ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/347445002

Botnet Attack Detection Using Machine Learning

Conference Paper · November 2020

DOI: 10.1109/IIT50501.2020.9299061

CITATIONS
64

READS
7,394

5 authors, including:

Salam Dhou
American University of Sharjah
70 PUBLICATIONS **916** CITATIONS

SEE PROFILE

Fadi A. Aloul
American University of Sharjah
166 PUBLICATIONS **5,533** CITATIONS

SEE PROFILE

All content following this page was uploaded by Salam Dhou on 24 February 2021.

The user has requested enhancement of the downloaded file.

Botnet Attack Detection using Machine Learning

Mustafa Alshamkhany, Wisam Alshamkhany, Mohamed Mansour, Mueez Khan, Salam Dhou, Fadi Aloul
Department of Computer Science and Engineering
American University of Sharjah
United Arab Emirates
{b00061219, b00061217, b00065913, b00068255, sdhou, faloul}@aus.edu

***Abstract*–With the advancement of computers and technology, security threats are also evolving at a fast pace. Botnets are one such security threat which requires a high level of research and focus in order to be eliminated. In this paper, we use machine learning to detect Botnet attacks. Using the Bot-IoT and University of New South Wales (UNSW) datasets, four machine learning models based on four classifiers are built: Naïve Bayes, K-Nearest Neighbor, Support Vector Machine, and Decision Trees. Using 82,000 records from UNSW-NB15 dataset, the decision trees model has yielded the best overall results with 99.89% testing accuracy, 100% precision, 100% recall, and 100% F-score in detecting botnet attacks.**

Keywords–IoT, botnet, machine learning, computer security, DDoS, cyberattack, classification

I. INTRODUCTION

As the Internet of Things (IoT) is an evolving technology, more of daily household devices are getting connected to the internet [1]. This allows more devices to potentially become botnet devices. This paper aims to use Machine Learning technique to detect botnet attacks.

A botnet consists of several internet-connected devices that could have been intentionally infected with malware by cyber hackers. Botnets can be used to perform distributed denial-of-service attacks (DDoS), steal data, or given access to devices. A botnet attack is a type of malicious attack that utilizes a series of connected computers to attack or take down a network, network device, website, or an IT environment. It is perpetrated with the sole intent to disrupt normal working operations or degrade the overall service of the target system. Therefore, the successful detection and prevention of botnets would have major significance in computer security.

As more devices are becoming candidates to being botnet devices, the process of detecting and distinguishing these botnet devices can be done using various machine learning techniques. This work aims to detect botnets or malicious traffic activity using the emerging machine learning techniques and provide an improvement in the accuracy compared to the other related work.

The paper is organized as follows. Section II presents the literature review of related work. Section III shows the proposed methodology. Section IV presents a discussion of the experimental results. Section V presents the conclusion and future work.

II. LITERATURE REVIEW

Many studies have been done in recent years which show the effectiveness of using Machine and Deep Learning in detecting botnet attacks which have been rising over the years.

Some studies also focus on finding the key features or characteristics of a botnet which can help distinguish between and attack or normal traffic. Dong *et al.* [2] discussed the use and effectiveness of machine learning in botnet detection. They analyzed the structure of botnets to find key features that can help distinguish botnet traffic from normal traffic. These features can then be used for feature selection when designing our machine learning model. One such method was used by Vishwakarma *et al.* [3] who used a honeypot to lure attackers and generated data from an IoT network. This fresh data was then used to analyze different characteristics of an attack such as IP addresses, MAC addresses, packet size, etc. Furthermore, Guerra-Manzanares *et al.* [4] presented the idea of using hybrid feature selection models to reduce the feature set size to get accurate results. The data used, contains 115 features, which is extremely large for any dataset. Feature selection was done using the filter, wrapper, and hybrid models to reduce the number of features. These features were then fed into a K-Nearest Neighbor (K-NN) and Random Forest model and high accuracy of 99% was seen with both models.

One of the most promising classifiers in P2P botnet detection is the Decision Tree (DT) Classifier. Haq and Singh [5] used different classifiers as well as clustering for botnet detection. The dataset used contained over 38,000 records of network traffic consisted of attack and normal traffic. In this paper, the DT classifier had the best accuracy of 90.2723% followed by Decision Tree classifier with an accuracy of 87.7853%. Similarly, according to Khan *et al.* [6], P2P botnets were difficult to detect because they own typical features of centralization and distribution. In 2013, Khan *et al.* suggested a detection method for P2P botnets, consisting of 2 stages. The first stage consisted of port judgement, DNS query and data flow count to filter non-P2P traffic. The 2nd stage used the bases of session features to reduce the number of packets being analyzed. Machine learning algorithms were also being used to classify and identify the traffic. The CTU-dataset, which contains 13 different botnet samples was used to do the experiment. Three main ML algorithms were based on session characteristics to detect P2P botnet traffic. The algorithms used were Naïve Bayes (NB), DT classification, and ANN. The results showed that the detection rate using NB and ANN was 75.5% and 93.8%, respectively, but the DT algorithm showed 94.4% accuracy. This showed that the two-stage technique by P2P traffic filtering and the DT classifier based on session characteristics proved to effectively detect P2P botnet traffic.

Other effective classifiers are the random forest and decision tree classifiers. Stevanovic and Pedersen [7] explored how botnet detection can be achieved with high accuracy by using supervised machine learning. Firstly, they

proposed a botnet detection system that uses flow-based traffic analysis and supervised machine learning as a tool for identifying botnets. They then proceed to test performances of eight of the most important machine learning algorithms (MLAs) for classifying botnets traffic. Finally, they explored how much traffic needs to be observed for successful classification. Traffic analysis was done by either "batch" analysis, which is monitoring from the start until the end of the trace, or by "limited" analysis, where time intervals and packet numbers are limited. The experiments were conducted using the ISOP dataset which includes malicious and non-malicious records. The results showed that while the random forest classifier had the highest accuracy of botnet detection, the random tree classifier was considered as optimal because it had the best balance between accuracy and time of detection. In a newer paper in 2018, Hoang *et al.* [8] proposed an evaluation on botnet detection model using machine learning algorithms in comparison to anomaly-based botnet detection methods. The paper used K-NN, C4.5, random forests (RF) and NB classifiers for their machine learning model. For the success of the machine learning model, they chose to use Domain Name Service's superior and the classifier with the best performance. The results showed an overall accuracy of 90% in detecting botnet using random forest. Jin *et al.* [9] also conducted research using DNS to detect botnets. Six special features of botnet domain traffic were selected based on their DNS logs. The selected features consisted of name-based features, such as the meaningful length ratio. This was followed by message-based features, such as the number of source IPs, types and A, AAAA, NS and MX queries. Finally, quantity-based features, such as the total queries per day and how much querying was done per hour. After selecting features, three popular ML classifiers were used to pick the malicious domains from the DNS traffic. The used classifiers were Adaboost, Bagging, and NB. The results showed good performance with precision rates above 90% for all classifiers, with only minor differences between them. Such results clearly demonstrate the success of detecting and interrupting malicious botnet behaviors when their domain names appear in the traffic. In future studies, Jin *et al.* suggest deploying this system with larger DNS logs.

Garg *et al.* [10] compared three machine learning algorithms in their ability to classify botnet vs normal traffic. This was done by selecting key features of the network traffic, and then grouping them in different combinations to produce several test cases for the algorithms. The three algorithms tested were the NB, Nearest Neighbour (IBk), and J48. After testing each algorithm with all the cases separately, the results showed that the detection accuracy of the J48 and IBk algorithms was higher than that of NB, but with limitations that J48 required a high training time and IBk required a high testing time. Despite these limitations, the high detection rates of more than 99% enabled the detection of P2P botnets.

The SVM classifier also proved a very effective classifier. Saad *et al.* [11] conducted a study on detected P2P botnets using network behavior analysis and machine learning. In their experiment they used two datasets of over 370,000 which represented only malicious traffic. For normal traffic, they used a labeled dataset with a million packets of normal traffic which they obtained from Ericsson Research Traffic Lab. The classifiers they used were K-NN, Linear Support Vector Machine (SVM), Artificial Neural Network, Gaussian Based Classifier, and NB Classifier. After running their experiment, the result showed that SVM had the highest training time and classification time. For the detection rate, SVM had the highest accuracy of almost 98% and the lowest error rate of around 6%. Gaussian Classifier came second with 96% accuracy however it had the highest error rate of 20% out of all classifiers. NB had the lowest accuracy of 89% and the second-highest error rate of around 12%. In 2014, Lin *et al.* [12] built a model based on a botnet detection using supervised machine learning classifiers, specifically a combination to between SVM and artificial fish swarm algorithm (AFSA). The data set they used was collected through a Local Area Network which was made to collect the packets data of the network and used it as a prototype simulation of botnet attacks traffic or normal traffic. Their results show after 5-fold cross-validation that the combination of SVM and AFSA performed better than other classifiers with 99% average accuracy rate.

Another common classifier used in this domain is the KNN classifier. In 2013, Feizollah *et al.* [13] conducted a study on Android malware detection, more specifically, anomaly-based mobile botnet detection. The study used five machine learning classifiers, namely K-NN, Multi-Layer Perceptron (MLP), DT and SVM. In this study, they used malware data samples from the Android Malware Genome Project and evaluated the classifiers on them. This study selected 3 network features: connection duration, TCP size and number of GET/POST parameters. The result showed that the best classifier obtained is the KNN with a true positive rate as high as 99.94% and false positive of 0.06%. In a newer paper in 2018, Arif *et al.* [14] proposed a network to use machine learning classifiers, namely DT, K-NN, NB and Random Forest (RF) to detect HTTP botnets. The dataset they used for their research was extracted from the network traffic which is based on TCP packet feature. The results showed that the best classifier to detect HTTP botnet attacks in the network traffic is KNN classifier with an average accuracy of 92.93% for each type of botnet family.

Deep and unsupervised learning have also shown promise in detecting botnet attacks. Pour *et al.* [15] used a multi-window convolution neural network combined with clustering to detect over 350 IoT botnets in darknet traffic. Al Shorman *et al.* [16] proposed an unsupervised intelligent system which was made from SVM and Grey Wolf optimization for detecting IoT botnets. This model was able to achieve low detection time and reduce the number features that were used for detection.

From these studies, it is found that machine learning is highly applicable and effective in botnet detection. The main contribution of this study is to create a botnet detection model using machine learning classification approach.

III. PROPOSED METHODOLOGY

A. *Dataset Description*

Several datasets are available for this work such as the Bot-IoT and the UNSW-NB15 datasets. The Bot-IoT dataset contains over 72 million records with 42 features (27 Integer, 13 Float, and 2 String types) and was created by

setting up a botnet network in a controlled environment and monitoring the network traffic to capture any packets that were being sent. The dataset contains labeled normal and malicious traffic which includes attacks such as DDoS, DoS, OS Scan, etc. The other dataset, UNSW-NB15 contains 43 features (14 Float, 6 Strings and 23 Integer types) and 2.5 million records which are labeled as either attack traffic or normal traffic and further expanded to the category of attack and the subcategory. In addition to DDoS and DoS attacks, the dataset contains records for Fuzzers, Backdoor, Reconnaissance and Worm attacks [17]. These records were collected in pcap files and then converted to CSV to create the dataset. Both datasets have been compiled and made publicly available by UNSW Canberra for research purposes [18]. Furthermore, it is found that the UNSW-NB15 dataset is a more polished dataset, where has similar attributes to the Bot-IoT dataset but is more diverse in the type of malicious records it has.

In this work, the UNSW-NB15 dataset was used as it is more detailed and was labeled by UNSW to be the best dataset for training. In this work, 82,000 records were randomly selected and utilized. The data was cleaned and prepared to classify the training and testing models. Categorical data such as the "proto", type of "service', "state", "sptks", "sload", and "attack cat" were coded into numerical data. Random splits were performed on the data to divide it into a training set, that has 80% of the data, and a testing set, that has 20% of the data.

B. *Feature selection and Dimensionality reduction*

For our model, we applied a feature selection and dimensionality reduction to reduce the dimensionality of the data while preserving the variance in the data. One method of dimensionality reduction is Principal Component Analysis (PCA). PCA is a data transformation method that projects the data into a new feature space where most of the variance in the data is represented by the first coordinate of the new space (called the first principal component), the second most variance on the second principal component, and so on. In this work, the features with the greatest variance are selected to be used as input to the classifiers. Using this process, we ensure that only relevant features are selected which will contribute to the computational efficiency and the simplicity of the machine learning models. Furthermore, this process has the feasibility to reduce any overfitting that might occur using all the features.

C. *Classification Algorithms*

Several classifiers were used and evaluated in this work. Scikit-learn library in Python was used in this work. For the evaluation, the confusion matrix was computed to calculate precision, recall, and F-Measure using the true and predicted labels from the model. The classifiers used in this paper are:

1) Naïve Bayes (NB) with Gaussian probabilities – a probabilistic classifier that uses the Bayes theorem and assumes a conditional independence between the different features of the dataset. NB estimates the class probability based on the training set.

2) K-Nearest Neighbor (k-NN) – a non-parametric algorithm used for classification and regression. To predict the class, the model assigns the class of the test sample based on the majority of the *k* nearest neighbors of that given test sample.

3) Support Vector Machine (SVM) with nonlinear kernel, namely radial basis function (RBF) – creates a decision boundary based on samples of different classes. The shape of the decision boundary is created based on the kernel function used, and key hyper parameters such as *C* which controls the tradeoff between the smoothness of the decision boundary and the correctness of the classification, and *gamma* which defines the influence of the data points' distribution on the shape of the decision boundary.

4) Decision Tree (DT) – a tree-like classification model where each node in the tree specifies a test on a single feature and each branch descending from that node corresponds to one of the possible values for that feature.

To apply these classifiers, the dataset was randomly split into training and testing datasets. The training data was used to train the classifiers. The classifiers were then tested using the testing dataset to predict the labels. The classifiers were all evaluated and compared as discussed in Section IV.

IV. Discussion Of Results

To summarize, we have applied DT, SVM, KNN and NB classifiers and analyzed the result of each classifier model. Meaning, we looked at the prediction test and train accuracy, precision, recall and F-measure scores as well as their confusion matrix for each classifier. The result organized in the table below.

A. *Principal Component Analysis (PCA)*

PCA was applied on the dataset to reduce the number of dimensions in a dataset while preserving its variance. The results of performing PCA on the dataset are shown in Table II. After running a simulation that varies the PCA parameters, the best parameters that yielded the highest accuracy were recorded. As can be seen in the table, the number of components chosen for DT and NB was 43. For k-NN and SVM with RBF kernel, the number of components used was 20. DT and SVM with RBF kerned had the highest accuracy with the optimal number of components.

TABLE I
RESULTS OF APPLYING PCA ON THE DATASET USING DIFFERENT CLASSIFIERS

Classifier	Number of Components Chosen	Training Accuracy	Testing Accuracy
Decision Trees	43	99.28%	98.87%
Naïve Bayes	43	85.69%	85.91%
k-NN	42	90.5%	82.14%
SVM – rbf	20	99.9%	99.25%

B. Feature Selection

In this work, the best features were selected to reduce the dimensionality of the dataset and remove the effect of training the classifiers using irrelevant features. Multiple runs were performed using Chi-squared and ANOVA F-value to determine the degree of correlation between the features and the output labels. It was found that Chi-squared criteria was best suited to be used to rank the features when it comes to accuracy. Using SVM with RBF, only 14 features ensured producing the best accuracy. However, DT achieved its highest accuracy using 29 features. Table II shows the accuracy of the classifiers reported using the selected features.

TABLE II
RESULTS OF APPLYING FEATURE SELECTION ON THE DATASET USING DIFFERENT CLASSIFIERS

Classifier	Percentile of selected features	Number of selected features	Training Accuracy	Testing Accuracy
Decision Trees	67%	29	100%	100%
Naïve Bayes	67%	29	96.40%	96.39%
k-NN	72%	31	90.47%	82.23%
SVM– rbf	32%	14	99.85%	99.00%

With these results, it is observed that the DT model provided the best training and testing accuracy with a total of 29 features involved with the training and testing accuracy of 100%. Table III displays the best features selected for the detection of malicious traffic and their description as given from the dataset.

TABLE III
BEST FEATURES SELECTED FOR THE DETECTION OF MALICIOUS TRAFFIC

Name	Type	Feature description
Spkts	integer	Source to destination packet count
Dpkts	integer	Destination to source packet count
Sbytes	Integer	Source to destination transaction bytes
Dbytes	Integer	Destination to source transaction bytes
Rate	Integer	Total packets per second in transaction
sttl	Integer	Source to destination time to live value
dttl	Integer	Destination to source time to live value
Sload	Float	Source bits per second
Dload	Float	Destination bits per second
dloss	Integer	Destination packets retransmitted or dropped
Sintpkt	Float	Source interpacket arrival time (mSec)
Dintpkt	Float	Destination interpacket arrival time (mSec)
Sjit	Float	Source jitter (mSec)
Djit	Float	Destination jitter (mSec)
swin	integer	Source TCP window advertisement value
stcpb	integer	Source TCP base sequence number
dtcpb	integer	Destination TCP base sequence number
dwin	integer	Destination TCP window advertisement value
dmean	integer	Mean of the packet size transmitted by the dst
smean	integer	Mean of the packet size transmitted by the src
res_bdy_len	integer	Actual uncompressed content size of the data transferred from the server's http service.
ct_srv_src	integer	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
ct_dst_ltm	integer	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
ct_src_ ltm	integer	No. of connections of the same source address (1) in 100 connections according to the last time (26).
ct_src_dport_ltm	integer	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
ct_dst_sport_ltm	integer	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
ct_dst_src_ltm	integer	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).
cat	nominal	The name of each attack category. In this data set , nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellcode and Worms

C. Model Selection and Classification:

In this work, using the best features found in each classifier, we applied grid search with a 5-fold cross validation on all the models to get the best combination of hyper parameters for each model and to ensure its optimality. The summary of results is shown in Table IV.

For the NB classifier, the best hyper parameter values found to produce best results were 1e-20 for 'var_smoothing'. The model preforms fairly good in comparison to other models where we the training and testing accuracy were 97% and 96% and precision, accuracy, recall and f-score all lie above 96%. The model obtain exceptionally high false classifications in the confusion matrix as it generates some false positives and false negatives classifications. Nevertheless, this model proves to be a good model for classifying botnet attacks.

Similarly, using grid search on KNN, it was found that the best hyper parameter for the number of neighbors 'K_neighbors' is 3. It is noticeable that the classifier's performance results were generally the lowest compared to the other models. The training and testing accuracies reported for this model were 91% and 83%, respectively while precision, accuracy, recall and f-score are all around 81%. There is a considerable amount of false positives and false negatives classifications according to the confusion matrix.

For the SVM with RBF kernel, the dataset records were reduced to 10,000 as SVM classifier is computationally expensive. The best hyper parameters that were found are of

C: 10, and *gamma*: 0.0001. The results using this model is satisfactory where the training and testing accuracies were 100% and 98%, with good the precision, accuracy, recall and f-score results.

Lastly, after performing a grid search on DT to find its optimum hyper parameters, we found that the best hyper parameters are max depth of 100, max features at 10, max leaf nodes are none and min sample leaf at 10. This model outshines the other classifiers as it uses the entire dataset and provides the highest performance values. In detail, the training and testing accuracy are averaged at 99.9% and 99.8%, respectively, while the precision, accuracy, recall and f-score all lie in 100% with a significantly low or almost zero false classifications in the confusion matrix. Not only it is not computationally expensive, the DT classifier model was able to correctly differentiate the malicious traffic and normal traffic statistically better than the previously discussed classifiers.

This work improved the performance of the other machine learning models proposed in the literature. It demonstrated a 1% increase in the detection accuracy over the work at [4] and [12] and a 9.8% increase over the work proposed in [5].

TABLE IV
SUMMARY OF PERFORMANCE RESULTS FOR ALL THE COMPARED CLASSIFIERS

	Evaluation	Classifier - 5-fold validation			
		DT	NB Gaussian	SVM-rbf	KNN
Accuracy	Training accuracy	99.91%	97.10%	100%	91%
	Testing accuracy	99.89%	96.90%	98.80%	83.00%
Confusion Matrix	TP	7443	6986	22	6024
	FP	8	461	24	1346
	FN	10	53	0	1422
	TN	9006	8967	1954	7646
Precision	Normal traffic	100%	99%	100%	81%
	Attack traffic	100%	95%	99%	85%
	Macro avg	100%	97%	99%	83%
	Weighted avg	100%	97%	99%	83%
Recall	Normal traffic	100%	94%	48%	81%
	Attack traffic	100%	99%	100%	84%
	Macro avg	100%	97%	74%	83%
	Weighted avg	100%	97%	99%	83%
F1-score	Normal traffic	100%	96%	65%	81%
	Attack traffic	100%	97%	99%	85%
	Macro avg	100%	97%	99%	83%
	Weighted avg	100%	97%	82%	83%
Support	Normal traffic	7451	7447	46	7400
	Attack traffic	9016	9020	1954	9067
	Macro avg	16467	16467	2000	16467
	Weighted avg	16467	16467	2000	16467
	Accuracy	16467	16467	2000	16467

V. CONCLUSION AND FUTURE WORK

In this paper, the detection of botnet or malicious traffic activity using the emerging machine learning techniques was proposed. Four classifiers were applied on this work, namely Naïve Bayes, K-Nearest Neighbor, Support Vector Machine, and Decision Trees. The experimental results revealed that the decision tree model performed better than the other classifier models as well as a slight improvement on the models that were previously mentioned in the reviewed literature. Theoretically, this model can be used to detect several botnet attacks and other type of malicious network activity.

As a future work, the entire UNSW-NB15 dataset must be tested with the same model to validate the results. This experiment can also be extended to include other datasets such as the Bot-IoT dataset and the CTU-13 which are more recent to compare the performance of the algorithms with different kinds of botnet traffic. More classifiers such as logistic regression and neural networks can be also tested. Furthermore, unsupervised learning methods such as clustering can be investigated and compared with the supervised learning methods used in this paper. Moreover, other methods of feature selection can be examined to refine these results further. Lastly, the machine learning model can be tested on a real-time controlled environment to accurately measure the model's performance and how it handles different types of threats such a zero-day threats.

REFERENCES

[1] S. Ranger, "What is the IoT? Everything you need to know about the Internet of Things right now | ZDNet," ZDNet, 2020. [Online]. Available: *https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/*.

[2] X. Dong, J. Hu and Y. Cui, "Overview of Botnet Detection Based on Machine Learning," *International Conference on Mechanical, Control and Computer Engineering*, Huhhot, pp. 476-479, 2018.

[3] R. Vishwakarma and A. Jain, "A Honeypot with Machine Learning based Detection Framework for defending IoT based Botnet DDoS Attacks," *International Conference on Trends in Electronics and Informatics* (ICOEI), Tirunelveli, India, pp. 1019-1024, 2019.

[4] A. Guerra-Manzanares, H. Bahsi and S. Nõmm, "Hybrid Feature Selection Models for Machine Learning Based Botnet Detection in IoT Networks," *International Conference on Cyberworlds (CW),* Kyoto, Japan, pp. 324-327, 2019.

[5] S. Haq and Y. Singh, "Botnet Detection using Machine Learning," *International Conference on Parallel, Distributed and Grid Computing* (PDGC), India, pp. 240-245, 2018.

[6] R. Khan, R. Kumar, M. Alazab and X. Zhang, "A Hybrid Technique To Detect Botnets, Based on P2P Traffic Similarity," *Cybersecurity and Cyberforensics Conference* (CCC), Melbourne, Australia, pp. 136-142, 2019.

[7] M. Stevanovic and J. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," *International Conference on Computing, Networking and Communications*, HI, pp. 797-801, 2014.

[8] X. Hoang and Q. Nguyen, "Botnet Detection Based On Machine Learning Techniques Using DNS Query Data," *Future Internet*, vol. 10, no. 5, p. 43, May 2018.

[9] J. Jin, Z. Yan, G. Geng and B. Yan, "Botnet Domain Name Detection based on machine learning," *International Conference on Wireless, Mobile and Multi-Media* (ICWMMN), Beijing, China, pp. 273-276, 2015.

[10] S. Garg, A. Singh, A. Sarje and S. Peddoju, "Behaviour analysis of machine learning algorithms for detecting P2P botnets," *International Conference on Advanced Computing Technologies*, pp. 1-4, 2013.

[11] S. Saad *et al.*, "Detecting P2P botnets through network behavior analysis and machine learning," *International Conference on Privacy, Security and Trust*, Montreal, QC, pp. 174-180, 2011.

[12] K.-C. Lin, S.-Y. Chen, and J. C. Hung, "Botnet Detection Using Support Vector Machines with Artificial Fish Swarm Algorithm," *Journal of Applied Mathematics*, vol. 2014, Article ID 986428, 2014.

[13] A. Feizollah, N. B. Anuar, R. Salleh, F. Amalina, R. Ma'arof, and S. Shamshirband, "A Study Of Machine Learning Classifiers for Anomaly-Based Mobile Botnet Detection," *Malaysian Journal of Computer Science,* 26(4), 2013.

[14] R. Dollah, Faizal. A., F. Arif, M. Mas'ud and L. Xin, "Machine Learning for HTTP Botnet Detection Using Classifier Algorithms," *Journal of Telecommunication, Electronic and Computer Engineering*, 10(1-7), pp. 27- 30, 2018.

[15] M. Pour, A. Mangino, K. Friday, M. Rathbun, E. Bou-Harb, F. Iqbal, S. Samtani, J. Crichigno, and N. Ghani, "On Data-driven Curation, Learning, and Analysis for Inferring Evolving Internet-of-Things (IoT) Botnets in the Wild," *Computers & Security*, vol. 91, April 2020.

[16] A. Al Shorman, H. Faris, and I. Aljarah, "Unsupervised intelligent system based on one class support vector machine and Grey Wolf optimization for IoT botnet detection," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 2809–2825, July 2020.

[17] Moustafa, Nour, and Jill Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," *Military Communications and Information Systems Conference* (MilCIS), Australia, 2015.

[18] N. Koroniotis, N. Moustafa, E. Sitnikova and B. Turnbull, "Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset," *Future Generation Computer Systems*, vol. 100, p. 779-796, November 2019.

View publication stats