Intrusion Detection System based on Network Traffic using Deep Neural Networks

Dimitra Chamou, Petros Toupas, Eleni Ketzaki,
Stavros Papadopoulos, Konstantinos M. Giannoutakis,
Anastasios Drosou, Dimitrios Tzovaras
Information Technologies Institute (ITI)
Center for Research & Technology Hellas (CERTH)
Thessaloniki, Greece
{dimicham,ptoupas,eketzaki,spap,kgiannou,drosou,Dimitrios.Tzovaras}@iti.gr

***Abstract*—Nowadays, the security of small-medium enterprises against cyber-attacks is a matter of great importance and challenging area, as they are financially and functionally affected. Novel and sophisticated attacks are emerging daily, targeting and threatening a large number of businesses in the world. For this reason, the implementation and optimization of the performance of Intrusion Detection Systems have attracted the interest of the scientific community. In this paper, a machine learning solution based on a deep neural network is proposed, in order to detect malicious behavior (DDoS and Malware cyber-threats) in the network traffic of an SME in real-time. The experimental results for the intrusion detection system showed that the proposed model can achieve high accuracy, having at the same time low false positive rate, while distinguishng between malicious and normal network traffic.**

***Index Terms*—Cybersecurity, Intrusion Detection System, Deep Neural Networks, DDoS Detection, Malware Detection**

I. Introduction

The attainment of cybersecurity and the protection of data and communication of the users are considered essential because of the rapid increment in Internet applications and their use by most of the world's population. At the same time, increased exposure to more sophisticated cyber-threats has been observed over the Internet and computer networks, in academic and industry digital world, especially in Small-Medium Enterprises [1], with financial and emotional costs. For this reason, the study and the ongoing development of cybersecurity intrusion detection technology are particularly timely and necessary, as first line of defense, to prevent, tackle intrusion threats and reveal new security issues.

Intrusion Detection Systems (IDS) provide an efficient security mechanism for the detection and protection of cyber-attacks in any network or in small-medium enterprises. Network Intrusion detection systems have a primary role in the reinforcement of a system and defend the network against ever-increasing malicious behaviors by monitoring and analyzing the network traffic in real time. The main purpose of IDS is to detect effectively cyber-attacks, viz. suspicious or abnormal behavior, violating system security and exposing other network users to danger, and to take the necessary measures to protect the system. Network traffic analysis uses network flows to monitor the network of the system and divides traffic packets into network flows [2], [3].

Researchers' attention focuses on implementing IDS to achieve effective solutions against intruders and attacks [4]. So far, in the research field of IDSs, among other proposed taxonomies [5], detection techniques are categorized as either signature-based [6] , anomaly-based [7] , or a hybrid combination of both. Compared to signature-based, anomaly-based shows a significant difference in identifying novel attacks (zero-day-attack) [4], thereat our proposed model rely on anomaly-based detection.

Machine learning and deep learning in the cybersecurity industry is constantly expanding and is a growing field of research with innovative technologies. In order to increase effectiveness of IDSs, research has been focused on novel technologies of machine learning, artificial intelligent etc. Machine learning techniques designed to detect DDoS and Malware attacks are based on certain knowledge provided for training. Soft computing techniques are based on fuzzy logic or neural networks to enhance the performance of a DDoS detection system, especially in case the information about the attack traffic is insufficient and knowledge-based approach based on the history of previous DDoS attacks to develop a defense solution [8], [9], [10], [11], [12]. Beyond the above techniques, [13] proposed also traffic pattern analysis, connection rate techniques, SNORT and Open Flow integrated techniques. The widespread use of machine learning extends to the fields of prediction, classification and estimation, especially in the field of cybersecurity.

As we can see, deep learning is a branch of machine learning and can be deployed for supervised, unsupervised and semi-supervised learning. Deep learning, depending on the applied field of classification, achieves better accuracy than machine learning methods. The most known deep learning technologies are Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM), Multi-Layer Perceptron (MLP). In this paper, we will implement an Intrusion Detection System using Multi-Layer Perceptron.

The purpose of this paper is to implement an anomaly based intrusion detection system using flow-based statistical data. The proposed solution detects and classifies with high accuracy between DDoS and Malware types of attacks and normal traffic separately in binary classification for each type of attack. Classification of flows achieved with the use of deep learning algorithms. This paper proposes two deep neural network with multiple fully connected layers (Multi-Layer Perceptron) in order to classify the network traffic of a SME into normal or malicious for DDoS and malware threats. The propose solution was implemented under the umbrella of FORTIKA H2020 project which is an ongoing EU-funded project for cyber-security [14].

The structure of this paper is as follows. Section II describes the related work regarding the intrusion detection systems using machine learning techniques, for malware and DDoS detection. Section III explains the DDoS Detection architecture that have been followed. In section IV, a brief description of the architecture of the Malware Detection system is provided. Section V, analyzes the implementation and training processes of the anomaly detection. Section VI provides the experimental results, and finally, Section VII concludes the paper and presents future work.

II. Related Work

The need to explore new IDS techniques and improve the performance of existing against novel cyber threats, such as DDoS and Malware, has drawn the attention of researchers in order to address security issues and fulfill vulnerability gaps.

Jiang, et al [15] presented a hybrid detection model to detect four varieties of ALDDoS attacks. Since they have studied system' s behavior and features of ALDDoS attacks, they combined traffic and user behavior features. They aimed to improve the accuracy of detection and reduce the time and complexity of training user behavior model, using time windows. The experiments conducted in CICDS2017 dataset for DDoS (ALDDoS) detection attacks, by applying three layer Back Propagation Neural Network (BPNN) for the classification. They compared their model with traffic-based and hybrid KNN model and achieved to succeed DDoS attacks detection with accuracy, precision, recall and F1 rate of 99%.

Aksu, et al [16] conducted a comparative study using 3 machine learning algorithms. They used CICIDS2017 dataset and the supervised machine learning algorithms Support Vector Machine (SVM), K Nearest Neighbor (KNN) and Decision Tree (DT) for binary classification between DDoS and benign examples. They used Fisher Score algorithm in order to reduce the dimension of the dataset and select the most appropriate features, from 80 features to 30, and non-related features were eliminated. The exported performance measurements, like accuracy, recall, precision and F-score metrics, shown that for the Decision Tree method, evaluation scores did not change, in contrast with KNN' s accuracy was increased and SVM' s was decreased.

Zhou and Pezaros [5] conducted an analysis using common supervised machine learning algorithms for anomaly flow-based detection. They implemented different algorithms, such as Random forest classifier, Gaussian naive bayes classifier, Decision tree classier, Multi-layer Perceptron (MLP) classifier, K-nearest neighbours classifier, Quadratic discriminant analysis classifier to the CIC-AWS-2018 Dataset, comparing their performance using the performance criterias of precision, recall, F1 score, and time expense.

Roopak, Tian and Chambers [17] carried out a comparative analysis between different deep-learning algorithms and mechanical learning algorithms. They have implemented four different classification deep learning models as Multilayer Perceptron (MLP), CNN, LSTM, CNN+LSTM and they compared them with machine learning algorithms, such as bayes and random forest. The evaluation was done on balanced CICIDS2017 dataset for detection of DDoS attack with binary classification. Based on the performance metric, accuracy, precision and recall, the model with the highest accuracy of 97.16% was CNN+LSTM for both deep and machine learning algorithms on the dataset used.

Faruki et al. [18] proposed a method that is effective against code obfuscation and repackaging that are widely used techniques to evade anti-virus signature and to propagate unseen variants of known malware, their results demonstrate robust detection of variants of known malware families. Their method creates variable length signature and compares it with a signature database using fuzzy logic techniques, their main goal was to detect unseen and zero-day samples of known malwares.

Huang et al. [19], attempt to explore the possibility of detecting malicious applications in the Android operating system. They analyze the required and requested permission for an Android application using machine learning algorithms on three data sets. Four commonly machine learning algorithms including AdaBoost, Naive Bayes, Decision Tree, and Support Vector Machine, are used to evaluate the above performance. Their experimental results detect more than 81% of malicious samples.

Shabtai et al. [20], proposed Andromly, a behavior-based Android malware detection system. They classify the application as normal or malware based on continuously monitoring specific features and patterns that indicate the device state such as battery level, CPU consumption etc. while it is running and then apply different machine learning algorithms to discriminate between malicious and benign applications.

Sanz et al. [21], present PUMA, a method for detecting malicious Android applications through machine learning techniques by analyzing the extracted permissions from the application itself. Their methods can be used as a first step before other more extensive analysis, such as a dynamic analysis.

Our contribution is the implementation of a novel anomaly detection solution, achieving promising results in binary classification between normal and malicious network traffic, by using a deep learning model with multiple fully connected layers (Multi-Layer Perceptron). A more detailed explanation of our proposed implementation is given in Section V.

III. DDoS Detection

DDoS input data depend on the choice of the detection method, but beyond the specification of the method, it is essential to provide features that describe information regarding the network traffic and features that will provide information for the detection process. The features that provide information for the network traffic are the Source IP address, the destination IP address and the traffic volume that can be measured directly from the network and the resource entropy that can be calculated and provide the information rate of the traffic. For the DDoS process the CPU Usage, memory usage, Latency, Packet loss, Link Utilization and the Throughput are the same features that demanded as input data.

The detection methods that concern a defense system for DDoS attacks should have the following characteristics:

- Real-time Performance. The system should be able to detect an ongoing attack before the attack overwhelm the victim with malicious traffic.
- Scalability. The attack rates of todays DDoS attacks are in the order of hundreds of Gbps, thus the scalability of the defense system plays an important role in the detection mechanism.
- Maintaining quality of service. Special mechanisms are needed in order to separate normal traffic from attack traffic with high confidence, so that the quality of service to legitimate users can be maintained.
- Source Identification. A DDoS attack defense system should be robust against IP spoofing. It should have a suitable mechanism such as traceback or pushback to locate the attack source.

Figure 1 illustrates the architecture of the proposed approach for Distributed Denial of Service (DDoS) detection. The proposed model is comprised of three modules, namely 1) Raw Data extraction, 2) Feature Vector Representation, and 3) Classification using Deep Neural Networks.

The Raw Data extraction module monitors the activity of the system and stores to a database the network traffic exchanged between the system and other systems, or within modules of the same system. Specifically, the Raw Data extraction module monitors packets, their size, and the time in which they were sent/received. The data stored in the database are used as input by the Feature Vector Representation module. Particularly, the data are aggregated in specific time windows, and the total number of packets and bytes exchanged within each window is measured. The size of the window is a parameter of the model. Given the aggregated information and the window size, seven features are extracted for each time window, namely, i) Number of Packets, ii) Average packet size, iii) Time Interval Variance, iv) Packet Size Variance, v) Number of Bytes, vi) Packet Rate, and vii) Bit Rate. Finally, the produced feature vectors are given as input to a Deep Neural Network for classification. The proposed network is comprised of a total of 5 layers, including fully connected layers, batch normalizations, and non-linear activations (Relu). The output of this network is a binary classification, representing either the existence of a DDoS attack, or normal traffic. The Deep Neural Network is trained on instances of both DDoS attacks and normal traffic in order to learn to differentiate between the two.

IV. Malware Detection

An extended work regarding malware detection for operational systems has been presented during the past few years and was an inspiriting motivation for the proposed methodology that has been developed.

The requirements process needed for malware defense of an organization require multiple layers of defenses [22]. The defense should consider both the enclave boundary area and the computing environment area:

- The enclave boundary concerns the interaction of the organization network with the Internet.
- The computing environment concerns the inner network of the organization.

More specifically, the enclave boundary defense should focus mainly on firewalls and intrusion detection systems.

1) The firewalls need to be strictly controlled and limiting inbound and outbound communications.
    - Simple Mail Transfer Protocol (SMTP) to any address from only the SMTP mail gateways.
    - Domain Name Server (DNS) to any address from an internal DNS server to resolve external host names.
    - HTTP and HTTPS from an internal proxy server for users to browse web sites.
    - Network Time Protocol (NTP) to specific timeserver addresses from an internal timeserver.
    - Network ports required by AV, spam filtering, web filtering or patch management software to only the appropriate vendor addresses.
2) The Intrusion Detection System (IDS) aims to identify network traffic in near real time. Most IDSs use signatures to detect port scans, malware, and other abnormal network communications. The ideal placement of an IDS is external to the organization as well as internally, just behind the firewall.

Figure 2 shows the architecture of the proposed approach for malware detection. It is comprised of three components, 1) Raw Data extraction, 2) Feature Vector Representation, and 3) Classification using Deep Neural Networks.

The Raw Data extraction component monitors the system calls for each application on the monitored system. It has a specific list of system calls to monitor, related to file system calls, process calls, network calls, and memory calls. The output of this component is a sequence of system calls for each application running in the system. These sequences of calls are given as input to the second component, namely, the Feature Vector Representation. For each application call sequence, N-grams of consecutive system calls are considered, for N=4. Each unique N-gram is encoded in a unique position in the feature vector. The size of the final feature vector is equal to the number of unique N-gram for all the applications.

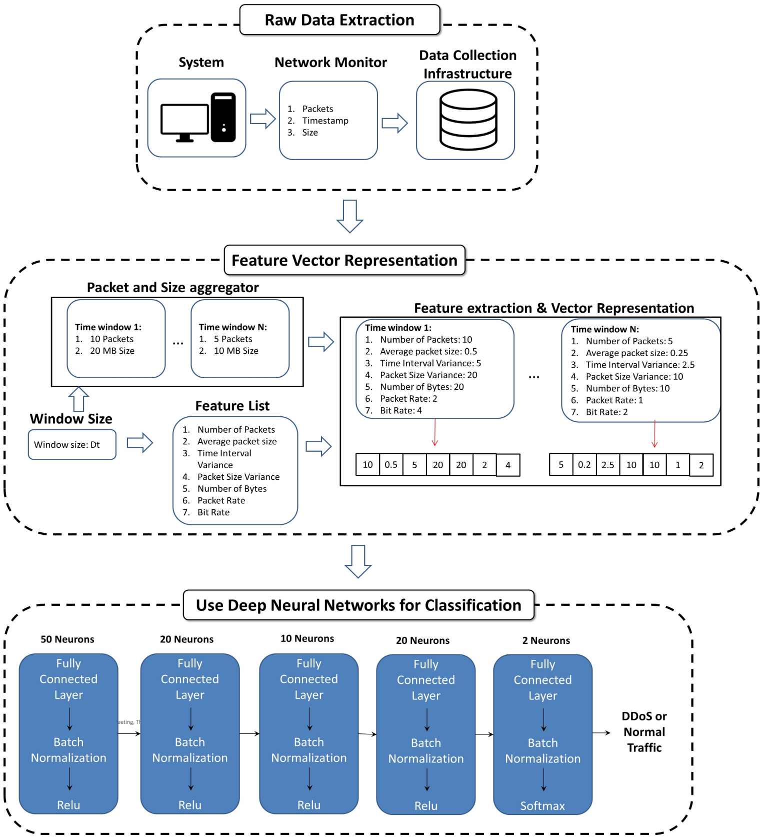


Fig. 1: DDoS architecture

The value of each position of the vector is equal to the number of appearances of the corresponding N-gram. Finally, these feature vectors are given as input to a Deep Neural Network for classification. The proposed network is comprised of a total of 5 layers, including fully connected layers, batch normalizations, and non-linear activations (Relu). The output of this network is a binary classification, representing either malware, or benign application. The Deep Neural Network is trained on instances of both malwares and benign applications in order to learn to differentiate between the two.

V. Implementation

To be able to train a DNN it is important to acquire a large volume of data. In this first approach, the data has been gathered from various sources on the web, where there are reported many cases of malicious network traffic for DDoS and Malware attacks, as long as for normal network traffic. As a first step, a tool has been developed in order to convert the network traffic captured in pcap files (binaries), into human readable json files (netflow v9 relevant format) containing important information about the flows in the network. This tool is based on tshark commands, which have been used to “filter” the information included in pcap files, and so keep only the features considered important in order to extract the DNN input feature vector.

The next step of the feature extraction process is to parse the json files produced by the previous task, and create the feature vectors for the DNN input. The range of values for each one of the 7 features is different and the deviations vary from very small to very big intervals. It was necessary to scale the range of all features in a, smaller, common for all features, range. In order to achieve that, the normalization technique has been used. Using the normalization of Python scikit learn library, the final range of all features varies between 0 and 1.

Before defining the DNN architecture, the dataset was split into three individual datasets: training, validation, and testing set. The DNN was trained using only the training set. The validation set was used in order to fine-tune the hyperparameters of the neural netowrk, like learning rate, dropout, choose between multiple optimizers, find the best value for number of layers and number of nodes in each layer, etc. Moreover, the test set does not interfere at any point with the training process, so the evaluation of the DNN can be performed using this set and considering the evaluation result to be quite safe. We have also evaluated our models witha 10 fold cross validation method. The proposed DNN architecture, consists of the input layer, followed by a first hidden layer, a second hidden layer, a third hidden layer, a fourth hidden layer, and the output layer . In the output of the four hidden layers “ReLU” activation function has been applied, while “Sigmoid” activation function has been applied on the output layer. Also, for the weights initialization in each one of the layers, “lecun uniform” initializer was used.

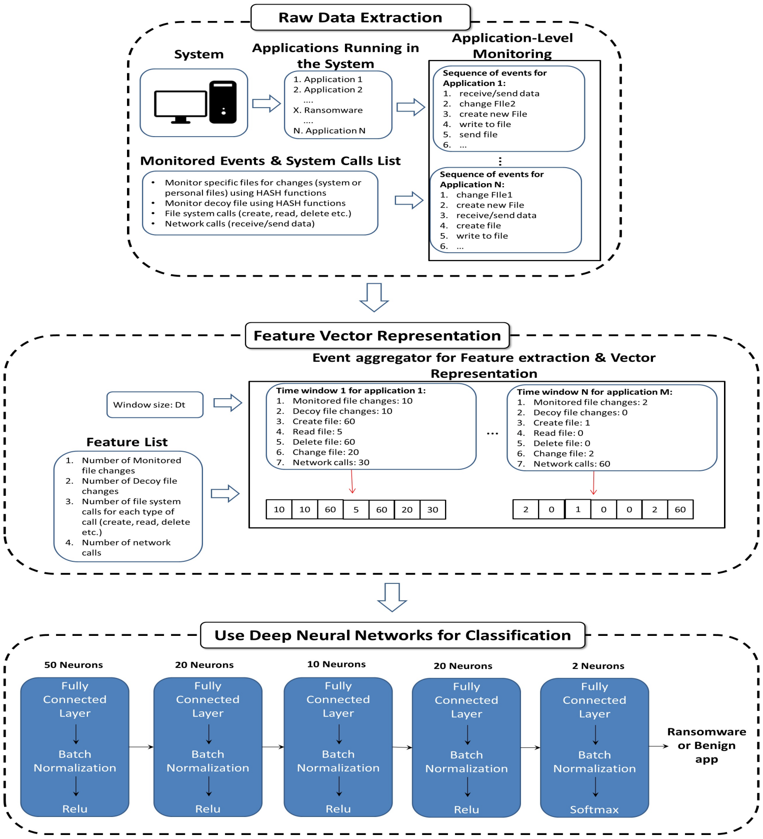

Raw Data Extraction
System
Applications Running in the System
1. Application 1
2. Application 2
....
X. Ransomware
....
N. Application N
Application-Level Monitoring
Sequence of events for Application 1:
1. receive/send data
2. change FIle2
3. create new File
4. write to file
5. send file
6. ...
Sequence of events for Application N:
1. change FIle1
2. create new File
3. receive/send data
4. create file
5. write to file
6. ...
Monitored Events & System Calls List
Monitor specific files for changes (system or personal files) using HASH functions
Monitor decoy file using HASH functions
File system calls (create, read, delete etc.)
Network calls (receive/send data)
Feature Vector Representation
Event aggregator for Feature extraction & Vector Representation
Window size: Dt
Feature List
1. Number of Monitored file changes
2. Number of Decoy file changes
3. Number of file system calls for each type of call (create, read, delete etc.)
4. Number of network calls
Time window 1 for application 1:
1. Monitored file changes: 10
2. Decoy file changes: 10
3. Create file: 60
4. Read file: 5
5. Delete file: 60
6. Change file: 20
7. Network calls: 30
Time window N for application M:
1. Monitored file changes: 2
2. Decoy file changes: 0
3. Create file: 1
4. Read file: 0
5. Delete file: 0
6. Change file: 2
7. Network calls: 60
...
10 10 60 5 60 20 30
2 0 1 0 0 2 60
Use Deep Neural Networks for Classification
50 Neurons
20 Neurons
10 Neurons
20 Neurons
2 Neurons
Fully Connected Layer
Batch Normalization
Relu
Softmax
Ransomware or Benign app

Fig. 2: Malware Architecture

For the training of the DNN "adam" optimizer was used for the back-propagation process using the default values of the optimizer. The loss function used, was "binary crossentropy" and DNN has been training for 30 epochs. All of the above-mentioned functions and components, which were used, are part of the Python tensorflow library. As mentioned above a validation set has been used to keep tracking of the training process and its performance.

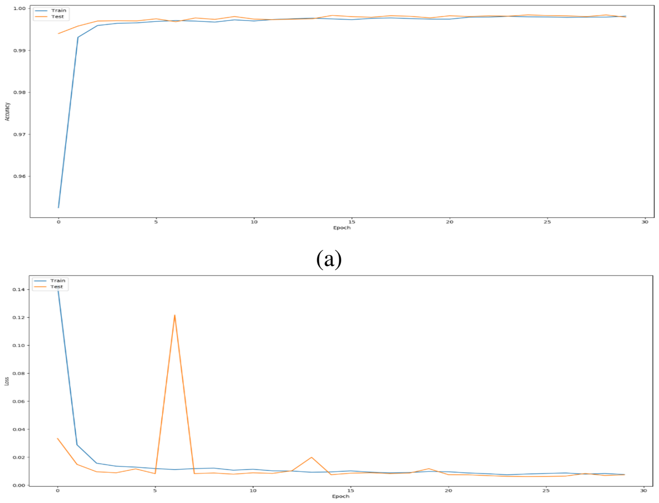
VI. Numerical Results

During the training process of DNN, a graph indicating training and validation accuracy, and one about training and validation loss over all epochs, have been created to gain a better intuition about the performance of the DNN. These graphs for DDoS and Malware are presented in Figure 3a, 3b, 4a and 4b respectively.

The evaluation of the DNN was based on the test set and it is shown in Table I.

TABLE I: Evaluation of DNN for DDos and malware

	DDoS	Malware
Test Set Accuracy	99.79%	99.44%
True Positives	151737	4030
False Positives	208	102
True Negatives	30091	30324
False Negatives	178	92

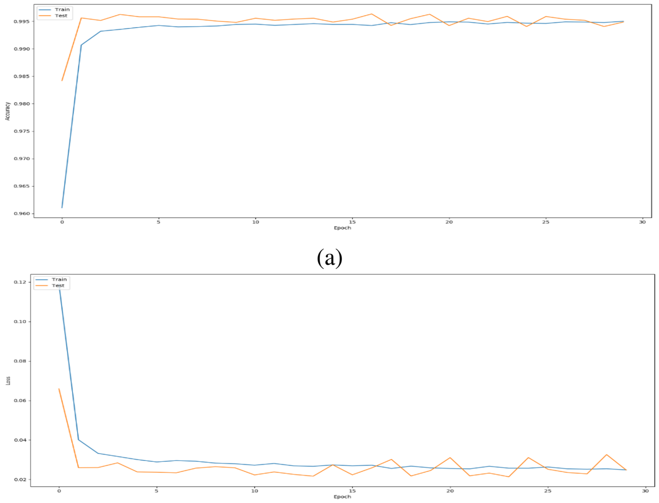


(a)

(b)

Fig. 3: (a) DDoS accuracy change during training for 30 epochs (b) DDoS loss change during training for 30 epochs

The prediction (feed-forward) of the DNN was also implemented and accelerated in FPGA achieving a speedup of $40x$ compared to the CPU implementation. This step was crucial since the prediction of a threat in a SME is taking place in real-time, so the faster the decision can be exported from the



(a)

(b)

Fig. 4: (a) Malware accuracy change during training for 30 epochs (b) Malware loss change during training for 30 epochs

DNN the better for the SME, because it can come up faster with a mitigation action regarding the threat.

VII. Conclusion and Future Work

In this work, an anomaly detection system combined with deep learning methods was presented. A specialized solution targeting the needs of SMEs to defend against cyber-attacks, such as DDoS and malware was introduced. The proposed solution used network flows in order to classify the cyber-threats into binary categories (normal or malicious) using multiple fully connected layers (Multi-Layer Perceptron). Classification achieved using flow-based statistical data. Thus, the implementation of anomaly detection system was achieved with high accuracy rate of 99.79% for DDoS detection and 99.44% for Malware detection.

In future, there is an intention to expand existing work and try novel deep learning algorithms in a complete, rich, up-to-date and well-formed dataset for training, validation and evaluation, in order to extract additional results. Furthermore, it is intended to use system logs data metrics, collected by system monitoring agents, for expanding the features of malware detection. Additionally, the problem will be approached by the aspect of multiclass classification in order to detect and classify more categories of cyber-threats.

Acknowledgment

This work has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation, with Title H2020-FORTIKA "cyber-security Accelerator for trusted SMEs IT Ecosystem" under grant agreement no 740690.

References

[1] C. Paulsen, "Cybersecuring small businesses," *Computer*, vol. 49, no. 8, pp. 92–97, Aug 2016.

[2] Y. Zhang, X. Chen, L. Jin, X. Wang, and D. Guo, "Network intrusion detection: Based on deep hierarchical network and original flow data," *IEEE Access*, vol. 7, pp. 37 004–37 016, 2019.

[3] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, "Internet traffic classification by aggregating correlated naive bayes predictions," *IEEE transactions on information forensics and security*, vol. 8, no. 1, pp. 5–15, 2012.

[4] A. Boukhamla and J. Coronel, "Cicids2017 dataset: Performance improvements and validation as a robust intrusion detection system testbed." *International Journal of Information and Computer Security*, 09 2018.

[5] Q. Zhou and D. Pezaros, "Evaluation of machine learning classifiers for zero-day intrusion detection - an analysis on CIC-AWS-2018 dataset," *CoRR*, vol. abs/1905.03685, 2019. [Online]. Available: http://arxiv.org/abs/1905.03685

[6] H. Holm, "Signature based intrusion detection for zero-day attacks: (not) a closed chapter?" in *2014 47th Hawaii International Conference on System Sciences*, Jan 2014, pp. 4895–4904.

[7] V. Jyothsna, V. V. Rama Prasad, and K. Munivara Prasad, "A review of anomaly based intrusion detection systems," *International Journal of Computer Applications*, vol. 28, pp. 26–35, 08 2011.

[8] R. Braga, E. Mota, and A. Passito, "Lightweight ddos flooding attack detection using nox/openflow," 10 2010, pp. 408–415.

[9] C. Chung, P. Khatkar, T. Xing, J. Lee, and D. Huang, "Nice: Network intrusion detection and countermeasure selection in virtual network systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 4, pp. 198–211, July 2013.

[10] N. Z. Bawany, J. A. Shamsi, and K. Salah, "Ddos attack detection and mitigation using sdn: Methods, practices, and solutions," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 425–441, Feb 2017. [Online]. Available: https://doi.org/10.1007/s13369-017-2414-5

[11] S. Dotcenko, A. Vladyko, and I. Letenko, "A fuzzy logic-based information security management for software-defined networks," in *16th International Conference on Advanced Communication Technology*, Feb 2014, pp. 167–171.

[12] J. Wang, R. C. . Phan, J. N. Whitley, and D. J. Parish, "Augmented attack tree modeling of distributed denial of services and tree based attack detection method," in *2010 10th IEEE International Conference on Computer and Information Technology*, June 2010, pp. 1009–1014.

[13] N. Z. Bawany, J. A. Shamsi, and K. Salah, "Ddos attack detection and mitigation using sdn: Methods, practices, and solutions," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 425–441, Feb 2017. [Online]. Available: https://doi.org/10.1007/s13369-017-2414-5

[14] E. Markakis, Y. Nikoloudakis, G. Mastorakis, C. X. Mavromoustakis, E. Pallis, A. Sideris, N. Zotos, J. Antic, A. Cernivec, D. Fejzic, J. Kulovic, A. Jara, A. Drosou, K. Giannoutakis, and D. Tzovaras, "Acceleration at the edge for supporting smes security: The fortika paradigm," *IEEE Communications Magazine*, vol. 57, no. 2, pp. 41–47, February 2019.

[15] J. Jiang, Q. Yu, M. Yu, G. Li, J. Chen, K. Liu, C. Liu, and W. Huang, "Aldd: A hybrid traffic-user behavior detection method for application layer ddos," 08 2018, pp. 1565–1569.

[16] D. Aksu, S. Ustebay, M. Aydin, and T. Atmaca, *Intrusion Detection with Comparative Analysis of Supervised Learning Techniques and Fisher Score Feature Selection Algorithm*, 09 2018, pp. 141–149.

[17] M. Roopak, G. Yun Tian, and J. Chambers, "Deep learning models for cyber security in iot networks," 01 2019, pp. 0452–0457.

[18] P. Faruki, V. Ganmoor, V. Laxmi, M. Gaur, and A. Bharmal, "Androsimilar: robust statistical feature signature for android malware detection," 11 2013, pp. 152–159.

[19] C.-Y. Huang, Y.-T. Tsai, and C.-H. Hsu, *Performance Evaluation on Permission-Based Detection for Android Malware*, 01 2013, vol. 21, pp. 111–120.

[20] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, ""andromaly": A behavioral malware detection framework for android devices," *J. Intell. Inf. Syst.*, vol. 38, pp. 161–190, 02 2012.

[21] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. Bringas, and G. Alvarez, *PUMA: Permission Usage to Detect Malware in Android*, 01 2013, vol. 189, pp. 289–298.

[22] R. Rehman, G. Hazarika, and G. Chetia, "Malware threats and mitigation strategies," 01 2005.