Enhancing Automatic Attack Detection through Spectral Decomposition of Network Flows

Lucas Airam C. de Souza, Gustavo F. Camilo, Miguel Elias M. Campista,
Luís Henrique M. K. Costa, Otto Carlos M. B. Duarte

Grupo de Teleinformática e Automação (GTA) - COPPE/Poli
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro, Brasil

***Abstract*—Flow classification employs machine learning techniques to identify attacks on computer networks. This classification relies on quantitative features that synthesize the information of packets from the same flow. Conventional features, however, such as packet size and the number of bytes, generate redundancies and do not capture the temporal correlations between the packets in a flow. Automated network attacks generate periodic patterns observable through spectral decomposition, which facilitates classification. This paper proposes FENED (Feature Extraction by Network spEctrum Decomposition), a method to extract features from network data. We consider the packet-arrived order within the same flow using the fast Fourier transform for binary classification. The proposed feature vector contains the module of the spectral components of the flow. Experimental results show that FENED outperforms conventional proposals because it extracts features that consider intra-flow packet-arrival order.**

***Index Terms*—Machine Learning; Feature Engineering; Network Security; Botnet Detection; FFT.**

I. Introduction

Botnets are today the biggest threat to computer networks. The infection of thousands of devices boosts damages while complicating countermeasures [1]. In the Internet of Things (IoT) scenario the situation is even more concerning, since IoT devices are usually resource-constrained and as such employ simpler security mechanisms, an easy target to botnets [2]. Communication with infected devices and attacks from botnets are automated, allowing rapid dissemination. In this scenario, machine learning technology emerges as a countermeasure to detect traffic patterns and mitigate network threats. Dataset feature extraction is the most critical step in creating a machine learning algorithm, as features directly affect model parameters tuning and classification or regression performance. Among the feature extracting methods and network flows representation, there are conventional ones, which quantify packet information, such as the number of bytes or the number of packets [3]. There are also more complex representations based on graphs [4] or even images [5]. None of these features, however, reflect the temporal dependency between the packets or the periodic behavior generated by the attack automation.

This paper proposes FENED[1] (Feature Extraction by Network spEctrum Decomposition)[2], a method for extracting features from network flows using the Fast Fourier Transform (FFT). FENED extracts a vector of features with the spectral components of the network flows for the training and testing of machine learning algorithms. The flow representation in the frequency domain reflects the temporal dependency of packets from the same flow and shows typical periodical behavior of botnets and automated attacks. The proposed method uses the size of the packets to generate the spectral components, being adaptable for analysis of encrypted traffic and preserving the privacy of the samples by not manipulating sensitive data. Moreover, the flow classification employing the proposed features is agnostic to the type of classifier adopted.

The obtained results show that the proposed method improves the performance of network flow classification in most experimental cases. The comparison considers the conventional method applied to feature extraction, which only uses counter features, i.e, number of packets and flow size. The vector dimension applied in the FFT calculation does not statistically change the classification metrics obtained. As a consequence, the proposal can operate with smaller windows to reduce computational costs while maintaining high classification performance. Finally, our proposal achieves similar or better classification results using significantly fewer features than the conventional feature extraction method.

This paper is organized as follows. Section II reviews the state of the art in methods for extracting features from computer network flows. Section III presents the proposed method. Section IV details the development of the prototype to experiment with FENED, and analyzes the results obtained. Finally, Section V concludes the paper and presents future directions.

II. Related Work

Feature extraction is a primary step in machine learning, in which the data used to adjust the parameters of the model and test it is selected. In network traffic analysis, it is common to

[1]A preliminary version of this paper was published in Portuguese and is available at http://www.gta.ufrj.br/ftp/gta/TechReports/SCD21.pdf.

[2]Available at https://github.com/GTA-UFRJ-team/FENED.git.

use the TCP/IP 5-tuple as flow identifier (source IP address, destination IP address, protocol, source and destination port), and features related to the packet data, such as packet size and number of packets in the flow, among others.

A. General Approaches for Attack Detection

The most popular traffic analysis tools, such as Netflow[3] and sFlow[4], use this approach. Flowtbag[5] is a traffic analysis tool that processes a network packet capture file and extracts 40 numerical features of flows identified by the 5-tuple [6]. Most of the proposals for intrusion detection [7], [3] are based on this type of flow features. Nevertheless, different research on feature engineering propose alternative representations of network traffic, such as graphs [4], images [5], or spectral analysis [8].

Sanz *et al.* [4] and Sagirlar *et al.* [1] propose network graph representations to detect malicious flows. The features extracted from graphs exhibit relationships between different flows instead of information from a single flow. Nevertheless, the graph construction process is CPU-intensive, making it difficult for these proposals to detect attacks in real-time. Another approach to a network flow representation is through an image that exhibits the difference between malicious and normal traffic [5]. This technique also has high computational cost. Still, image classification is easily performed by deep learning algorithms, which are highly parallelizable through Graphical Processing Units (GPU) and reduce computation overhead.

Another approach for classifying flows is constructing a time series with information of network packets [9]. The classifier uses statistical data obtained from the time series to predict the flow class. Nonetheless, we can also represent the time series in the frequency domain.

B. Spectral Analysis for Attack Detection

The frequency domain exposes the tendency of recurrent behaviors, i.e, the transformation can decompose a periodic signal in time in its frequency components. We can use this representation to detect automated network threats, such as botnet attacks. The spectral analysis of the flows additionally allows distinguishing discontinuous behaviors, such as denial of service (DoS) attacks with low rates [10], or periodic behaviors.

Communication patterns of automated attacks from botnets are periodic, while normal flows have a spectrum similar to the white noise [11]. Thus, the spectral analysis of network flows allows characterizing the periodicity and identifying attacks [8]. Zhou *et al.* propose the use of flow spectrum analysis in conjunction with the measure of variance and average of the packets size to detect automatically generated malicious behavior [12]. Chimedstseren *et al.* employ the Fourier transform for intrusion detection [13]. The authors define a flow through the source and destination IP addresses pair and create a time series containing the size in bytes of the packets in each flow. They assign a positive magnitude signal to packets sent by the source and received by the destination and a negative signal for the opposite direction. The purpose of the magnitude signal is to obtain a signal with temporal oscillation and verify the periodicity in the communication pattern between client and server.

AsSadhan and Moura propose to analyze the network traffic periodogram for botnet detection [14], while Manasrah *et al.* use the spectral density [15]. The botnet automated behaviors imply periodic components with high module. The authors use the autocorrelation function in the extracted periodogram to identify the IPs belonging to the robot network. PsyBoG [16] is a mechanism that analyzes the periodicity of DNS traffic to detect botnets. Nevertheless, the proposals are limited to searching the network traffic for periodic components with high module that go beyond a threshold to detect attacks instead of analyzing other spectrum properties.

Yu *et al.* rely on the Discrete Fourier Transform (DFT) property that preserves Euclidean distance to measure the similarity between flows and identify attacks [17]. From the DFT, the authors create a vector with a reduced number of spectral components for calculating the distance between flows since the first coefficients of the lower frequencies retain most of the spectral energy. Fouladi *et al.* propose to apply the Fourier transform to detect distributed denial of service (DDoS) through the similarity measure between each flow [18]. The proposals consider that the flows belonging to the same botnets have high similarity. Thus, two malicious flows are closer than normal flows. Therefore, the authors measure the distance between the components of two flows and classify as suspicious any flow that is below the established threshold. The proposals present a low number of false positives, however, the authors only use the measure of distance between extracted components to identify the attacks.

Powell proposes the use of the discrete Fourier transform for the detection of extrusion [19]. The data extrusion occurs when an attacker access a network or system and leaks inside information. The Fourier transform extracts features with temporal correlations and allows a classification of traffic with low false-positive rates. Nevertheless, the author limits the evaluation of the proposal to a set of extrusion detection data and do not evaluate the proposal for a set of botnet data. The use of spectral decomposition techniques aimed at botnets has advantages since automated attacks have spectral patterns easily distinguishable from normal traffic patterns.

Unlike the previous works, FENED extracts a vector of features representing the spectral components of network flows for the training and testing of supervised machine learning algorithms.

The main contributions of this paper are:

- A simple feature extraction method for network analysis. While other works use a large number of features, our

[3]Available at https://cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html.

[4]Available at https://sflow.org/.

[5]Available at https://github.com/DanielArndt/flowtbag.

proposal can achieve high classification performance in precision, recall, and F1-score, with a lower number of features.

- A model agnostic feature extraction method: FENED works with different machine learning models.
- A network traffic representation that captures periodic behaviors: Although other works capture similar behaviors, FENED is not restricted to similarity and periodicity measures since we can combine different machine learning models into our system.

III. The Proposed Method for Extraction of Spectral Features

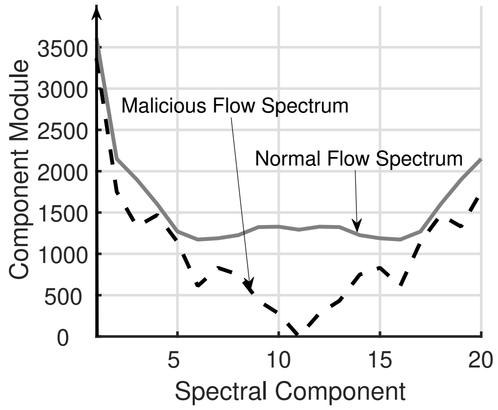

Figure 1. Comparison between the spectrum with 20 components extracted, chosen at random from samples of a) a normal flow, which resembles white noise, and b) a malicious flow, in which the frequency components have a high standard deviation.

We propose a feature extraction method that returns a vector containing each component module of the discrete Fourier transform for a packet flow. The basic idea is to detect malicious flows by analyzing the difference in spectral behavior between the packet size of a normal flow and that of a malicious flow. Regular-communication flows have a spectrum similar to that of white noise [11], with components close to the average, as shows the solid grey line in Figure 1. Differently, attack flows automatically generated by a botnet present some spectral components that are far from the window average, as shows the dash black line in Figure 1. Thus, from the extracted feature vector, it is possible to obtain information on periodic and automated behaviors, common patterns of botnets. We assume that special network traffic as telemetry sent by network equipment, interactions between front-ends and databases, and software update requests are isolated from our system by an IP filter to avoid triggering the system with false positives of normal periodic traffic. The steps for constructing the feature vector are detailed next.

A. Feature Extraction with FENED

Figure 2 shows the steps FENED uses to extract spectral features from a flow. Initially, we capture network traffic and separate it into flows[6]. For each flow a vector is allocated that contains a list of packet sizes, in bytes, in chronological order. Packets sent by the origin and received by the destination are stored as positive values, while packets sent on the other direction receive negative sign of magnitude. The magnitude sign produces a discrete oscillating signal [13].

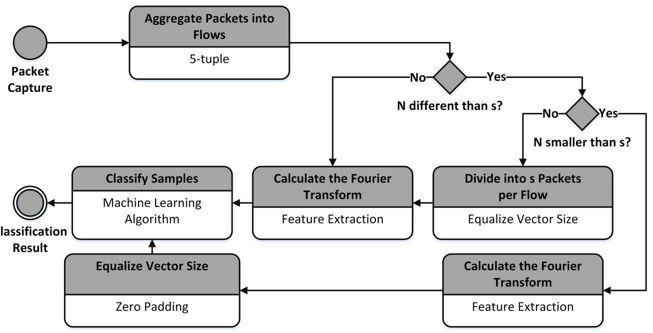

Figure 2. Proposed execution diagram for extracting features and classifying network flows with $\boldsymbol{N}$ packets. The captured packets are identified and separated into flows using the 5-tuple TCP/IP and the modules of the spectrum that form the feature vector are calculated. The feature vector is resized to size $\boldsymbol{s}$.

Traditional machine learning algorithms, such as decision tree and support vector machine, process and classify samples containing structured feature vectors. Thus, a restriction for the sample classification is that the feature vector must have the same dimension in the entire dataset. The vector dimension resulting from the Fourier transform, however, is equal to the number of packets, N, used in the calculation. Therefore, it is necessary to adjust the vector size to guarantee the same dimension after feature extraction. We define the parameter s, which represents the window size, and consequently, the dimension of the feature vector used by the classifier. FENED processes a network capture file continuously, without sizing the time vector a priori. Thus, the extraction generates $\lceil \frac{N}{s} \rceil$ feature vectors for a flow with N packets in a trace file. FENED can, however, classify packet streams by slightly modifying the application code, which we do not implement yet. If the flow has a larger number of packets, a selection policy is applied to reduce its size to s. Nonetheless, if the flow contains less than s packets, the creation of the feature vector must introduce padding elements until the vector reaches the correct dimension.

The strategy to reduce the number of spectral components extracted from the flow is to use windows with s packets to calculate the FFT. The advantage of using this strategy is to enable agile extraction of features since the feature vector calculation is limited in the packet amount. Thus, the packet removal before calculating the FFT reduces the computational cost, speeds up extracting features, and decreases the final proposal execution time.

[6]We define flows as a set of packets that contains the same source IP, TCP/UDP source port, destination IP, TCP/UDP destination port, and transport layer protocol.

The strategy for resizing the vectors when the number of N packets is less than s is to calculate the original vector FFT and padding with zeros to the result. This strategy keeps the flow spectrum unchanged since we only add null elements to the initial spectrum. Moreover, the insertion of elements in the vector has a low impact on the execution time.

IV. FENED Development and Evaluation of a Prototype

We implement the proposed feature extraction method in Python v3.7.7 and test on models created with the scikit-learn v1.0.1 [20][7]. We perform the experiments on an Intel Xeon E5-2650 CPU 2.00 GHz server with 16 cores and 252 GB of RAM.

The experiments use the CTU-13 [21] database, which contains 13 datasets with samples of normal and malicious traffic generated from machines infected by botnets. The datasets allows evaluating the periodic behavior of multiple botnets in different scenarios. We show the results of the experiments with the average value of the classification metrics with a 95% confidence interval.

In TCP connections, flowtbag monitors the state of the connection to define the beginning, through the SYN flag, and the end of a flow, through a FIN or RST flag, returning the flow feature vector. This flow definition is not appropriate for the vector construction to the FFT calculation because it has a variable number of packets in a flow. For the calculation of the FFT, the best would be a vector with a fixed dimension, corresponding to a number s of packets, which contains the size of each packet. A SYN flag for TCP connections starts a flow and lasts for s packets. We refer to this flow as an s_{pkt}-flow. Thus, a s_{pkt}-flow with N packets in FENED corresponds to $\lceil \frac{N}{s} \rceil$ packet windows of s size vector per window. The two feature extraction methods keep the distribution balance between normal and malicious flows (or s_{pkt}-flows), with a difference of less than 5% with respect to the original distribution.

We analyze FENED through four experiments: (i) investigation of the dimension of the feature vector that provides the highest recall performance and the shortest processing time; (ii) determination of the classification performance of three different classifiers with our new representation proposal; (iii) evaluation of the classification performance using flowtbag flows and s_{pkt}-flows; and (iv) comparison of the time to extract features of the two methods. Experiment (i) obtains the best size of the s window for the extracted feature vector, while Experiment (ii) determines the classifier that gives the best execution-time/performance classification trade-off. Finally, Experiments (iii) and (iv) compare FENED with the conventional feature extraction method. We choose the recall metric because it measures the classifier's ability to detect attacks.

Experiment (i) analyzes the classification performance as a function of the feature vector dimension. The experiment

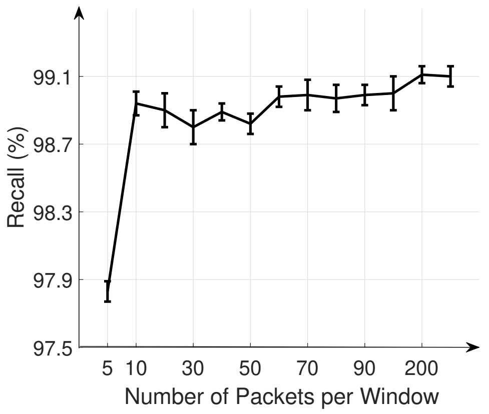

Figure 3. Recall metric of the decision tree as a function of the number of packets used to extract the feature vector.

defines the vector of spectral features with the smallest dimension that produces the best classification sensitivity to reduce computational cost without harming the classification of malicious flows. Figure 3 illustrates the recall result when the dimension of the feature vector changes. The experiment shows that the performance of malicious flow detection as evaluated by the recall metric does not change significantly after 10 packets per window. Thus, fixing the window size, s, to 10 reduces the computational time while keeping a high attack detection rate.

Experiment (ii) determines the classifier with the best execution-time/performance classification trade-off using the recall metric. We choose three popular algorithms for our evaluation [6]. The first one is the decision tree, which has low complexity as compared to other algorithms [22]. We also evaluate the random forest because, for many problems, it has better classification performance than decision trees. Finally, we test the Multilayer Perceptron (MLP), which has the potential to create a fine-grained decision boundary and achieve better classification performance. Nevertheless, adopting the random forest or MLP implies higher training and classification time than decision trees in line with the higher computational complexity. Figure 4 shows the results of experiment (ii).

We observe that the decision tree and random forest outperform MLP. Nevertheless, the tree-based algorithms have the same statistical performance even if random forest presents a slightly higher mean value. We can conclude that our feature representation proposal allows us to adopt a lower complexity algorithm, the decision tree, while still achieving good classification performance.

Experiment (iii) evaluates FENED performance for different datasets and compares the proposal with conventional flowtbag alternative. Table I shows the results for the experiment for the two methods of extracting features evaluated for the decision tree algorithm, using a random division of 70% of the data for training and 30% for test. We also evaluate

[7]Available at https://scikit-learn.org/stable/.

Table I
CLASSIFICATION PERFORMANCE RESULTS OBTAINED WITH THE TWO METHODS OF EXTRACTING FEATURES FOR DIFFERENT PERFORMANCE METRICS, EVALUATED FOR DIFFERENT SCENARIOS OF THE CTU-13 DATABASE. THE PROPOSED SYSTEM PRESENTS BETTER RESULTS THAN FLOWTBAG IN SEVERAL SCENARIOS IN THE METRICS HIGHLIGHTED IN BOLD.

	Accuracy (%)[‡]		Precision (%)[‡]		Recall (%)[‡]		F1-Score (%)[‡]	
Scenario	flowtbag	FENED	flowtbag	FENED	flowtbag	FENED	flowtbag	FENED
1	94.5 ± 0.1	**99.01 ± 0.03**	85.4 ± 0.6	**96.1 ± 0.2**	76.5 ± 0.6	**97.7 ± 0.1**	80.7 ± 0.5	**96.9 ± 0.1**
2	99.71 ± 0.07	98.76 ± 0.05	99.7 ± 0.1	98.01 ± 0.08	99.6 ± 0.2	98.7 ± 0.1	99.64 ± 0.09	98.34 ± 0.06
3	99.99 ± 0.01	98.63 ± 0.03	70 ± 30	**99.23 ± 0.06**	70 ± 20	**92.7 ± 0.2**	70 ± 20	**95.88 ± 0.09**
4	99.91 ± 0.03	99.12 ± 0.02	88 ± 5	**99.75 ± 0.04**	90 ± 3	88.0 ± 0.2	89 ± 4	**93.5 ± 0.2**
5	99.2 ± 0.2	97.45 ± 0.07	95 ± 1	89.4 ± 0.3	96 ± 2	95.5 ± 0.3	95.4 ± 0.8	92.3 ± 0.2
6	99.82 ± 0.05	99.70 ± 0.01	97 ± 1	97.4 ± 0.4	98 ± 1	97.8 ± 0.5	97.3 ± 0.7	97.6 ± 0.2
7	99.2 ± 0.2	97.8 ± 0.2	88.3 ± 0.4	**97.6 ± 0.4**	82.8 ± 0.6	**95.6 ± 0.6**	84.8 ± 0.3	**96.6 ± 0.3**
8	99.6 ± 0.2	99.54 ± 0.02	98 ± 1	97.6 ± 0.2	96 ± 1	**99.0 ± 0.2**	97 ± 1	**98.31 ± 0.07**
9	95.56 ± 0.09	**98.58 ± 0.02**	97.30 ± 0.07	**97.50 ± 0.03**	95.4 ± 0.2	**99.45 ± 0.01**	96.34 ± 0.08	**98.46 ± 0.02**
10	99.89 ± 0.03	99.43 ± 0.01	87.9 ± 0.5	**99.21 ± 0.01**	97.0 ± 0.2	**99.95 ± 0.03**	92 ± 0.3	**99.579 ± 0.007**
11	99.4 ± 0.6	88.6 ± 0.5	40 ± 20	**70 ± 10**	70 ± 30	60 ± 20	50 ± 20	50 ± 10
12	99.1 ± 0.2	97.97 ± 0.07	97.5 ± 0.5	96.4 ± 0.3	97 ± 1	97.1 ± 0.3	97.0 ± 0.8	96.7 ± 0.1
13	97.8 ± 0.1	**99.35 ± 0.02**	95.8 ± 0.4	**98.76 ± 0.06**	94.4 ± 0.3	**98.94 ± 0.07**	95.1 ± 0.2	**98.85 ± 0.04**

[‡] We adjust the decimal places according to the obtained confidence interval.

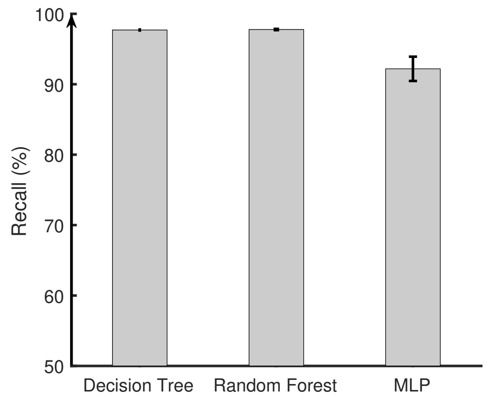

Figure 4. Recall metric of three classifiers using the new feature representation approach. The decision tree has better classification performance than Multilayer Perceptron (MLP) and a similar performance to the random forest. Nevertheless, the decision tree has lower complexity, justifying its adoption among the other two classifiers.

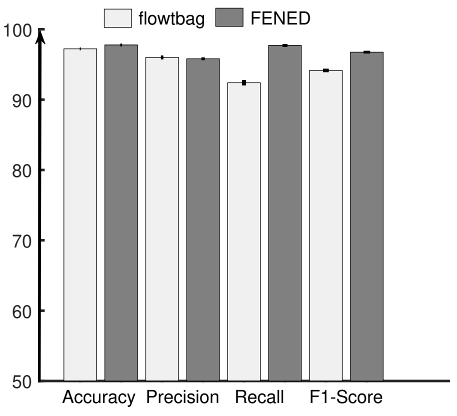

Figure 5. Flowtbag and FENED classification performance. The white bar represents the results of flowtbag while the grey represents FENED results. The evaluation combines all scenarios of the CTU-13 database.

FENED and flowtbag in a scenario that combines all the 13 datasets. Figure 5 shows the results. The dimension of the feature vector extracted with FENED is equal to 10, the size of the feature vector obtained in Experiment (i) that presents the best cost-benefit. FENED detects 6% more automatic attacks of DDoS carried out by botnets than flowtbag, present in scenarios 4 and 10. The frequency domain representation of the packets generated by the DDoS facilitates attack detection. The port scan present in scenarios 6, 8, 9, and 13 is also easily detected through spectral decomposition, as its behavior is automated, generating periodic components with high energy. Spectral decomposition also increased the detection of SPAM attacks and click fraud in scenarios 1, 9, and 13.

Finally, Experiment (iv) compares the latency of FENED with flowtbag application for the feature extraction and confirms the computational linear time of the proposal. The comparison is simulated in MATLAB to mitigate the differences caused by implementations using different programming languages. FENED achieves similar or better classification results than flowtbag using significantly fewer features. We argue that this behavior is observed because FENED explores the temporal correlation between packets in a flow while flowtbag only considers conventional counter features.

The time required to extract the features of a single flow with FENED is approximately 100 μs, while for flowtbag, it is approximately 10 μs. Figure 6, however, proves that the time varies linearly for cases in which FENED proposal and flowtbag application extract features from 100 and 10,000 flows. Therefore, the proposal has linear time execution with the number of flows for the case in which all flows have the same number of packets.

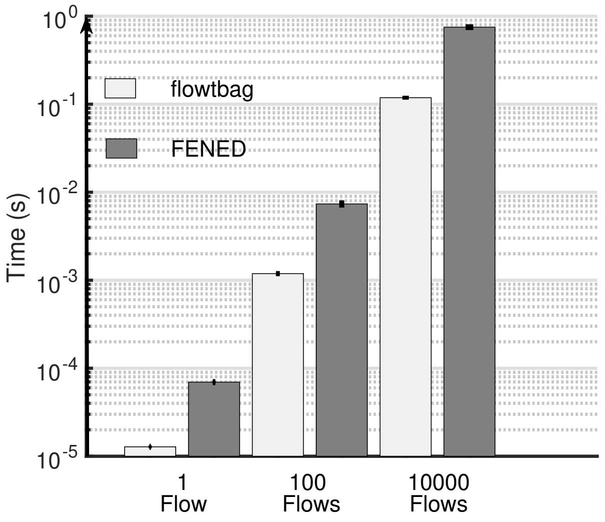

Figure 6. Comparison between the methods evaluated to extract flow features. The white bar represents the results of flowtbag while the grey represents FENED results. Both have linear execution time for the case where all flows have the same number of packets.

V. Conclusion

We propose a method that uses signal processing techniques to detect network threats produced automatically by botnets. The proposal uses the fast Fourier transform to generate feature vectors that expose the temporal dependence between the packets belonging to a flow. The visualization of the resulting spectrum demonstrates that normal flows have frequency representations equivalent to white noise, while botnet attacks deviate from this pattern. This flow representation allows the correct identification of automated and periodic attacks, such as DDoS, port scan, and click fraud. The feature extraction method is agnostic to the machine learning algorithm used, as it produces vectors of equal and previously defined dimensions. Also, the proposal preserves privacy because it only uses the size of the packets, not revealing the contents of the packet and sensitive information from users. Finally, the results show that fixing the number of packets inspected per window does not significantly change the classification metrics. The better classification performance than flowtbag justifies the higher execution time overhead that our feature extraction proposal introduces. In the future, we will extend the proposal to other network datasets.

VI. Acknowledgment

This work was financed by CNPq; by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES), Finance Code 001; FAPERJ Grants E-26/203.211/2017, E-26/211.144/2019, E-26/202.689/2018; and Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Grants 2015/24494-8, 2018/23292-0, 2015/24485-9, and 2014/50937-1.

References

[1] G. Sagirlar, B. Carminati, and E. Ferrari, "AutoBotCatcher: Blockchain-Based P2P Botnet Detection for the Internet of Things," in *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, 2018, pp. 1–8.

[2] V. Bezerra, V. T. da Costa, R. Martins, S. Barbon, R. Miani, and B. B. Zarpelão, "Providing IoT host-based datasets for intrusion detection research," in *Brazilian Symposium on Information Security and Computer Systems (SBSeg)*, 10 2018, pp. 15–28.

[3] A. G. P. Lobato, M. A. Lopez, I. J. Sanz, A. A. Cardenas, O. C. M. B. Duarte, and G. Pujolle, "An adaptive real-time architecture for zero-day threat detection," in *IEEE ICC*, May 2018, pp. 1–6.

[4] I. Sanz and O. Duarte, "Graph-based feature enrichment for online intrusion detection in virtual networks," in *Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)*, 2019, pp. 129–136.

[5] W. Liu, X. Liu, X. Di, and H. Qi, "A novel network intrusion detection algorithm based on Fast Fourier Transformation," in *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, 2019, pp. 1–6.

[6] L. C. B. Guimarães *et al.*, "TeMIA-NT: ThrEat Monitoring and Intelligent data Analytics of Network Traffic," in *Conference on Cloud and Internet of Things (CIoT)*, 2020, pp. 9–16.

[7] M. Pelloso, A. Vergutz, A. Santos, and M. Nogueira, "A self-adaptable system for DDoS attack prediction based on the metastability theory," in *IEEE GLOBECOM*, 2018, pp. 1–6.

[8] G. Bottazzi *et al.*, "Frequency domain analysis of large-scale proxy logs for botnet traffic detection," in *International Conference on Security of Information and Networks (SINCONF)*, 2016, pp. 76–80.

[9] S. D. Anton, L. Ahrens, D. Fraunholz, and H. D. Schotten, "Time is of the Essence: Machine Learning-based Intrusion Detection in Industrial Time Series Data," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018, pp. 1–6.

[10] M. Aiello *et al.*, "An on-line intrusion detection approach to identify low-rate DoS attacks," in *2014 International Carnahan Conference on Security Technology (ICCST)*, 2014, pp. 1–6.

[11] Y. Chen and K. Hwang, "Spectral Analysis of TCP Flows for Defense against Reduction-of-Quality Attacks," in *International Conference on Communications (ICC)*, 2007, pp. 1203–1210.

[12] M. Zhou and S.-D. Lang, "A frequency-based approach to intrusion detection," in *Proc. of the Workshop on Network Security Threats and Countermeasures*, 2003.

[13] E. Chimedtseren *et al.*, "Intrusion detection system using Discrete Fourier Transform," in *Seventh Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2014, pp. 1–5.

[14] B. AsSadhan and J. M. Moura, "An efficient method to detect periodic behavior in botnet traffic by analyzing control plane traffic," *Journal of advanced research*, vol. 5, no. 4, pp. 435–448, 2014.

[15] A. M. Manasrah, W. B. Domi, and N. N. Suppiah, "Botnet detection based on DNS traffic similarity," *International Journal of Advanced Intelligence Paradigms*, vol. 15, no. 4, pp. 357–387, 2020.

[16] J. Kwon *et al.*, "PsyBoG: Power spectral density analysis for detecting botnet groups," in *2014 9th International Conference on Malicious and Unwanted Software: The Americas (MALWARE)*, 2014, pp. 85–92.

[17] X. Yu *et al.*, "Online botnet detection by continuous similarity monitoring," in *2009 International Symposium on Information Engineering and Electronic Commerce*, 2009, pp. 145–149.

[18] R. F. Fouladi, O. Ermiş, and E. Anarim, "Anomaly-Based DDoS Attack Detection by Using Sparse Coding and Frequency Domain," in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–6.

[19] B. A. Powell, "Malicious Overtones: Hunting data theft in the frequency domain with one-class learning," *Transactions on Privacy and Security (TOPS)*, vol. 22, no. 4, pp. 1–34, 2019.

[20] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers & Security*, vol. 45, pp. 100–123, 2014.

[22] L. A. C. de Souza, G. Antonio F. Rebello, G. F. Camilo, L. C. B. Guimarães, and O. C. M. B. Duarte, "DFedForest: Decentralized Federated Forest," in *IEEE Blockchain*, 2020, pp. 90–97.