RESEARCH ARTICLE

A Smart System for Detecting Behavioural Botnet Attacks using Random Forest Classifier with Principal Component Analysis

O. E. Taylor and P. S. Ezekiel

ABSTRACT

Over the years, malware (malicious software) has become a major challenge for computer users, organizations, and even countries. In particular, a compromise of a set of inflamed hosts (aka zombies or bots) is one of the severe threats to Internet security. Botnet is described as some computer systems or devices controlled on the Internet to carry out unintentional and malicious acts without the owner's permission. Due to the continuously progressing behavior of botnets, the conventional methods fail to identify botnets. In other to solve the stated problem, this paper presents a smart system for detecting behavioural bootnet attacks using Random Forest Classifier and Principal Component Analysis (PCA). The system starts with a botnet dataset that was used in building a robust model in detecting Bootnet attacks. The dataset was pre-processed using pandas library for data cleaning. PCA was used in reducing the dimension of the dataset, so as to avoid data imbalance. The result of the PCA was used as input to the random forest classifier. The random forest classifier was trained using the number of estimators as 1000. The result of the model shows a promising accuracy of about 99%.

Keywords: Behavioural Bootnet, Distributed Denial of Service, Principal Component Analysis, Random Forest Classifier.

Published Online: March 23, 2022

ISSN: 2796-0064

DOI : 10.24018/ejai.2022.1.2.4

O. E. Taylor *
Department of Computer Science, Rivers State University, Port Harcourt, Nigeria.
(e-mail: taylor.onate@ust.edu.ng).
P. S. Ezekiel
Department of Computer Science, Rivers State University, Port Harcourt, Nigeria
(e-mail: ezekielpromise27@gmail.com)

Corresponding Author

I. Introduction

Over the years malware (e.g., malicious software) has turned out a major challenge for computer users, organizations, and even countries. In particular, a compromise of a set of inflamed hosts (aka zombies or bots) is one of the severe threats to Internet security. An excessive quantity of malicious and fraudulent acts may be carried out in a dispensed style through these bots. As the outcome, computer system structures may be compromised through a botnet problem to a range of various attack vectors but not constrained to worms, viruses, vulnerabilities, social engineering, etc. They are controlled through C&C servers that conceal cybercriminals while acting their automatic and disbursed malicious tasks [1]. Botnet is described as some computer systems or devices controlled on the Internet to execute unintentional malicious acts without the owner's permission. Because the continuously progressing of botnet behaviour, the conventional approach has failed in recognizing botnets. The characterized illustration of botnet, acts in phrases of network connectivity, has been beneficial for coming across variants of bootnets. The behavioural approach of botnet is illustrated by coming across the subsequent and usual acts on the network which also can be used for recognition purposes. These verbal exchange traces may be represented because the basic patterns of verbal exchange act to discover botnet traffic [2].

Botnet detection strategies may be labeled as host-based and totally formulated on the computer network. With the dependent host method, it identifies unusual system utilization (e.g., improved CPU utilization and reminiscence utilization) is identified. This approach is not stricken by encrypted communication ports. However, the downside is that the resource utilization of all hosts that have ended needs to be monitored which may be pretty expensive in respect to time. In contrast, with the dependent host method, the operation of the connection of the network is checked and the network traffic throughout the bot lifecycle is recognized [3].

In detecting botnet attacks based on computer networks, there are anomaly-based and signature-based techniques. The signature-based techniques carry out analyses consistent with rules set via way of means of making use of Deep Packet Inspection (DPI) at the contents of TCP/UDP packets. This technique has a small rate of false-positive and is efficient in detecting known botnets. However, it could best discover most effective acknowledged threats: i.e., it can't deal with new sorts and subspecies [4]. Therefore, subsequent updating of signatures is necessary. Another problem with the usage of signature-based techniques is that signatures may be hidden through encryption and obfuscation of the C2C channel, Fast-Flex, DGA, etc. With anomaly-based techniques, abnormalities that include packet payload and bot institution conduct are identified. These strategies have completed the usage of diverse algorithms such as machine learning strategies, graph evaluation and are extraordinarily exceptional for extracting sudden patterns of networks. Anomaly-based techniques are normally the desired manner to discover unknown botnets. However, these strategies have a high rate of false positives, which ends up causing high rate of detection errors [5].

II. Related Works

[1] made use of a deep neural network in detecting botnet attacks carried out on congested network flow. The overall performance of their proposed approach was evaluated with generally accepted datasets. Their experimental results demonstrate that the use of deep learning approach is best efficient in detecting botnet attacks with a high of true positive (93.6%) and low rate of false-positive.

[3] adopted deep-learning algorithm in detecting botnet attacks. The dataset used here was examined from a congested fact of malware gathered from a computer network system. The results obtained from their experiment show that their proposed model can stop botnets attacks from an infected computer network system that serves as host. The system achieved a detecting rate of 99.2%

[6] presents a framework for identifying botnets attacks on a network system using deep learning method. Their proposed system gathers the movements of the network packets, transforms them into connection data, and applied deep learning approach in detecting and estimating the attacks from IoT gadgets. For a flawless system, they compared their proposed method with other existing methods, and the results show that their system was outstanding, having an accuracy level of 96.8%.

[7] applied dimensionality reduction and deep learning methods in building a robust model in identifying botnet attacks on IoT networks. The dimensionality reduction approach was used in reducing the number of features on the dataset while the Long Short-Term Memory Autoencoder approach was applied on the reduced features in building a model in detecting IoT botnet attacks. Their experimental result shows a better detection rate of about 99.48%.

[8] applied feed-forward neural network in detecting a botnet attack on a computer network. The authors examined a well-known dataset that comprises thirteen network packets. From their experiment conducted, the results show that feed-forward neural network has a high detection rate of about 99.6% across 20 plus iterative steps.

[9] presented a federated deep learning model to prevent data leakage in IoT-based edge devices. A simulation was carried out using a zero-day-botnet as input data. The results of the simulation show that the federated deep learning method detects zero-day-botnet attacks good rate of classification evaluation and low latency of the network.

[10] present a framework for detecting botnet on social network communities. They applied a deep autoencoder technique in checking for similar behaviours of botnet attacks on twitter network system. For evaluating the performance of their proposed model, an experiment was being conducted using two Twitter datasets. The results from their experiments show that their model has a better detection rate of about 90.86%.

[11] used a gradual statistical tool to extract fundamental congested features of IoT gadgets and applied the Z-Score approach for features normalization. After that, they made use of multivariate correlation analysis for data generation. They also applied a convolutional neural network (CNN) to examine the dataset and make use of the trained CNN for traffic identification. Their proposed CNN model shows a high detection rate of about 99.57%.

[12] presents a framework in identifying the activities of botnet attacks on users’ network systems. They applied the word embedding approach for packet tokenization and long short-term memory technique for building a robust model using the tokenized data. The results obtained from their experiments show the performance of their proposed model with an accuracy result of about 98.52%.

[13] presented a deep neural network and smote deep neural network in building a robust system for detecting botnet attacks on IoT devices. The bot-IoT dataset was used in this work and their experimental results show that their proposed model had an accuracy result of 99.50%.

III. Design Methodology

A. *Bootnet Dataset*

The dataset used in this work is a botnet IoT dataset. The dataset comprises of thousands of traffics that was carried out on a network system. The dataset includes attacks like Distributed Denial of Service (Ddos), Denial of Service (Dos), Data Theft, Operating system and service scan, Reconnaissance, Operating system fingerprint, keylogging, and Data Exfiltration. These attacks were categorized into different subcategories. The attacks were group according to how they are being carried out. Attacks that were carried via Http were group together, attacks carried out by TCP were grouped together and attacks carried out by Keylogging were also grouped together and many more. The dataset comprises of over 72, 000 records, and the size of the dataset was over 1gb. For better training of a robust model in detecting these attacks, we used a dimensionality reduction technique in reducing the dimension of the dataset.

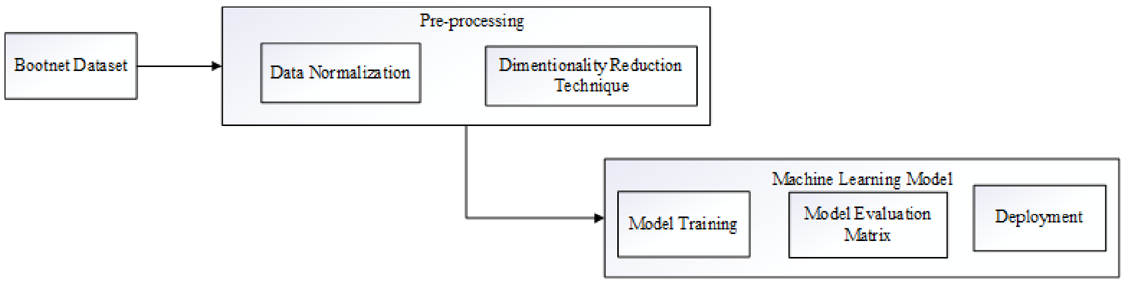


Fig. 1. Architecture of the Proposed System.

B. *Pre-processing*

Data pre-processing has to do with the restructuring of the bootnet IoT dataset into a structured and standardized format that will be suitable in training a machine learning model. Here we perform some data cleaning by checking and removing of noise, empty spaces, and bringing all the data into a standardized form through scaling. Further pre-processing has to do with reducing the dimension of the dataset through a dimensionality reduction technique. The dimensionality reduction technique used here is Principal Component Analysis (PCA). Principal Component analysis was used in selecting the most import features of the botnet IoT-dataset without distorting the dataset. A pseudocode on how PCA work can be seen below:

Algorithm and Pseudocode for Principal Component Analysis

Step 1: Standardize the Botnet IoT dataset.

Step 2: Calculate the covariance matrix for the input features in the Botnet IoT data

Step 3: Calculate the eigenvalues and eigenvectors for the covariance matrix.

Step 4: Sort eigenvalues and their corresponding eigenvectors.

Step 5: Select k eigenvalues and form a matrix of eigenvectors.

Step 6: Perform a transformation of the original matrix.

Pseudocode for PCA

```
 1: procedure PCA
 2: Compute dot product matrix: $\mathbf{X}^T\mathbf{X} = \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^T(\mathbf{x}_i - \boldsymbol{\mu})$
 3: Eigenanalysis: $\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$
4: Compute eigenvectors: $\mathbf{U} = \mathbf{X}\mathbf{V}\boldsymbol{\Lambda}^{-\frac{1}{2}}$
5: Keep specific number of first components: $\mathbf{U_d} = [\mathbf{u}_1,...,\mathbf{u}_d]$

6: Compute $d$ features: $\mathbf{Y} = \mathbf{U_d}^T\mathbf{X}$
```

C. *Machine Learning Model*

The machine learning model was built using Random Forest classifier. The result of the Principal Component Analysis was feed to the Random Forest model for training. Before the, the result of the PCA which serves as the most important features of the Bootnet IoT data was divided into two parts. 80% was used for training and 20% was used for testing. An algorithm of the random forest model can be seen bellow:

D. *Precondition*

A training set S:= (x1,y1),...,(xn,yn), features F, and number of trees in forest B.

```
1.  function RandomForest(S, F)
2.  H ← ∅
3.  for i ∈ 1,..., B do
4.  S(i) ← A bootstrap sample from S
5.  hi ← RandomizedTreeLearn(S(i), F)
6.  H ← H ∪ {hi}
7.  end for
8.  return H
9.  end function
10. function Randomized Tree Learn (S, F) 11 At each
    node:
11. f ← very small subset of F
12. Split on best feature in f
13. return the learned tree
14. end function.
```

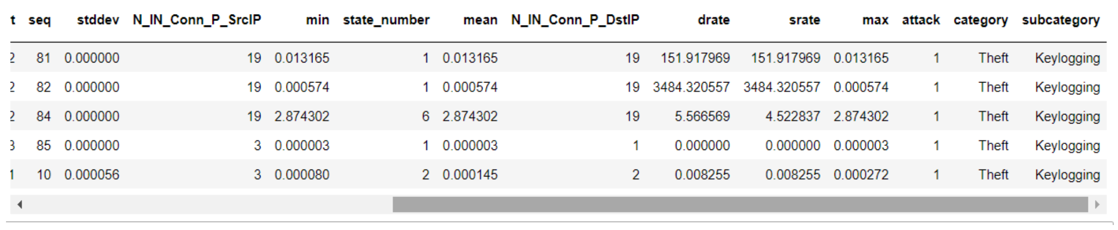
An evaluation matrix was used in checking the performance of a random forest model in terms of accuracy, precision, recall and F1-Score. A classification report and a confusion matrix were used in giving a detailed explanation on the model performance. This was used in explaining the correct prediction and false prediction of the model on the test data. The correct and false prediction has to do with true positive, true negative, false positive and negative. Table I shows the matrix evaluation in terms of accuracy, precision, Error, Recall, and F-measure.

TABLE I: MATRIX EVALUATION

Definition	Formula
Accuracy	$\frac{TP+TN}{TP+FN+TN+FP}$
Error	1-accuracy
Recall	$\frac{TP}{TP+FN}$
Precision	$\frac{TP}{TP+FP}$
F-Measure	$\frac{2\ Precison \times Recall}{Precision+Recall}$

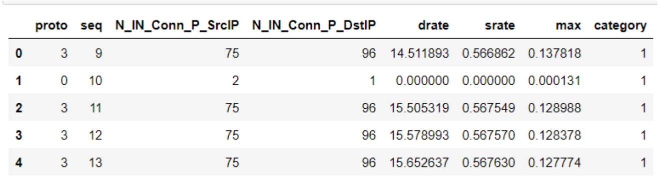
IV. RESULTS AND DISCUSSION

From the experiment conducted, this system presents a smart system for detecting bootnet attacks on a computer network system. The system uses an IoT bootnet dataset that comprises over 72,000 records. The records comprise of various attacks that were carried out on computer network systems and the channel via which these attacks were carried out. The dataset was highly imbalanced, and this was resolved by performing a random resampling on the data and by dimensionality reduction technique in reducing the size of the dataset. The dataset was pre-processed by performing data cleaning, data normalization and transformation, and feature selection. The data cleaning was performed by removing parenthesis, null values, etc., whereas the normalization of the data was performed using StandardScaler(). This was used in bringing all values to a scalable and better form. For data transformation, LabelEncoder() was used in converting the category and subcategory columns into digits. Fig. 2 and Fig. 3 show the shows the category and sub-category columns before and after transformation. In selecting the most important features, Principal Component Analysis (PCA) was used in selecting the most important features in the dataset. This was used so as to enable the model to have a better training performance. The result of the Principal Component analysis that shows the import important features of the dataset can be seen in Fig. 4. A correction matrix and interpretation of the IoT Bootnet data through data visualization can be seen in Fig. 4, 5 and 6. After the selection of the most important features, the dataset (The result of the principal Component Analysis) was divided into two parts, which are 80% for training and 20% for testing.



t	seq	stddev	N_IN_Conn_P_SrcIP	min	state_number	mean	N_IN_Conn_P_DstIP	drate	srate	max	attack	category	subcategory
2	81	0.000000	19	0.013165	1	0.013165	19	151.917969	151.917969	0.013165	1	Theft	Keylogging
2	82	0.000000	19	0.000574	1	0.000574	19	3484.320557	3484.320557	0.000574	1	Theft	Keylogging
2	84	0.000000	19	2.874302	6	2.874302	19	5.566569	4.522837	2.874302	1	Theft	Keylogging
3	85	0.000000	3	0.000003	1	0.000003	1	0.000000	0.000000	0.000003	1	Theft	Keylogging
1	10	0.000056	3	0.000080	2	0.000145	2	0.008255	0.008255	0.000272	1	Theft	Keylogging

Fig. 2. Sample of the dataset before transformation.



	proto	seq	N_IN_Conn_P_SrcIP	N_IN_Conn_P_DstIP	drate	srate	max	category
0	3	9	75	96	14.511893	0.566862	0.137818	1
1	0	10	2	1	0.000000	0.000000	0.000131	1
2	3	11	75	96	15.505319	0.567549	0.128988	1
3	3	12	75	96	15.578993	0.567570	0.128378	1
4	3	13	75	96	15.652637	0.567630	0.127774	1

Fig. 3 Sample of the Dataset after transformation.

The training data were used in feeding the random forest model. The random forest model was trained using the 100 nodes as the number of estimators. The result of the random forest model was evaluated using the classification report and confusion matrix. The classification matrix shows that the model had better training and testing performance in terms of accuracy, precision, f1-score, etc. The result of the classification report and confusion matrix can be seen in Fig. 7 and Fig. 8. The random forest model was saved and deployed to the web for further testing and detection of malicious packets. The result of the detected malicious packets can be seen in Fig. 9.

From the graphical representation of the result of Principal Component Analysis (Fig. 4) it can be seen that the x-axis represents the number of important features in the IoT Botnet dataset, whereas the y-axis represents the number of explained variance. By explained variance, we mean how many percentages of the dataset that can be explained by the six important features. The result of the Principal Component Analysis showed that out of the 47 columns, just six columns contain over 90% of the most important content of the dataset.

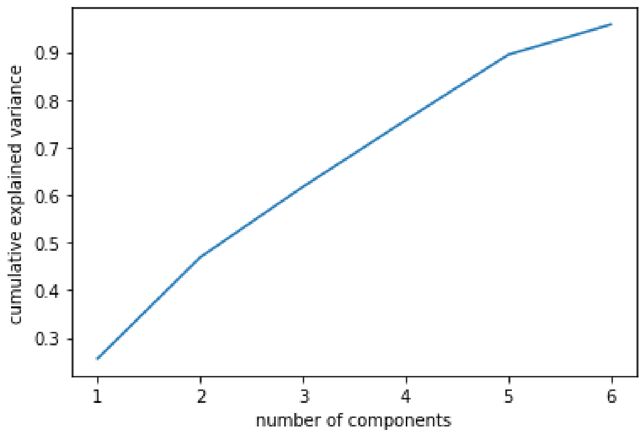


Fig. 4. Result of Principal Component Analysis.

The correlation matrix (Fig. 5) shows the coefficients between the features of our training data. The correlation matrix shows that there is a regular correlation between the input values.

The countplot (Fig. 6) shows the most type of attack that was carried out on Network system. This shows that the attackers used more of Distributed Denial of Serve, followed by Denial of Service, Reconnaissance and data theft.

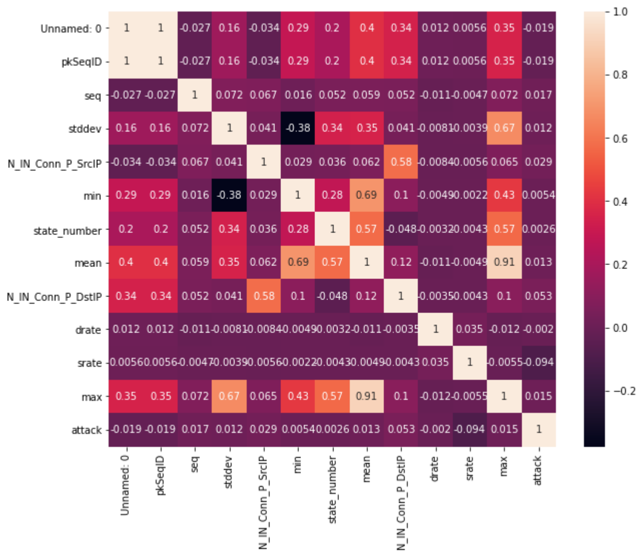


Fig. 5. Correlation Matrix.

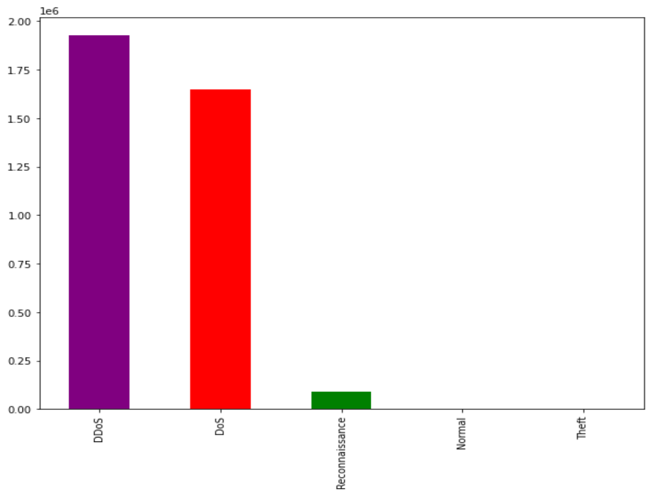


Fig. 6. Countplot of malicious Attacks.

Fig. 7 shows the channels by which these attacks were carried out. From the bar chart, it is obvious that most of these attacks are carried out more on UDP, TCP, Service Scan, Operating system fingerprint etc.

The classification report (Fig. 8) shows the accuracy and precision level for each of the five attacks that were carried out on the test data. The confusion report shows that the model has a good detection rate. This is to say that the proposed model can detect malicious packages correctly. The model has an accuracy result of about 100%.

Fig. 9 shows the number of correct predictions and the correct prediction that was carried out on the test data. In the y-axis (Malicious attacks). This shows the malicious packets that were carried out on the dataset. The value 0 represents Distributed Denial of Service attack, 1 represents Denial of Service attacks, 2 represents Reconnaissance, 3 represents normal packets and 4 represents data theft. From the confusion Distributed Denial of Service was predicted correctly to be 1347633, and it was mistakenly classified as a Denial-of-service attack 684 times, where Denial of service was detected correctly for 1154782 number of times.

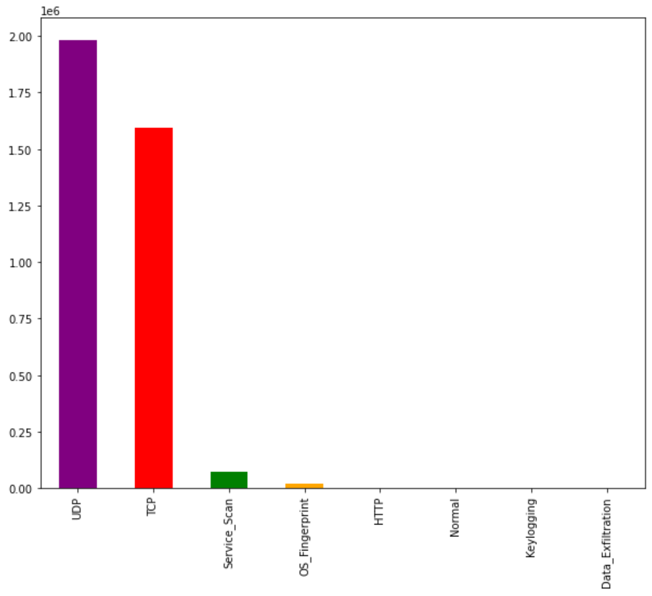
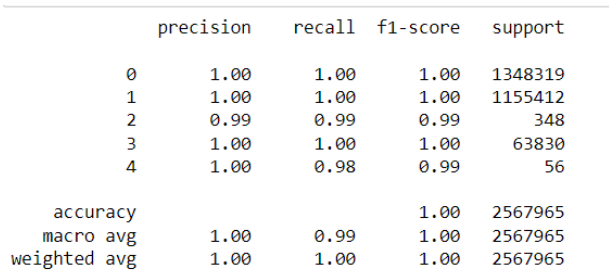


Fig. 7. Channels of Attacks.



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1348319
           1       1.00      1.00      1.00   1155412
           2       0.99      0.99      0.99       348
           3       1.00      1.00      1.00     63830
           4       1.00      0.98      0.99        56

    accuracy                           1.00   2567965
   macro avg       1.00      0.99      1.00   2567965
weighted avg       1.00      1.00      1.00   2567965
```

Fig. 8. Classification Report.

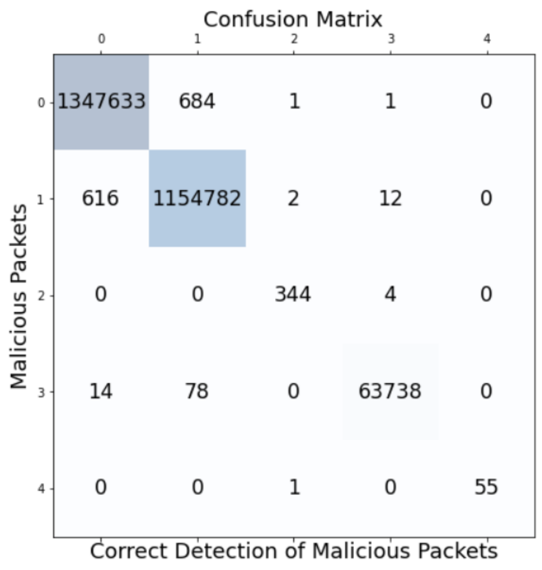
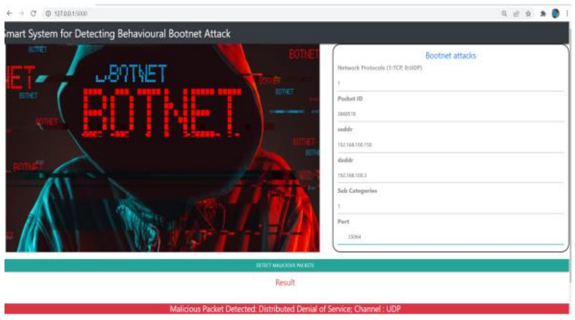


Fig. 9. Confusion Matrix.

Here, the system detected malicious packet. Here it shows that the detected attack was that of Distributed denial of service and it also stated the channel (UDP). via which the attack was carried out.



Smart System for Detecting Behavioural Bootnet Attack

Bootnet attacks

Network Protocols (1:TCP, 0:UDP)
1
Packet ID
3668518
saddr
192.168.100.150
daddr
192.168.100.3
Sub Categories
1
Port
15064

DETECT MALICIOUS PACKETS

Result

Malicious Packet Detected: Distributed Denial of Service; Channel : UDP

Fig. 10: Alert: Malicious Packet detected.

V. Conclusion and Future Research

Malicious packets on computer networks have been a greater challenge. This is because these attacks are carried out in various forms and using different formats. Botnet detection strategies may be labeled as host-based and totally formulated on the computer network. With the dependent host method, it identifies unusual system utilization (e.g., improved CPU utilization and reminiscence utilization) is identified. This approach is not stricken by encrypted communication ports. However, the downside is that the resource utilization of all hosts that have ended needs to be monitored which may be pretty expensive in respect to time. The behavioural approach of botnet is modeled by coming across the subsequent and usual acts on the network as patterns, which also can be used for detection purposes. These verbal exchange traces may be represented because the underlying patterns of verbal exchange act to discover botnet traffic. In order to solve the problem, this system proposed a machine learning model using Principal Component Analysis (PCA) and random forest classifier. A botnet dataset was used in building a robust model in detecting Bootnet attacks. The dataset was pre-processed using pandas library for data cleaning. PCA was used in reducing the dimension of the dataset, so as to avoid data imbalance. The result of the PCA was used as input of the random forest classifier. The random forest classifier was trained using the number of estimators as 1000. The result of the model shows a promising accuracy of about 99%. The model was then saved and deployed to the web for production. The work can further be extended by building a Reinforcement Learning model in detecting Bootnet attacks.

References

[1] Pekta A., Acarman T. Botnet detection based on network flow summary and deep learning. *International Journal of Network Management*, 2018;28(6):1-15.

[2] Bou-Harb E., Debbabi M., Assi C. Big data behavioral analytics meet graph theory: on effective botnet takedowns. *IEEE Network*, 2017; 31(1):18-26.

[3] Maeda S., Kanai A., Tanimoto S., Hatashima T., Ohkubo K. A Botnet Detection Method on SDN using Deep Learning. *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019; pp. 1-6, doi: 10.1109/ICCE.2019.8662080.

[4] Kirubavathi G., Anitha R. Botnet detection via mining of traffic flow characteristics. *Computers and Electrical Engineering*, 2016; 50, 91–101.

[5] Stevanovic M., Pedersen J. M., Stevanovic M., Pedersen J. M. On the Use of Machine Learning for Identifying Botnet Network Traffic. *Journal of Cyber Security*, 2016; 4:1–32.

[6] Sriram S., Vinayakumar R., Alazab M., KP S. Network Flow based IoT Botnet Attack Detection using Deep Learning. *IEEE Conference on Computer Communications Workshops*, 2020; pp.189-194.

[7] Popoola S. I., Adebisi B., Hammoudeh M., Gui G., Gacanin H. Hybrid Deep Learning for Botnet Attack Detection in the Internet-of-Things Networks, *in IEEE Internet of Things Journal*, 2021;8(6) pp. 4944-4956.

[8] Ahmed A. A, Jabbar W. A., Sadiq A. S., Patel H. Deep Learning-Based Classification Model for Botnet Attack Detection. *Ambient Intelligent Human Computing*, 2020; 2-12.

[9] Popoola S. I., Ande R., Adebisi B., Gui G., Hammoudeh M., Jogunola O. Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT Edge Devices. *in IEEE Internet of Things Journal*, 2021; doi: 10.1109/JIOT.2021.3100755.

[10] Lingam G., Rout R. R., Somayajulu D., Das S. K. Social Botnet Community Detection: A Novel Approach based on Behavioral Similarity in Twitter Network using Deep Learning. *Proceeding of the 15th ACM Asia Conference on Computer and Communications Security*, 2020;708-718.

[11] Liu J., Liu S., Zhang S. Detection of IoT Botnet Based on Deep Learning. *Proceedings of the 38th Chinese Control Conference*, 2019; 7-30.

[12] McDermott C. D., Majdani F., Petrovski A. V. Botnet Detection in the Internet of Things using Deep Learning Approaches. *International Joint Conference on Neural Networks (IJCNN),* 2018; pp.1-8, doi: 10.1109/IJCNN.2018.8489489.

[13] Popoola S. I., Adebisi B., Ande R., Hammoudeh M., Anoh K., Atayero. SMOTE-DRNN: A Deep Learning Algorithm for Botnet Detection in the Internet-of-Things Networks. *Sensors* 2021; 21(19), 1-20.