Detection of Fast-Flux Networks Using Various DNS Feature Sets

Z. Berkay Celik and Sema Oktug

Department of Computer Engineering
Istanbul Technical University
Maslak, Istanbul, Turkey 34469
{zbcelik,oktug}@itu.edu.tr

***Abstract*—In this work, we study the detection of Fast-Flux Service Networks (FFSNs) using DNS (Domain Name System) response packets. We have observed that current approaches do not employ a large combination of DNS features to feed into the proposed detection systems. The lack of features may lead to high false positive or false negative rates triggered by benign activities including Content Distribution Networks (CDNs). In this paper, we study recently proposed detection frameworks to construct a high-dimensional feature vector containing timing, network, spatial, domain name, and DNS response information. In the detection system, we strive to use features that are delay-free, and lightweight in terms of storage and computational cost. Feature sub-spaces are evaluated using a C4.5 decision tree classifier by excluding redundant features using the information gain of each feature with respect to each class. Our experiments reveal the performance of each feature subset type in terms of the classification accuracy. Moreover, we present the best feature subset for the discrimination of FFSNs recorded with the datasets we used.**

***Index Terms*—network security, Fast-flux Service Networks (FFSNs), feature selection, classification**

I. Introduction

Fast-flux service networks continuously update their DNS entries at regular intervals, *i.e.*, fast IP changing hosts [1]. The repeated and rapid IP change in DNS A and/or DNS NS resource records conceals the actual location of malicious servers, and helps them evade blacklists and take downs. This process also adds an extra layer to the FFSNs' communication structure to increase its resilience and anonymity in a wide variety of malicious activities including phishing (*e.g.*, via spam e-mail) and malware hosting. On the other hand, CDNs which show similar characteristics to FFSNs (*i.e.*, referred as illegitimate content distribution networks) also associated with multiple records for load balancing and regional server assignments to increase responsibility and availability [2]. Hence, the problem of reliable discrimination of them has posed additional technical difficulties, and requires comprehensive, in-depth analysis of DNS feature space.

The principal network-based methods to detect FFSNs either rely on passive DNS analysis [3] used to discover a domain name participated for malicious operations, or consecutive DNS query packets [4], [5], or DNS server response duration [6]. Algorithmically generated domain names have also been exploited to detect domain fluxing by botnets (*e.g.*, [7]). Such textual analysis may give the first alarm as a sign of domain fluxing. These detection methods are imperfect, *e.g.*, they may suffer from high detection latency (*i.e.*, extracting features as a timescale greater than Time To Live (TTL), or multiple DNS lookups) or false positives detected by utilizing imperfect and insufficient features.

The work presented here is related to the detection of FFSNs, in particular, by jointly building timing, domain name, spatial, network and DNS answer features which are extracted from the first DNS response packet. We consider many rather than a few features and also their combinations which are constructed by surveying the recent literature. The main aim of this survey is to study and analyze the benefits of the features and also, in some cases, the necessity of applying joint feature subsets. We provide an overview of the various feature subsets to be used for classification. We illustrate them by highlighting the efforts done by the authors in developing novel procedures, and analyze them in terms of their discrimination power.

In this paper, the detection of FFSNs is purely based on the DNS request and the corresponding response packets collected from a recursive DNS server. Unlike recent FFSN detection proposals, our objective is to build a feature pool which may increase the detection of FFSNs. Some of the features require additional operations such as WHOIS messages, IP Coordinate Database, and a list of ground-truth labeled benign domain names. These operations add an additional delay to the detection, however, do not consider consecutive DNS lookups that take TTL value of each domain into account or take several minutes/hours to collect. Our objective is to study the detection of FFSN activity using various feature sets by classification. Specifically, we construct a 19-dimensional feature vector, and evaluate the performance of each subset of features based on its accuracy, and assess the feature space in order to find an optimal feature subset of the constructed feature vector. Our experiments help us to characterize each feature for the discrimination of FFSNs from benign networks by assessing each feature subset.

The remainder of this paper is organized as follows: In Section II, we describe our feature space and the DNS features employed. In Section III, the feature generation procedure, detection system and the way the datasets were acquired and processed are described. In Section IV, we present detailed analysis of classification results for each subset of the feature space. Finally, we summarize the results and give future research directions in Section V.

978-1-4799-3755-4/13/$31.00 ©2013 IEEE

TABLE I: FEATURE SET OF FFSN DETECTION SYSTEM

Set Type	Subset Name	Features
DNS Answer-based	ϕ_1	Number of unique A records Number of NS records DNS packet size TC (Truncated) Flag is set
Domain name-based	ϕ_2	Edit Distance KL (Kullback-Leibler) Divergence (unigrams and bigrams) Jaccard Index (unigrams and bigrams)
Spatial-based	ϕ_3	Time Zone Entropy of A records Time Zone Entropy of NS records Minimal service distances (mean and standard deviation)
Network-based	ϕ_4	Number of distinct autonomous systems Number of distinct networks
Timing-based	ϕ_5	Round Trip Time of DNS request Network delay (mean and standard deviation) Processing delay (mean and standard deviation) Document fetch delay (mean and standard deviation)

II. Feature Selection

In our framework, we split the feature set into five categories according to the data collection method: DNS answer, domain name, spatial, network, and timing features as presented in Table I. The subsections in this section summarize the subsets of features, and as well as illustrate the complexity and additional operations related to the features.

A. DNS Answer-based Features

DNS answer-based features are computed, without any additional cost, by directly inspecting the fields of the DNS response packets. The cardinality of these features is expected to be large for FFSNs and small for benign domains. These features have been widely applied [4], [5], [8]. IP and NS record diversity may still have discrimination power in single lookup; as the count of these features increases, a higher probability of the FFSN detection is observed. In our experiments, we also check if DNS-packet flags such as TC (Truncated) flag are set or not. Whenever the TC flag set to 1 in a response packet, it implies that the response cannot fit in 512-byte limit of a UDP packet; so the client will need to launch an additional DNS request with a TCP query [9]. However, we have seen a couple of TC flags set DNS responses in our experiments while querying FFSN domains. Hence, we do not include it as a feature; instead, it may be used as a filtering feature for further analysis. On the other hand, DNS packet size which is an important metric includes both DNS sections of question, answer and additional records could be a good discriminator as a whole or separately for each record. Since a number of A and NS records and domain name features are closely related with the DNS packet size, and not strongly dependent on mimicry attacks that allows botmasters to avoid detection [10], we exclude DNS packet size from our feature space. However, more sophisticated classifiers may be used to exploit dependencies between size and sequence information of packets as proposed in [11].

B. Domain name-based Features

In particular, domain name-based features are designed to detect algorithmically generated domain names (*e.g.*, by the domain generation algorithm (DGA) [12]). However, our assumption is that FFSNs use a long sequence of candidate domain names and will tend to use distributions for letters/syllables/n-grams that do not closely match the true distribution of valid domain names. For that reason, domain-level textual features may improve the detection. Hence, we calculate the similarity and divergence metrics between a given set of domain names. First we build a list of benign domain names $(x_1, x_2, x_3, \ldots, x_n)$ over a fixed time window, and then we evaluate the average distance/similarity metric of given (malicious or benign) domain name under scrutiny. Our calculation requires a set of benign domain names in order to generate the metrics (see Section III-A for more information), *i.e.*, a whitelist of domains forming "non-malicious dataset". For each domain name in the datasets, we consider metrics using on second level domain (SLD) field, *i.e.*, *a.b.example.com* is reduced to *example*. The metrics we use in our calculations are similar to those of [7], where the authors use them for a first alarm to indicate domain fluxing in a network targeting recently developed botnets such as Torpig, Karaken and Conficker. The authors showed that these metrics lead to a good detection of algorithmically generated domain names.

Let d_1 and d_2 be two probability distributions of a discrete random variable where d_1 is for set of whitelist domain names and d_2 is the given domain name either FFSN or benign domain name. First we employ Kullback-Leibler (KL) divergence of unigram and bigram distributions,

$$D_{KL}(d_1||d_2) = \sum_i d_1(i) log \frac{d_1(i)}{d_2(i)}.$$

We use the back-off smoothing method [13] in order to allow operation on a full set of random variables occurring in domain space.

Secondly, we can calculate the Jaccard Similarity, $SIM(X,Y) = |X \cap Y|/|X \cup Y|$, between sample datasets X and Y. It is defined as the size of the intersection of the unigrams and bigrams of domain names divided by the size of their union. The result ranges from no unigrams or bigrams in common to one, which means that the given domain names are identical. Obviously, we expect that given benign domain

URLs have a higher similarity than FFSN domain names. Finally, Levenshtein Edit Distance of two domain names S_1 and S_2 is calculated by finding the minimum number of edit operations required to transform S_1 into S_2. The edit operations allowed are inserting a character into a string, deleting a character from a string, or replacing a character of a string by another character.

C. Spatial-based Features

Recently, [8] proposed two spatial metrics to provide a delay-free FFSN detection mechanism. Given the list of IP addresses of A and NS records, authors map the IP addresses into spatial distribution of these records to assess the uniform distribution degree of malicious hosts. Given an address set Q, each IP address is mapped to a coordinate $C(Q) =< C1(Q), C2(Q) >$ where $C1(Q)$ and $C2(Q)$ map to the latitude and longitude, respectively. Then, each IP address in $C(Q)$ is transformed to GMT Time Zones, and finally Time Zime Entropy (TZE) is defined by

$$TZE(C(Q)) = -\sum_{t \in GMT} (N_t(C(Q))/|Q|)(\log(N_t(C(Q)/|Q|))),$$

where N_t is the number of times $C(Q)$ located in the tth time zone for the given hosts.

TZE may be ineffective for CDNs since CDNs may have FFSN-like spatial distribution. For this reason, the authors defined the average and the standard deviation of minimal service relationship estimator as a second feature. Euclidean distance, $q_{mm'}$, is calculated from each IP address in the answer section, q_m, to the each NS IP address in the additional section, $q_{m'}$, as a "service" distance. Then the average and the standard deviation of the minimum service distances are designated features which are expected to help discriminating CDNs from FFSNs depending on the assumption that CDNs may have closer spatial service relationship than FFSNs.

D. Network-based Features

Similar to the spatial-based features, network-based features identify the number of associated networks and autonomous system numbers (ASNs) of the IP addresses in A records. [5] showed that benign hosts are mostly located in a circumscribed geographical area and owned by the same company and are all members of the same autonomous system. Compared to the spatial-based features which require an additional look-up of an up-to-date GEOIP database, network-based features require WHOIS command to extract the related feature values. Furthermore, [14] showed that ASNs together with spatial characteristics respond well to identify potential FFSNs.

E. Timing-based Features

The strength of the timing-based features relies on the assumption that FFSNs may have a single associated IP address. Although, FFSNs often consist of only a few bots, timing based features may well discriminate the FFSNs. However, requirement of HTTP packets incurs additional overhead, especially when the traffic load is high. In addition, while constructing dataset, only active FFSN URLs are processed in order to make the timing features consistent with the other feature sets. In [6], the authors proposed three timing-based metrics as follows: 1) network delay defined as time between HTTP GET request and TCP SYN+ACK packet (*i.e.*, capturing network congestion), 2) processing delay defined as the time to process a dummy HTTP request (*i.e.*, workload of a server), and 3) document fetch delay defined as time required to fetch a webpage. Our system monitors the packet exchanges between the client and the server, and extracts these features on the fly. In addition to these features, we also identified the Round Trip Time (RTT) as a promising, and not strongly dependent on multiple IP addresses. Since processing delay is also based on the subtraction of network RTT and application level RTT, we exclude RTT from our feature set. In our calculations, in order to get consistency in delay metrics, all computations are the averages of the total of three connection attempts, and then the overall average and the standard deviation of the delay metrics are calculated.

III. System Overview

A. DNS Data Collection

We have collected URLs from ATLAS Fast-Flux database[1] and FastFlux Tracker[2] during the period of 4 months (from October 2012 to January 2013). Overall, we were able to collect 476 domains as fast-flux and 1,853 as benign classes. During the web page retrieval and DNS queries, we dissect each DNS packet response with modified version of the tshark [15] libraries before feeding to our detection system.

In order to compare the textual differences of domain-based features between FFSNs recorded for the dataset we used and the recent traditional Botnets such as Conficker, Torpig and Kraken, we collected botnet domain names from Pc Tools[3], and Damballa[4]. Note that except for Kraken botnet, other botnets utilize domain generation algorithm (DGA) to construct a list of the domain names as a rallying host computed from the predefined algorithms embedded to binary code of the bots independently until rallying host provides a response (*e.g.*, Torpig uses DGA by seeding with the current date and a numerical parameter [12]). Kraken botnets use more complicated methods by matching the frequency of occurrences of vowels, consonants and concatenating the domain names with suffixes [7]. We also construct a whitelist of domain names to measure distance and similarity metrics between FFSNs and non-malicious domain names. Hence we collect as a total of over 5000 domain names from Alexa Top Global Sites[5] and Google most visited sites[6]. Some of the features in our feature space require additional database and operations each listed along with the complexity in Table II.

[1] http://atlas.arbor.net/
[2] http://dnsbl.abuse.ch/fastfluxtracker.php
[3] https://www.pctools.com
[4] https://www.damballa.com
[5] http://www.alexa.com
[6] http://www.google.com/adplanner/static/top1000/

TABLE II: COMPLEXITY AND ADDITIONAL OPERATIONS OF FEATURE SUBSETS

Operations	Complexity	Additional Requirements
DNS Answer-based		
Packet Analysis	O(N)	–
Domain-based		
KL Divergence	O(ND)	Whitelist of benign domain names
Jaccard Index	O(ND^2W)	
Edit Distance	O(N^2D^2)	
Spatial-based		
Database Lookup	O(NM)	IP Coordinate Database
Network-based		
WHOIS Processing	O(N)	WHOIS command
Timing-based		
Delay Calculation	O(N)	HTTP Requests

Notation	
N	Number of test domain names
W	Number of domain names in whitelist
D	Max domain name size
M	IP coordinate size

B. The Classifier

In this work the C4.5 algorithm [16] is used for classification. The C4.5 algorithm creates a decision tree, where at each node of the tree the feature(s) with normalized largest information gain is used to split the data into sub-groups, ending at the leaf nodes. A decision tree should have the property that at each leaf node, a strong majority of the samples belong to one class, which is also chosen as the predicted class for samples belonging to that leaf node.

The C4.5 algorithm uses two types of tests for each feature X. The equality test X is applied for discrete attributes, and $X \leq \theta$ is applied for numeric attributes where θ is a constant threshold. The candidate threshold values are specified by sorting the distinct values of X that appear in training set by obtaining a threshold between the adjacent values.

At each step of the algorithm, one feature is selected from the set of current leaf nodes with the attribute split that will yield the greatest *normalized information gain*. With a given discrete class random variable C and binary split of feature X, normalized information gain of a leaf node is obtained as:

$$NIG(C|X) = \frac{H(C) - H(C|X)}{H(C)}, \tag{1}$$

where H is Shannon's entropy, and $H(C)$ is defined as:

$$H(C) = -\sum_{c_i} p(C = c_i) \log_2(p(C = c_i)) \tag{2}$$

and $H(C|X)$ is defined as:

$$H(C|X) = -\sum_{j} p(X = x_j) \sum_{i} \beta_i, \tag{3}$$

where β_i is defined as $p(C = c_i|X = x_j) \log_2(p(C = c_i|X = x_j))$

In order to classify a given data sample, the leaf node to which the data sample belongs is found. This is performed by following the branches that the data sample satisfies, starting from the root and ending at a leaf node. The class of that node associated at a leaf node with a sufficient high class purity becomes the predicted class. Moreover, the C4.5 algorithm has many options such as error-reduced pruning, avoiding overfitting, and handling missing values. In our experiments, we use the subtree raising algorithm to overcome the overfitting problem. In this algorithm, a subtree from downward may be moved upwards towards the root of the tree to join with the parent node. Given a particular confidence, we find confidence limits, and we use that upper confidence limit as an estimate for the error rate of the node. An error estimate for a subtree is calculated as a weighted sum of error estimates for all its leaves and itself. If the node's estimated error is less than the combined error estimate of its leaves, they are pruned away.

In our experiments, the classification process of the C4.5 algorithm is treated as a binary problem, where classes are labelled as benign and fast-flux. The confidence level is set to 0.25 and the minimum number of instances per leaf is set to 2 for pruning. As the decision tree classifier builds a tree during the training phase, the features that best separate the benign and fast-flux classes can be clearly observed. The attributes resulting in the highest information gain are considered to have more discrimination power, thus the results obtained can be used as a filter to rank features according to their calculated information gain values. Finally, with a given accuracy threshold, a set of features can be decided with the best discrimination performance of our dataset.

IV. Experimental Results

In order to find the best subset of our features that result in minimum error rate, we use the 10-fold cross validation approach by dividing the dataset into 10 folds of approximately equal size which have proportionally the same number of classes in all 10 folds. In Figure 1, the average values of accuracy, true positive rates (TPR) and false positive rates (FPR) on all 10 folds are presented for each subset and combination of all feature subsets. We observed that spatial (ϕ_3) feature set performs best, followed by network (ϕ_4), DNS answer (ϕ_1), domain name (ϕ_2) and timing (ϕ_5) feature sets. The ranked results for each feature subset by taking averages of information gain of each feature belong to that subset is presented in Figure 2. We observe that there is a linear relationship between spatial, DNS answer and network features. The cardinality of IP addresses resolved for a given domain name connected to the distribution of both spatial and network based features, *i.e.*, in general all feature pairs are statistically dependent and results in approximately same accuracy results as 98.8%, 97.7%, and 97.1%, respectively. The increase in accuracy is observed by correctly classifying CDNs that will be detailed in the sequel. When we jointly use all feature sets, the accuracy becomes 98.9%, and the predictions become insensitive to the timing and domain feature sets. However, this may not be the case for other classifiers. Accuracy may change *e.g.*, by assuming that the features are conditionally independent given the class of origin and learning a model for the class conditional probability distribution of each feature or

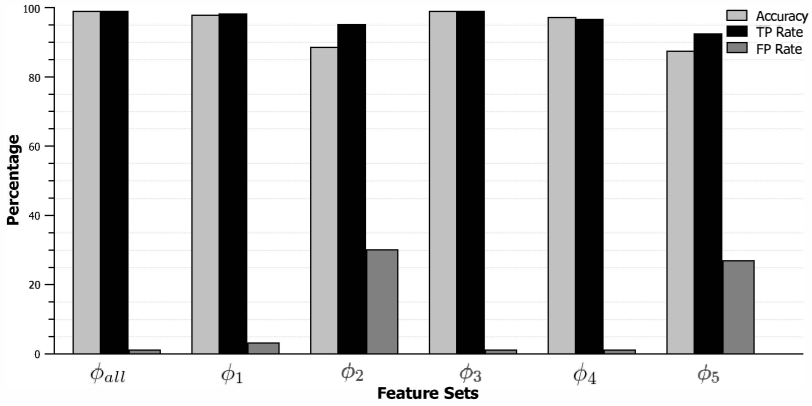

Fig. 1: Percentage of accuracy, true positive, and false positive rates

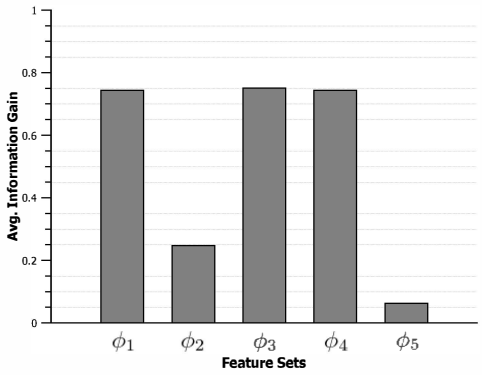

Fig. 2: Average information gain value for each feature type

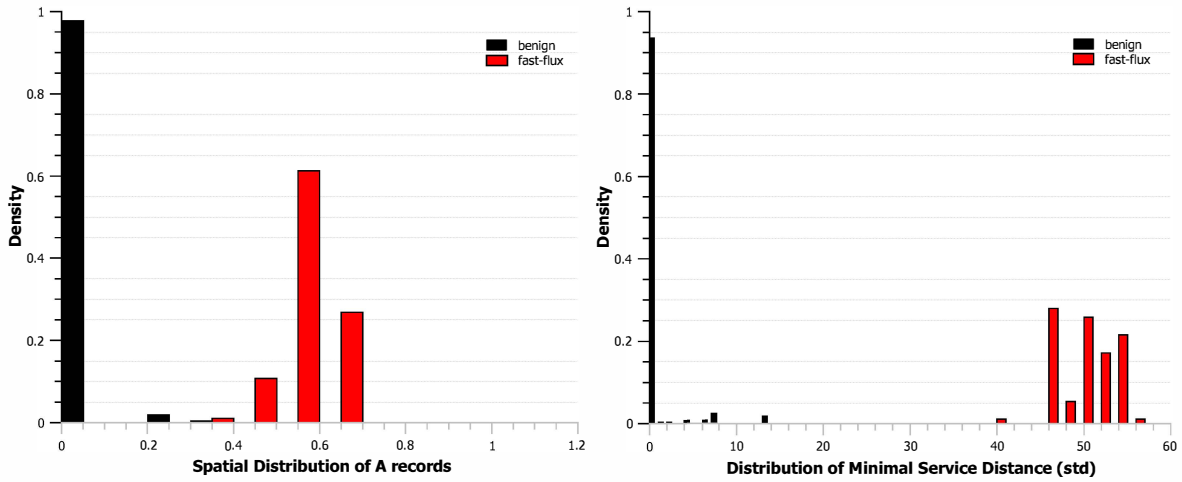

Fig. 3: Distribution of the two highest information gain ranked spatial features

using ensemble of classifiers with a voting procedure.

In order to investigate the performance of the spatial-based features, we extracted the class-conditional feature histograms from benign and fast-flux datasets for the two highest information gain ranked features (*i.e.*, spatial distribution of A records and standard deviation of minimum service distance) as shown in Figure 3. We observe that distributions have a different range and form, and such differences do provide a foundation of discriminating fast-flux networks. Nearly all benign instances were associated with four or fewer A and NS records, while most of the fast-flux instances were associated with more A and NS records in one single DNS response packet. Even if FFSNs and CDNs are distributed over multiple networks and geographical locations, the IP addresses returned by CDNs have closer service distance. As a result, as expected, the use of service distance leverages the CDN detection and false positive rates decreases from 3.2% to 1.1% compared with the use of only number of IP addresses in A and NS records. Our spatial experimental results obtained are in general similar to those obtained by [8] and [14]. It should be noted that in our implementations, IP-coordinate database is well maintained for each query without any missing values.

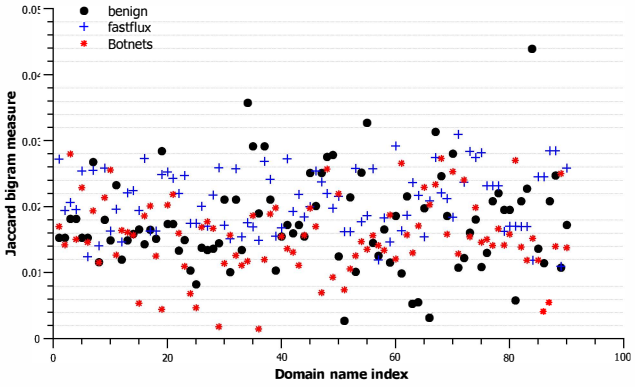

Fig. 4: Scatter plot of Jaccard distance for bigrams using 90 domain names

The resulting accuracy, TP and FP rates of domain-name based features are 88.48%, 95.1%, and 30.1% respectively. In general, our experiments reveal that overall the KL Divergence outperforms others, followed by Jaccard index and then Edit distance. The scatter plot presented in Figure 4 shows the

Jaccard Index over bigram distributions of 90 test domains between benign and fast-flux domain names, as well as domain names extracted from traditional botnets. Note that fast-flux domains used in our experiments are all easy-to-remember, human readable and consist of at least 11 characters such as *sportinghookup.com* and *findpartnertoday.com*, which adds an extra layer for textual detection. Even we observe that FFSN domain names do not closely match the true distribution for unigrams and bigrams compared to benign domain names, we believe that more sophisticated probability models can be adapted to evaluate the likelihood of a given domain name for estimating association with FFSNs, and may give better results compared to metrics used in our experiments.

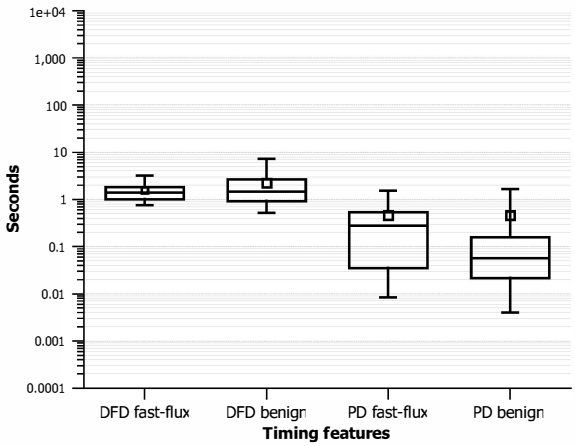

Fig. 5: Comparison of standard deviation of document fetch delay (DFD) and processing delay (PD)

Finally, Figure 5 shows the logarithmic scale box plot of standard deviation of document fetch delay (DFD) and processing delay (PD) distribution for benign and fast-flux classes. We observe that there is no significant difference between mean values of the classes. Since network conditions, limitations of slow servers, powerful bots, application process time, and the size of the congestion window may affect the features, it is observed that the C4.5 algorithm is invariant based on the way the timing-based features are handled.

V. Conclusion

In this paper, we developed and evaluated features for detection of FFSNs that are delay-free, and lightweight in terms of storage and computational cost. We extracted five different sets of features out of feature space of size 19, and studied by using the C4.5 algorithm to evaluate discrimination power of each set. A primary motivation for our study was that these features are often used by research community; however, they are not jointly investigated or compared for the FFSN detection. In order to address these problems, we demonstrated the reliability of each feature by measuring the information gain for each class, and evaluated in terms of accuracy, false and true positive rates. Finally and most importantly, we addressed exploiting the joint use of domain name and DNS packet characteristics. It is observed that spatial along with network and DNS answer features lead to the best classification results, and the domain name features are the promising ones for the datasets we recorded. As a future work, we plan to investigate dependencies between the features using more sophisticated classifiers, and more advanced probability models for the domain name features.

Acknowledgment

The authors would like to thank Dr. George Kesidis for his guidance, and Dr. David J. Miller and Fatih Kocak for their constructive comments.

References

[1] "The Honeynet Project. Know your enemy: Fast-Flux Service Networks," *http://www.honeynet.org/papers/ff*, 2007.

[2] R. Perdisci, I. Corona, and G. Giacinto, "Early detection of malicious flux networks via large-scale passive dns traffic analysis," in *Proc. IEEE Dependable and Secure Computing*, 2012.

[3] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "EXPOSURE: Finding malicious domains using passive DNS analysis," in *Proc. Network and Distributed System Security Symposium (NDSS)*, 2011.

[4] T. Holz, C. Gorecki, K. Rieck, and F. Freiling, "Measuring and detecting fast-flux service networks," in *Proc. Symposium on Network and Distributed System Security*, 2008.

[5] E. Passerini, R. Paleari, L. Martignoni, and D. Bruschi, "Fluxor: detecting and monitoring fast-flux service networks," in *Proc. Detection of Intrusions and Malware, and Vulnerability Assessment*, 2008.

[6] C. Su, C. Huang, and K. Chen, "Fast-flux bot detection in real time," in *Proc. Recent Advances in Intrusion Detection*, 2011.

[7] S. Yadav, A. Reddy, A. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proc. 10th annual conference on Internet measurement*, 2010.

[8] S. Huang, C. Mao, and H. Lee, "Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection," in *Proc. 5th ACM Symposium on Information, Computer and Communications Security*, 2010.

[9] R. Bellis, "RFC 5625, DNS Proxy Implementation Guidelines (Best Current Practice)," *www.ietf.org*, 2009.

[10] M. Knysz, X. Hu, and K. Shin, "Good guys vs. bot guise: Mimicry attacks against fast-flux detection systems," in *Proc. INFOCOM*, 2011.

[11] D. Miller, F. Kocak, and G. Kesidis, "Sequential anomaly detection in a batch with growing number of tests: Application to network intrusion detection," in *Proc. IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2012.

[12] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: Analysis of a botnet takeover," in *Proc. IEEE Workshop on Network Security*, 2009.

[13] B. Bigi, "Using kullback-leibler distance for text categorization," *Advances in Information Retrieval*, 2003.

[14] H. Wang, C. Mao, K. Wu, and H. Lee, "Real-time fast-flux identification via localized spatial geolocation detection," in *Proc. Computer Software and Applications Conference (COMPSAC)*, 2012.

[15] G. Combs, "Tshark, dump and analyze network traffic," http://www.wireshark. org.

[16] R. Kohavi and R. Quinlan, "Decision tree discovery," in *Handbook of Data Mining and Knowledge Discovery*, 1999.