Fast flux botnet detection framework using adaptive dynamic evolving spiking neural network algorithm

Al-Nawasrah, A, Al-Momani, A, Meziane, F and Alauthman, M

http://dx.doi.org/10.1109/IACS.2018.8355433

Fast Flux Botnet Detection Framework using Adaptive Dynamic Evolving Spiking Neural Network Algorithm

Ahmad Al-Nawasrah
College of Computing, Science and Engineering, Salford University, Manchester M5 4WT, UK,
a.alnawasreh@edu.salford.ac.uk

Ammar Al-Momani
IT-department, Al-huson University College, Al-Balqa Applied University, P. O. Box 50, Irbid, Jordan,
ammarnav6@bau.edu.jo

Farid Meziane
College of Computing, Science and Engineering, Salford University, Manchester M5 4WT, UK,
f.meziane@salford.ac.uk

Mohammad Alauthman
Department of Computer Science, Faculty of information technology, Zarqa University, Jordan
malauthman@zu.edu.jo

Abstract*— A botnet, a set of compromised machines controlled distantly by an attacker, is the basis of numerous security threats around the world. Command and Control servers are the backbones of botnet communications, where the bots and botmasters send report and attack orders to each other. Botnets are also categorized according to their C&C protocols. A Domain Name System method known as Fast-Flux Service Network (FFSN) – a special type of botnet – has been engaged by bot herders to cover malicious botnet activities and increase the lifetime of malicious servers by quickly changing the IP addresses of the domain name over time. Although several methods have been suggested for detecting FFSNs, they have low detection accuracy especially with zero-day domain. In this research, we propose a new system called Fast Flux Killer System (FFKS) that has the ability to detect FF-Domains in online mode with an implementation constructed on Adaptive Dynamic evolving Spiking Neural Network (ADeSNN). The proposed system proved its ability to detect FF domains in online mode with high detection accuracy (98.77%) compare with other algorithms, with low false positive and negative rates respectively. It is also proved a high level of performance. Additionally, the proposed adaptation of the algorithm enhanced and helped in the parameters customization process.

Keywords: Fast-Flux, dynamic evolving spiking neural network, ADeSNN, botnet detection.

I. INTRODUCTION

Botnets comprising networks of compromised computers that are controlled remotely by attackers are the basis of numerous security threats, such as distributed denial-of-service (DDoS) attacks, identity theft, phishing, and spam [1-3]. Fast flux networks (FFNs) are a special type of botnet being used by criminals in the same manner as those used in round-robin domain name systems (RRDNSs) and content distribution networks (CDNs) to offer high availability and flexibility for their malicious websites [1]. Botnet writers disguise their malicious activities and design new tactics and mechanisms to hide their communications. The core idea of FFNs is to use bot computers as proxies (flux agents) that forward user queries to backend servers called "motherships." A recurrent and fast change in the IP addresses of proxies is essential to evade detection and a potential shut down and to ensure high availability to those backend servers.

The report of the Cost of Cyber-Crime Study [4] points out that the mean annualized cost of cyber-attacks for 252 benchmarked organizations is $7.7 million/year. The report also shows that these attacks are carried out with or supported by either a botnet or a web-based attack, and fast flux is used as evasion technique to provide availability and resiliency.

Dynamic evolving spiking neural network (DeSNN), that employ both rank order (RO) learning algorithm and dynamic synapses to learn spatial and temporal data in a fast and on-line mode. By employing both RO learning and Spike Driven Synaptic Plasticity SDSP, they will be discussed in details later in this paper. ADeSNN is constructed using two public published datasets, Fast Flux [5] and IRIS [6] datasets, respectively. The target in this paper is to show the proposed solution of the fast flux problem, prove that with the proposed modification on DeSNN the adaptive version shows promising results, and to minimize the number of the parameters needed to be set into the algorithm.

One of the core problems in botnet detection is the so-called unknown "zero-day" fast flux domain. Zero-day domains are defined as those related to bots (FF-agents) that are not blacklisted [7]. A fast flux attack is a complex evasive technique that cannot be identified by many current techniques because attackers can use new and previously unseen bots. A number of potential solutions to fast flux botnet attacks have been proposed, but these solutions are not yet effective. These solutions range from passive, active, to real-time approaches. The misclassification of malicious and legitimate domains increases with time, especially when dealing with unknown zero-day fast flux botnet domains. The rest of this proposal is arranged as follows. A problem of fast flux botnets are provided in Section 2. Research objectives are mentioned in section 3. The related work is presented in Section 4. The proposed solution is discussed in detail in Section 5. Finally, Section 6 presenting the conclusion and the expected outcomes.

II. LITERATURE REVIEW

Numerous studies have explored botnet detection, especially fast flux botnet detection. Most previous research discussed the detection of FFSNs or malicious fast flux domains, which serve as the main element of the fast flux botnet technique. The related

works on fast flux argued about fast flux in terms of what is fluxed or what technique is used to detect an FF domain. However, to the best of our knowledge, the present study is the first to investigate fast flux botnet approaches on the basis of the solution scope of detection techniques. In the present work, fast flux botnet approaches are classified according to the solution scope (Host-based, Router-based, and DNS-based methods). Moreover, the mode of each detection technique is discussed below and identified whether it is active, passive, or real time.

A. Host-based detection methods

A detection system is applied to a host device or a set of devices from the user point of view. According to the previous work, the majority was a host-based detection approaches, these approaches are divided into three subsections: passive, active, and real-time approaches such as in [8, 9]. Fast flux detection requires a fast and accurate approach to identify malicious domains before they change their IP addresses. Thus, real-time approaches have been developed to increase the power of detection techniques. Real-time approaches could detect malicious domain names in most cases. However, the above-mentioned techniques have certain limitations, which cast doubt on their results (accuracy, TP, TN, FP, and FN). We still lack a stable technique that can detect malicious domains, particularly zero-day domains, in an acceptable period of time with high detection accuracy.

B. Router-based detection methods

Various information extracted from network traffic to solve several network problems, generally and particularly for the fast -flux botnet problem. Network traffic comprises both DNS traffic and non-DNS traffic such in [10, 11] However, the speed and the large amount of data passing through the router cause problems to any proposed systems: high false rates based on the concept of a fast detection of FF botnets, memory problems (databases) with regard to handling large traffic data flows, and scalability problem. Therefore, detecting fast flux botnets and particularly zero-day domains at this part of the network is not an easy task.

C. DNS-based detection methods

Researchers studied DNS data traffic in their country of origin. Thus, their work focused on monitoring and analyzing DNS data traffic and detecting malicious activities, specifically fast flux activity. Some researchers employed passive, active, and real-time approaches such as in [10, 12].

The detection systems initiated over a DNS server do not exhibit network time delays. Many researchers determined that systems that depend on DNS features cannot provide an accurate detection rate for fast flux domains [13].

The main problem of fast flux botnet detection methods is detecting the evasion detection mechanism before the attack is initiated to support botnet malicious activities, particularly when detecting zero-day domains without any prior knowledge about the incoming domain name, which serves malicious websites/C2 servers/ motherships. At the same time, detection accuracy and low detection error rates are monitored. On the basis of the developing strategies of attackers, the detection system should develop new systems that are long-lasting and adaptive to allow the future modification of their functions. As the proposed framework promises as one of the host-based methods.

III. THE PROPOSED SOLUTION

The expected system is an online detection system, and it's going to deal with real data so spiking neural network (SNN) is conducted. Systems based on SNN have already shown their ability in capturing spatial and temporal data. Adaptive Dynamic evolving Spiking Neural Network (ADeSNN), based on a one-pass rank order (RO) learning rules and a scheme to evolve a new spiking neuron and connections, which lead to learning new patterns from arriving data. This paper introduces a new type of DeSNN, which employ both RO learning and dynamic synapses to learn spatial and temporal data in a fast and on-line mode, however (in our proposed version the initial weight is replaced by the spiketime of the input record)for improving the learning process . By employing both RO learning and Spike Driven Synaptic Plasticity SDSP, ADeSNN could be used in unsupervised, supervised, or semi-supervised learning mode. The SDSP learning is used to dynamically update the connection weights of the network that capture data clusters both through training and through recall.

$$\omega_{j,i} = spiketime_{j,i} \tag{1}$$

After initiation of the weight on the synapses of j neuron based on the spiketime matrix of that input record as in formula (1), the dynamic synapses adjust their weights based on the SDSP algorithm. While the spikes arrive at any time t then its value increases, as there is no spikes arrive at this time its value decreases:

$$\Delta\omega_{j,i}(t) = e_j(t).D \tag{2}$$

Where $e_j(t)$ equals to 1, if there is a sequenced spike timely t arrives at synapse j at the time of arriving the learning patterns at the output neuron i, and its equals to (-1) otherwise. D is the drift parameter, which is able to be changed for up or down drifts.

In parallel, all synapses change their values in every time t unit, while the input patterns Pi arrive at the output neuron i. based on this values which may go up or down, the synapses altogether of the neuron could capture nearly relationships of spike timing through the learned pattern. Continuously, as the incoming training patterns arrive (input spikes on different synapses), they are encoded within the time window T. Which then the threshold Th_i of the neuron defined. Based on the value of this threshold the neuron i spikes or not. The threshold is defined as a fraction of the entire PSPi (PSPimax) collected through the appearance of the Entire input pattern

$$PSP_{imax} = \sum_{t=1,2,\dots,T} \sum_{j=1,2,\dots,M} f_j(t).\omega_{j,i}(t) \tag{3}$$

$$Th_i = C.PSP_{imax} \tag{4}$$

Where: T is the time window in which the input patterns arrived, M is the number of neurons I input synapses, $f_j(t)$ equals to 1 if the spike appears in time window at the synapse j for this input pattern, if not it equals to 0. $\omega_{j,i}(t)$ is the

efficacy of the dynamic synapse between the neurons *j,i* which calculated in equation (2).

Fig.1 shows the architecture of the ADeSNN algorithm, also positions the rank order encoding method based on multiple Gaussian receptive fields. In addition, the figure presents the SDSP learning rule, which adjusts the synapses weights, these weights change up /down based on the drift parameter value, which discussed before.

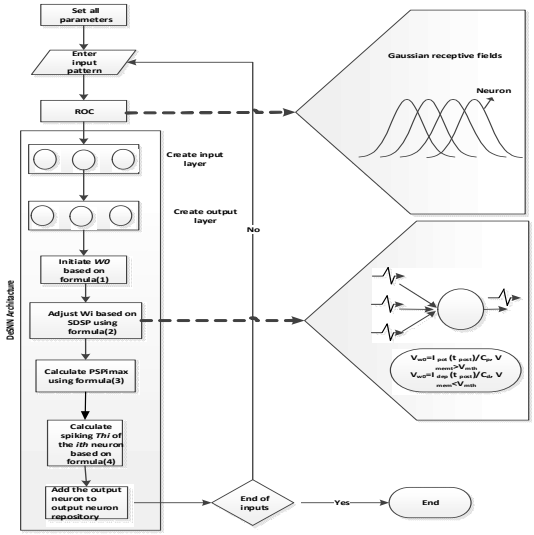

Fig. 1 the flowchart of the ADeSNN architecture

IV. DATASET

The proposed ADeSNN method evaluated and tested using two datasets. Firstly, a public fast flux dataset is chosen [14], which used in [15], and the majority of the real-time and active approaches used the same sources [8, 13]. This dataset consists of DNS responses of labeled domain names as benign and fast flux. The benign domains are selected and labeled based on the source of the top trusted websites like Alexa, top blogs as Blogs on Top "BOT". While the fast-flux domains are collected from the famous fast flux blacklisted websites such as ATLAS, DNSBL and FluXOR (information security expert's detection systems). This dataset is conducted to test the proposed system efficiency of detecting fast flux domains. Besides, the dataset found as a response of the DNS server answers. So, a script of python is written for feature extraction and analysis.

Secondly, the public IRIS dataset, which consists of 3 classes each with 50 instances, this dataset is the best known dataset for pattern recognition. One class is linearly separable from the other two, but the latter two classes are not separable from each other. The IRIS dataset has 4 features which are (sepal length, sepal width, petal length, and petal width), this data is used in order to prove that the ADeSNN shows better classification result than the DeSNN itself.

V. FEATURE SET

The first stage at the proposed solution is the feature extraction. A well-built fast flux botnet detection method should distinguish between a legitimate and malicious network (domain). On the other side, a well-built Fast Flux Network (FFN) seems like a benign CDN, by returning a records that belong to the same close geographic areas. This leads the detection systems that depend on IP address features to misclassify those types of well-built FFN as benign CDNs. In addition, the FFN developers are trying to change the characteristics of the fast flux Network to evade detection, even if this modification is reflected in the performance of the FFN. Therefore, a new detection system should rely on features belonging to the FFN itself, as these features are not prone to change quickly.

Based on the current chosen dataset, some of the features used in the proposed solution are used before in related works, moreover, new features are suggested to improve the accuracy of classifying the fast flux and benign domains. Table 1 shows the selected features set of the first dataset:

TABLE 1 SELECTED FEATURES SET

Feature	Description	New feature
IPans	Number of IP addresses in the answer section	-
NSadd	Number of IP addresses in the additional section	-
NASN_ans	Number of ASN for the IP addresses of the answer section	-
NASN_add	Number of ASN for the IP addresses of the additional section	-
AVGSIM	The average of similarity of the ASN	√
Qtime	Time of the query	√
Msgs	Message size	√

The majority of the chosen features are known, but the following feature is new; AVGSIM which refers to the average of the similarity between the autonomous system number of the user's IP himself and autonomous system numbers of the proxy bots (compromised computers) returned in the answer section of the DNS response, and is computed according to the formula (5) [16]:

$$M_i(\mu(e), \delta(e)) = 1 - \frac{\sum_{j=1}^{n} |\mu_{ij}(e) - \delta_{ij}(e)|}{\sum_{j=1}^{n} |\mu_{ij}(e) + \delta_{ij}(e)|} \quad (5)$$

It could be said that $x\mu$ and $y\delta$ are significantly similar if $M(x\mu,y\delta) \geq ½$, which means that the bigger value refers to high similarity between the two variables (vectors). Based on this measure, the similarity among the new input and the trained samples to find the close-trained input that matches the new one. The other two new features are the time of the query and the DNS packet size. The feature set of the second dataset is mentioned above in the dataset section.

VI. EXPERIMENTS AND DISCUSSION

The software and hardware used in those experiment were based on Linux mint operating system rum core i7 7500U CPU, 16GB RAM, a simulations of the compared method were conducted using Matlab 8.5 and Python 2.7 environments. Two experiments are conducted in this paper; the former is based on the first dataset- fast flux dataset. This experiment showed the ability of the proposed system to detect the fast flux domains. Each record contains the selected feature set that helps to identify each class. Fast flux dataset contains (1710) instances, while the

benign dataset contains (1790) instances. 3-flod cross-validation is used to do 3 experiments, then the average has been taken, all the results and discussions are presented below. In addition, the two related classifiers (Linear decision function, and C4.5) were been compared with the proposed algorithm to present the advantages over the related work.

Due to evaluate the proposed algorithm various detection accuracy measures are conducted (true positive rate TPR, true negative rate TNR, false positive rate FPR, false negative rate FNR, area under rock-curve AUC, and F-measure), and the results of these measures are presented in the table 2 and 2 to Shows the accuracy measures of the detection of the proposed and the other two algorithms (Linear decision function, C4.5, and DeSNN)

TABLE 2 COMPARISON RESULTS OF THE FIRST EXPERIMENT

	C4.5	Linear	ADeSNN
FNR	0.069869	0.0393013	0
FPR	0.0533333	0.0533333	0.0240964
TPR	0.930131	0.9606987	1
TNR	0.9466667	0.9466667	0.9759036
AUC	0.9383988	0.9536827	0.9879518
F-measure	0.938326	0.9544469	0.9875

According to Lin et al.[7] a genetic approach was proposed as a real-time detection solution of the fast flux domains problem. This method suggested two-detection feature to classify the benign and the flux domains. Firstly, entropy of the domain name (E-DPN) of the preceding node of the flux node (flux-agent), by using the traceroute of all the returned IPs from the DNS response. Of course, if the E-DPN is high the most probably the domain classified as benign otherwise classified as fluxed.

Secondly, the Standard Deviation of Round Trip Time (SD-RTT) between the user and all the return IPs of the flux-agents, so assumed that the scatter flux-agent with is going to produce a high value of the SD-RTT. This spatial feature takes the number of different ASNs and number of IPs return in single DNS response in their calculations. However, this two detection features evaded by the botmaster, as botmaster is controlling the returned list of IPS that the user receives. Which the returned list could has that IPs in the same ASN or adjacent the user ASN, so the above measures can inaccurately classify the benign and flux domains. On the other hand, botmaster may return a list contains just a single IP address, which leads to ineffective detection of the domains[17, 18]. Although genetic algorithms provide good accuracy, but they are very complex and take noteworthy time in building data models. Also, the linear decision function used as classifier needs to estimate the categorizer of the linear function, so if the estimation is good then the linear function work properly, otherwise the error will be high in the classification process [19].

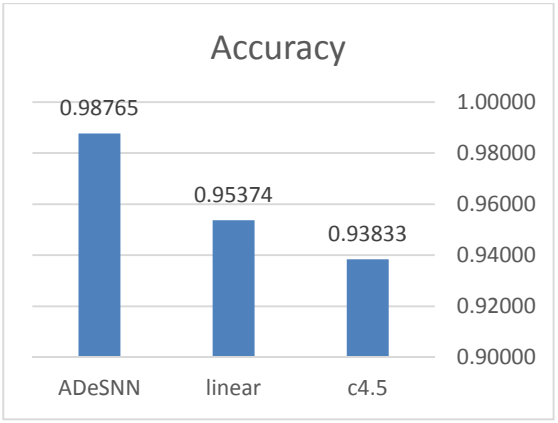

Fig.2 The overall detection accuracy

On the other hand, the second compared algorithm is the C4.5 as proposed in [20]. A number of feature sets were examined to detect fast flux network, such feature set consist of timing based, spatial based, network-based domain based, and DNS answer based feature sets. As mentioned in the literature the data set was small even the accuracy of the experiment is high, also when all features are involved in the experiment the prediction results become insensitive to two features (timing and domain-based feature sets) [18]. Besides, as C4.5 algorithm considered as a supervised learning algorithm, it could not be used to discover the unknown attacks especially the zero-day fast flux domains.

For the second experiment, a public dataset was used; IRIS dataset has non-leaner separable attributes. Experiments of 3-fold cross-validation datasets were done and then the average was taken to present the results shown in the table 3 and Fig 3 below:

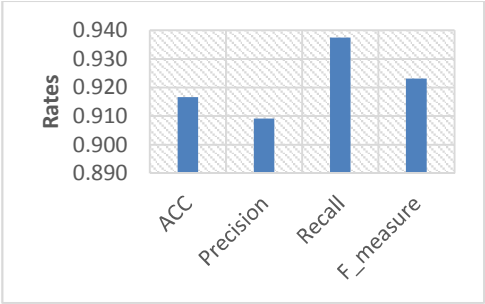

Fig.3 Accuracy, Precision, Recall and F-measure of ADeSNN

TABLE 3 EXPERMENT RESULTS

Measures	Rates
FNR	0.0625
FPR	0.1071
TPR	0.9375
TNR	0.8929
AUC	0.9152

For the sake of space, the comparison between the DeSNN and the ADeSNN is omitted, and a short comparison was made. The overall accuracy of DeSNN was 56.0 while it was 91.67 for ADeSNN, and the other measures were depicted in fig 3,

according to the IRIS dataset two classes were non-linearly separable which case to show almost 91% accuracy while it was tested on the first linearly separable classes and give 100% accuracy. This leads to the ability of the adaptive DeSNN to classify classes even when inputs are mutually mixed. The algorithm of the DeSNN suffers from many parameters needed to be set before running the algorithm, so one of the future work should take into account to minimize the number of the parameters to be set. Our adaptive DeSNN modifies the process of the initial weight setting, so this adaptation added a value in the parameters customization problem by excluded the (Mod) parameter.

VII. CONCLUSION

A botnet, a set of compromised machines controlled distantly by an attacker, is the basis of numerous security threats around the world. Command and Control servers are the backbones of botnet communications, where the bots and botmaster send report and attack orders to each other. Fast-Flux Service Network (FFSN) – a special type of botnet – has been engaged by bot herders to cover malicious botnet activities and increase the lifetime of malicious servers by quickly changing the IP addresses of the domain name over time. Although several methods have been suggested for detecting FFSNs, they have low detection accuracy especially with the zero-day domain, quite a long detection time and consume high memory storage. In this research, we propose a new system called Fast Flux Killer System (FFKS) that has the ability to detect FF-Domains in online mode with an implementation constructed on Adaptive Dynamic evolving Spiking Neural Network (ADeSNN). The proposed system proved to detect fast flux domains with high detection accuracy according to the experiments, a comparison was conducted with two related work in the same area and the results were excellent. Also, the adaptive DeSNN showed an enhancement in its classification performance regarding the experiment done on the public dataset. Besides, the adaptive algorithm contributed to the parameters customization problem as mentioned before.

References:

[1] K. Alieyan, A. ALmomani, A. Manasrah, and M. M. Kadhum, "A survey of botnet detection based on DNS," *Neural Computing and Applications*, pp. 1-18, 2015.

[2] A. Almomani, B. Gupta, S. Atawneh, A. Meulenberg, and E. Almomani, "A survey of phishing email filtering techniques," *Communications Surveys & Tutorials, IEEE,* vol. 15, no. 4, pp. 2070-2090, 2013.

[3] A. Almomani, T.-C. Wan, A. Manasrah, A. Altaher, M. Baklizi, and S. Ramadass, "An enhanced online phishing e-mail detection framework based on evolving connectionist system," *International Journal of Innovative Computing, Information and Control (IJICIC),* vol. 9, no. 3, pp. 169-175, 2013.

[4] H.-P. Enterprise. "2015 Cost of Cyber Crime Study: Global," http://engage.hpe.com/LP_510004609_HPSW-ESP_WW_EN-US_PonemonGate.

[5] "Public Fast Flux dataset: URL:https://sites.google.com/site/huangpublication/datasets/-1-fast-flux-attaack-datasets . Access date: 1/1/2017

[6] Z. Benjamin, and F. R.A., *Iris DataSet*, 2013.

[7] H.-T. Lin, Y.-Y. Lin, and J.-W. Chiang, "Genetic-based real-time fast-flux service networks detection," *Computer Networks,* vol. 57, no. 2, pp. 501-513, 2/4/, 2013.

[8] M. T. Qassrawi, and H. L. Zhang, "Detecting Malicious Fast Flux Domains." pp. 1264-1273.

[9] W. Horng-Tzer, M. Ching-Hao, W. Kuo-Ping, and L. Hahn-Ming, "Real-Time Fast-Flux Identification via Localized Spatial Geolocation Detection." pp. 244-252.

[10] B. N. Al-Duwairi, and 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, 7//, 2014.

[11] T. Paul, R. Tyagi, B. Manoj, and B. Thanudas, "Fast-flux botnet detection from network traffic." pp. 1-6.

[12] J. Kwon, J. Lee, H. Lee, and A. Perrig, "PsyBoG: A scalable botnet detection method for large-scale DNS traffic," *Computer Networks,* vol. 97, pp. 48-73, 3/14/, 2016.

[13] S. Martinez-Bea, S. Castillo-Perez, and J. Garcia-Alfaro, "Real-time malicious fast-flux detection using DNS and bot related features." pp. 369-372.

[14] H. Hagras, A. Pounds-Cornish, M. Colley, V. Callaghan, and G. Clarke, "Evolving spiking neural network controllers for autonomous robots." pp. 4620-4626.

[15] S.-Y. Huang, C.-H. Mao, and H.-M. Lee, "Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection," in Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, Beijing, China, 2010, pp. 101-111.

[16] S. Alkhazaleh, A. R. Salleh, and N. Hassan, "Possibility fuzzy soft set," *Advances in Decision Sciences,* vol. 2011, 2011.

[17] F.-H. Hsu, C.-S. Wang, C.-H. Hsu, C.-K. Tso, L.-H. Chen, and S.-H. Lin, "Detect fast-flux domains through response time differences," *Selected Areas in Communications, IEEE Journal on,* vol. 32, no. 10, pp. 1947-1956, 2014.

[18] T. Otgonbold, "ADAPT: An anonymous, distributed, and active probing-based technique for detecting malicious fast-flux domains," 2014.

[19] P. S. Chahal, and S. S. Khurana, "TempR: Application of Stricture Dependent Intelligent Classifier for Fast Flux Domain Detection," *International Journal of Computer Network & Information Security,* vol. 8, no. 10, 2016.

[20] Z. B. Celik, and S. Oktug, "Detection of fast-flux networks using various dns feature sets." pp. 000868-000873.