domains, and found that 10,583 domains were correctly classified as legitimate domains (TN = 95.62%) (see Table 4).
- ***Evaluation of False Negative (FN):*** To measure the false negative rate, we checked the number of the domains belonging to the 6,550 malware domains were classified as legitimate, and how many of the domains belonging to the 456 fast-flux domains were classified as legitimate. We found that 246 of the 6,550 malware domains were misclassified as legitimate (FN = 3.76%), and 10 of 456 fast-flux domains were misclassified as legitimate (FN = 2.19%) (see Table 4).

To compare to previous works such as Zhou et al. [7], we performed this experiment on the same our dataset, and used the RF classifier (which is considered the best classifier in five classifiers) to compare the effectiveness of detection.

The results of the two comparative experiments are presented in Table 5. Our approach achieved a higher prediction accuracy (98.79 %) compared to an accuracy of 93.12 % achieved by Zhou et al. [7]. Our proposed method received the FN rate of 2.19%, lower than Zhou's approach with the FN rate of 8.9%. This is because some of our proposed features have the 95% CI clearer than the 95% CI of some of the features proposed by Zhou. Figure 6 is an example to show that the "sub_growth_8" feature proposed by Zhou et al. [7] is difficult to clearly distinguish between malware doamain, Fast-flux domain and Legitimate domain.

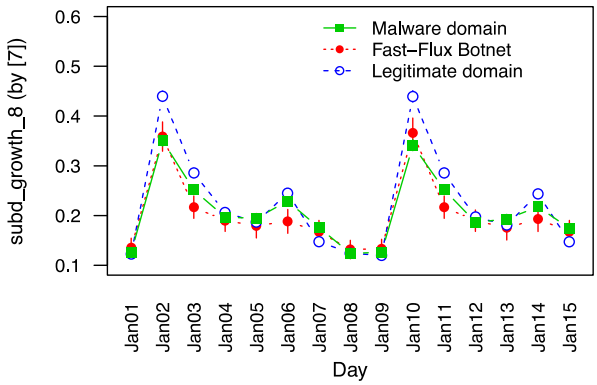


Figure 6. The distribution of 95% Confidence Interval (CI) for Mean and Standard Error (SE) between benign, malware and fast-flux domains of a feature proposed by [7]

Table 5. The performance comparison between our approach and [7]

Method	TP%	FP%	TN%	FN%	Acc%	Prec%
Zhou [7]	91.3	4.8	94.2	8.9	93.12	95.01
Our approach	94.52	0.13	95.62	2.19	98.79	99.86

5 Conclusion

In this work, we have presented a system for detecting Fast-Flux domains. We conducted a controlled experiment with a real-world dataset consisting of billions of DNS requests, and use 10 features that we extracted to characterize different properties of domain names. Among the ten introduced features, there are seven features are first proposed in this study. In addition, this work also shows the effectiveness of features by comparing the distribution of 95% confidence interval for mean and standard error between benign, malware and fast-flux domain names on each feature. We have experimented on various MLAs and show that Random Forest is the best classifier to detect a domain is malicious or legitimate. We also evaluate the effectiveness on the recorded dataset, the detection rate we observed is similar to the detection rate estimated by the percentage split and cross-validate evaluations on the training set. Obviously, our approach is able to detect a high number of unknown malicious domains includes fast-flux domains from DNS traffic with a significant detection effect.

Acknowledgments

We are grateful to anonymous reviewers for their careful reading and thoughtful comments towards the improvement of our manuscript.

References

[1] W. Salusky, R. Danford, Know Your Enemy: Fast-flux Service Networks, https://www.honeynet.org/book/export-/html/130, 2007.

[2] P. Lombardo, S. Saeli, F. Bisio, D. Bernardi, D. Massa, Fast Flux Service Network Detection via Data Mining on Passive DNS Traffic, *21st International Conference on Information Security, ISC 2018*, Guildford, UK, 2018, pp. 463-480.

[3] D. Cafuta, V. Sruk, I. Dodig, Fast-Flux Botnet Detection Based on Traffic Response and Search Engines Credit Worthiness, *Tehnički vjesnik*, Vol. 25, No. 2, pp. 390-400, April, 2018.

[4] T. Holz, C. Gorecki, K. Rieck, F. C. Freiling, Measuring and Detecting Fast-Flux Service Networks, *13th Annual Network and Distributed System Security Symposium (NDSS'08)*, San Diego, California, USA, 2008, pp. 1-12.

[5] F. T. Zou, S. Y. Zhang, W. X. Rao, Hybrid Detection and Tracking of Fast-Flux Botnet on Domain Name System Traffic, *China Communications*, Vol. 10, No. 11, pp. 81-94, November, 2013.

[6] X. Chen, G. Li, Y. Zhang, X. Wu, C. Tian, A Deep Learning Based Fast-Flux and CDN Domain Names Recognition Method, *2019 2nd International Conference on Information Science and Systems*, Tokyo, Japan, 2019, pp. 54-59.

[7] C. Zhou, K. Chen, X. Gong, P. Chen, H. Ma, Detection of Fast-Flux Domains Based on Passive DNS Analysis, *Acta Scientiarum Naturalium Universitatis Pekinensis*, Vol. 52, No. 3, pp. 396-402, May, 2016.

[8] A. Almomani, Fast-flux Hunter: A System for Filtering Online Fast-flux Botnet, *Neural Computing and Applications*, Vol. 29, No. 7, pp. 483-493, April, 2018.

[9] R.-D. Huang, S.-Y. Kuo, Y.-H. Chou, Detecting Strategy of Fast Flux Domain Based on Hidden Markov Model, *Journal of Internet Technology*, Vol. 16, No. 2, pp. 277-287, March, 2015.

[10] F.-H. Hsu, C.-S. Wang, C.-H. Hsu, C.-K. Tso, L.-H. Chen, S.-H. Lin, Detect Fast-Flux Domains Through Response Time Differences, *Ieee Journal on Selected Areas in Communications*, Vol. 32, No. 10, pp. 1947-1956, October, 2014.

[11] T. Gržinić, D. Perhoč, M. Marić, F. Vlašić, T. Kulcsar, CROFlux-Passive DNS Method for Detecting Fast-flux Domains, *37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2014, pp. 1376-1380.

[12] C.-M. Chen, M.-Z. Huang, Y.-H. Ou, Detecting Hybrid Botnets with Web Command and Control Servers or Fast Flux Domain, *Journal of Information Hiding and Multimedia Signal Processing*, Vol. 5, No. 2, pp. 263-274, April, 2014.

[13] B. N. Al-Duwairi, A. T. Al-Hammouri, Fast Flux Watch: A Mechanism for Online Detection of Fast Flux Networks, *Journal of Advanced Research*, Vol. 5, No. 4, pp. 473-479, July, 2014.

[14] S. Martinez-Bea, S. Castillo-Perez, J. Garcia-Alfaro, Real-time Malicious Fast-flux Detection Using DNS and Bot Related Features, *2013 Eleventh Annual International Conference on Privacy, Security and Trust*, Tarragona, Spain, 2013, pp. 369-372.

[15] H.-T. Lin, Y.-Y. Lin, J.-W. Chiang, Genetic-based Real-time Fast-Flux Service Networks Detection, *Computer Networks*, Vol. 57, No. 2, pp. 501-513, February, 2013.

[16] C.-M. Chen, M.-Z. Huang, Y.-H. Ou, Detecting Web-based Botnets with Fast-flux Domains, in: J. S. Pan, C. N. Yang, C. C. Lin (Eds.), *Advances in Intelligent Systems and Applications-Volume 2*, Springer, 2013, pp. 79-89.

[17] Z. B. Celik, S. Oktug, Detection of Fast-Flux Networks Using Various DNS Feature sets, *2013 IEEE Symposium on Computers and Communications (ISCC)*, Split, Croatia, 2013, pp. 000868-000873.

[18] H.-T. Wang, C.-H. Mao, K.-P. Wu, H.-M. Lee, Real-time Fast-flux Identification via Localized Spatial Geolocation Detection, *2012 IEEE 36th Annual Computer Software and Applications Conference*, Izmir, Turkey, 2012, pp. 244-252.

[19] M. T. Qassrawi, H. L. Zhang, Detecting Malicious Fast Flux

Domains, *Applied Mechanics and Materials*, Vol. 157-158, pp. 1264-1273, February, 2012.

[20] R. Perdisci, I. Corona, G. Giacinto, Early Detection of Malicious Flux Networks via Large-Scale Passive DNS Traffic Analysis, *IEEE Transactions on Dependable and Secure Computing*, Vol. 9, No. 5, pp. 714-726, September-October, 2012.

[21] C.-H. Hsu, C.-Y. Huang, K.-T. Chen, Fast-flux Bot Detection in Real Time, in: S. Jha, R. Sommer, C. Kreibich (Eds.), *Recent Advances in Intrusion Detection*, Lecture Notes in Computer Science, Springer, 2010, pp. 464-483.

[22] E. Passerini, R. Paleari, L. Martignoni, D. Bruschi, Fluxor: Detecting and Monitoring Fast-flux Service Networks, in: D. Zamboni (Ed.), *Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, 2008, pp. 186-206.

[23] E. B. Fjellskål, *PassiveDNS: A Tool to Collect DNS Records Passively*, https://github.com-/gamelinux/passivedns.

[24] *DNSDB: Security Information Exchange*, https://www.dnsdb.info/.

[25] R. Perdisci, I. Corona, D. Dagon, W. Lee, Detecting Malicious Flux Service Networks Through Passive Analysis of Recursive DNS Traces, *2009 Annual Computer Security Applications Conference (ACSAC'09)*, Honolulu, HI, USA, 2009, pp. 311-320.

[26] *Alexa Top Global Sites*, http://www.-alexa.com/topsites.

[27] Arbor Networks, *ATLAS Summary Report: Global Fast Flux*, http://atlas.arbor.net/summary-/fastflux.

[28] Malware Domain List, http://www.-malwaredomainlist.com/hostslist/hosts.txt.

[29] Malware Domains, http://mirror1.-malwaredomains.com/files/domains.txt.

[30] PhishTank, http://data.phishtank.com/data/-online-valid.csv.

[31] hpHosts, http://hphosts.gt500.org/hosts.txt.

[32] *CyberCrime Tracker*, CYBERCRIME-07-26.20, http://cybercrime-tracker.net/all.php.

[33] I. H. Witten, E. Frank, M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2011.

[34] S. Manocha, M. Girolami, An Empirical Analysis of the Probabilistic K-nearest Neighbour Classifier, *Pattern Recognition Letters*, Vol. 28, No. 13, pp. 1818-1824, October, 2007.

[35] L. Wang, *Support Vector Machines: Theory and Applications*, Springer, 2005.

[36] X.-B. Li, A Scalable Decision Tree System and Its Application in Pattern Recognition and Intrusion Detection, *Decision Support Systems*, Vol. 41, No. 1, pp. 112-130, November, 2005.

[37] A. P. Bradley, The Use of the Area under the ROC Curve in the Evaluation of Machine Learning Algorithms, *Pattern Recognition*, Vol. 30, No. 7, pp. 1145-1159, July, 1997.

[38] D.-T. Truong, G. Cheng, A. Jakalan, X.-J. Guo, A.-P. Zhou, Detecting DGA-based Botnet with DNS Traffic Analysis in Monitored Network, *Journal of Internet Technology*, Vol. 17, No. 2, pp. 217-230, March, 2016.

Biographies

Dinh-Tu Truong was born in 1979. He works at Ton Duc Thang University, Ho Chi Minh City, Vietnam. His research interests include network security, network intrusion detection, network measurement and traffic sampling, IoT Security, Machine Learning and Deep Learning.

Dac-Tot Tran was born in 1984. He works at Ho Chi Minh City University of Food Industry, Ho Chi Minh City, Vietnam. His research interests include network security, network intrusion detection, network measurement and traffic sampling, computer vision.

Bao Huynh is Vice Dean of IT faculty, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Vietnam. He completed the Ph. D program in 2017 with major is computer science at VSB - Technical University of Ostrava, the Czech Republic. His research interests include Data mining, Big data, parallel computing, network infrastructure, and network security.