MALICIOUS TRAFFIC DETECTION IN DNS INFRASTRUCTURE USING DECISION TREE ALGORITHM

Hazna At Thooriqoh[1], M. Naufal Azzmi H.[2], Yoga Ari Tofan[3], and Ary Mazharuddin Shiddiqi[4]
[1]Department of Informatics, Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60117
e-mail: haznaatthooriqoh@gmail.com[1], naufalazzmi@gmail.com[2], y.aritofan@gmail.com[3], ary.shiddiqi@if.its.ac.id[4]

ABSTRACT

Domain Name System (DNS) is an essential component in internet infrastructure to direct domains to IP addresses or conversely. Despite its important role in delivering internet services, attackers often use DNS as a bridge to breach a system. A DNS traffic analysis system is needed for early detection of attacks. However, the available security tools still have many shortcomings, for example broken authentication, sensitive data exposure, injection, etc. This research uses DNS analysis to develop anomaly-based techniques to detect malicious traffic on the DNS infrastructure. To do this, We look for network features that characterize DNS traffic. Features obtained will then be processed using the Decision Tree algorithm to classify incoming DNS traffic. We experimented with 2.291.024 data traffic data matches the characteristics of BotNet and normal traffic. By dividing the data into 80% training and 20% testing data, our experimental results showed high detection aacuracy (96.36%) indicating the robustness of our method.

***Keywords**: DNS traffic analysis, machine learning, decision tree.*

DETEKSI *MALICIOUS TRAFFIC* PADA INFRASTRUKTUR DNS MENGGUNAKAN *DECISION TREE ALGORITHM*

Hazna At Thooriqoh[1], M. Naufal Azzmi. H[2], Yoga Ari Tofan[3], and Ary Mazharuddin Shiddiqi[4]
[1]Departemen Teknik Informatika, Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60117
e-mail: haznaatthooriqoh@gmail.com[1], naufalazzmi@gmail.com[2], y.aritofan@gmail.com[3], ary.shiddiqi@if.its.ac.id[4]

ABSTRAK

Domain Name System (DNS) adalah komponen penting pada infrastruktur internet yang digunakan untuk mengarahkan nama sebuah domain ke ip address ataupun sebaliknya. Selain beragam kebermanfaatannya, para panyerang sering memanfaatkan DNS sebagai jembatan penyerangan. Sistem analisa traffic DNS dibutuhkan agar dapat memudahkan deteksi awal serangan yang terjadi pada internet. Namun, perangkat keamanan yang tersedia masih memiliki banyak kekurangan untuk deteksi malicious pada traffic DNS, misalnya broken authentication, sensitive data exposure, injection, dll. Pada penelitian ini kami menggunakan analisis DNS untuk mengembangkan teknik deteksi berbasis anomaly untuk mendeteksi malicious traffic pada infrastruktur DNS. Untuk melakukan ini, kami mencari network features yang menjadi ciri traffic. Kemudian fitur yang didapatkan akan diolah menggunakan algoritma klasifikasi decision tree. Kami melakukan uji coba dengan 2.291.024 data traffic yang sesuai dengan karakteristik BotNet dan traffic normal. Dengan membagi menjadi 80% data training dan 20% data testing, hasil eksperimen kami menunjukkan akurasi deteksi yang tinggi (96,36%) yang menunjukkan kehandalan metode yang diusulkan.

***Keywords**: Analisa trafik DNS, machine learning, decision tree.*

I. INTRODUCTION

DOMAIN Name System (DNS) is an internet protocol to ease human when accessing a website [1]. DNS works every time a user accesses a web page by requesting a DNS to find the IP Address of the a website address. However, there is a gap that allows attackers to embed a Botnet (a collection of machines or bots that operates a network) known as one of the most dangerous cyberattacks threats [2]. Attackers can control computers that have been attacked with Botnet because they remote by their operators through the C & C channel [3]. By becoming a bot on a Botnet network, our computers can be used by attackers for their purposes such as DDOS attacks, spreading malware, etc. Users usually do not realize that their computers have been injected with bots that become the attacker's action [4]. To avoid malicious activity, we need a method to detect bots on the internet network.

Many Botnet detection methods developed using machine learning algorithms to classify, group, and subtract dimensions to get many feature attributes [4, 5]. Ali et al. developed botnet detection using machine learning on mobile devices. Sean et al. described feature extraction as a process to retrieve the best and most accurate subset of variables representing data from existing variables. In botnet detection, the selection features process is to select the subset that best describes the bot's behavior [4]. In this research, we propose a new method for malicious traffic detection in DNS by implementing a machine learning algorithm. We use the machine learning algorithm to predict which traffic is malicious or benign. By acquiring sufficient information about the traffic data of DNS, the traffic can optimally detect early. In this research, we offer a method to detect botnets in DNS data traffic on the web. Only by using the traffic information from the DNS data can this process be done. It is beneficial to detect a botnet earlier by utilizing a decision tree algorithm.

II. Literature review

Studies on botnet detection have been carried out in many years with different methods and approaches. A study in [6] explained five general techniques used to detect DNS-based botnets, namely flow-based, anomaly-based, flux-based, DGA based, and Bot Infection Detection. Research in [7] uses a method to detect botnets by looking at network activity using the random forward classifier, by analyzing TCP, UDP, and DNS, as well as several other additional parameters. The study analyzed network traffic from over 40 infected computers used for further evaluations. Another study in [8] used a technique by making a cluster of malicious and non-malicious traffic, where this technique used a comparative approach with existing ground truth as the references. While the study in [9] developed a framework for detecting a group of hosts performing a malicious action and finding similar communication patterns between them.

Nowadays, the most popular way is to apply a machine learning technique in botnet detection[10]. Ali et al. evaluate five classification models from machine learning such as Naïve Bayes (NB), K-Nearest Neighbor (KNN), Decision Tree (DT), Multi-Layer Perceptron (MLP), and Support Vector Machine (SVM) using sample data from the Android Malware Genome Project as datasets [10]. The research aimed to find the best classification model for detecting suspicious activity on mobile devices using machine learning classifications. There are three stages in the research: data collection, feature selection and extraction, and machine learning classification. The classification process using two validation methods, namely 10-fold using K-Fold and 70% of training data and 30% of testing data using Train Test Split. The experiment results showed that the KNN produced the highest accuracy of 99.94% with K-Fold and 99.53% with train test split.

Suleiman et al. developed a method to detect botnets on android devices using a deep learning approach based on the Convolutional Neural Network (CNN) [11]. The classification process used 10-fold cross-validation. The results were compared with other classification models from recent researchers such as Naïve Bayes, SVM, RF, ANN, SL, J48, Random Tree, REPTree, and Bayes Net. The method in [11] produced the highest accuracy, with 98.9% for botnet detection on Android devices. In this study, Sulaiman et al. detecting botnets on Android devices so using CNN method is more suitable, whereas in our research we will focus on detection in the web service and the attribute classes in our dataset are true and false, so Decision Tree method we use is more suitable.

Jiaxuan detects botnets using machine learning with Artificial Neural Networks that used the dataset from 360 labs and Alexa's top one million domains [12]. The model used for the experiment was a combination of features and NB, features and ANN, N-gram and NB, and N-gram and ANN. The result with a combination using ANN (features and ANN, and N-gram and ANN) to get the best accuracy. The combination of feature and ANN produced 85% of precision, 85% of recall, and 85% of f1-score. The combination of N-gram and ANN produced 96% precision, 85% recall, and 85% f1-score.

Elaheh et al. discussed how to classify 16 features into several categories: Byte Based, Packet Based, Time Based, and Behavior-Based in the feature selection process [13]. The purpose of this feature grouping is to find out the best features in each iteration. In the first experiment, the detection results using testing data with 3 types of botnets produced an accuracy of 99% with 0.0001% false-positive. In the second experiment, by changing the testing data to seven types of botnets, the detection percentage decreased to 86% but with increased false-positive to 3%. The research concluded that the high detection percentage could be due to the type of botnet that dominates the testing data on the dataset, such as SMTP spam and UDP storm, which are easy to identify. Another cause is the similarity of data used as testing data and training data.

Saad et al. compared five machine learning algorithms to an online P2P botnet detection [14]. They divided seventeen features into two groups: flow-based and host-based features. Flow-based features such as average packet length, the total number of bytes, and so on are used to classify P2P and non-P2P traffic. In contrast, host-based features such as the number of connections over the number of destination IP addresses, the ratio of source port to destination ports, and so on identify hosts with the same form of CNC communication. Parameters used to evaluate machine learning methods are training speed, classification-speed, true detection rate, and total error rate. Experimental results showed the percentage of true detection is 90%, and the total error is lower than 7% for the SVM, ANN, and NN algorithms. While the Gaussian Based Classifiers and Naive Bayes algorithms produced a true detection percentage of 88% and a total error of more than 10%.

Mahardika et al. discussed botnets, malicious software that can perform malicious activities, such as DDoS (Distributed Denial of Services), spam, phishing, keylogging, click fraud, and stealing personal information, and important data, and so on [15]. Botnets can perform self-replicate without being noticed by users. Several botnet detection systems have been experimented using machine learning methods with a feature selection approach. Currently, the creation of data features based on network flow, DNS traffic, and content-based datasets that represent botnet behavior. The research proposed a network flow using a log connection approach to the dataset. First, they created a data model using a pair of source IP (Internet Protocol), destination IP and source port, destination port in a certain period to extract new features. Feature extraction was validated using K-Fold Cross Validation with ten experiments to examine the method’s accuracy. The experiment results showed that botnet types detection produced 98.70% of precision, 99.40% F-measure, 98.80% recall, and 98.80% accuracy for the Rule Induction Algorithm, while K-Nearest Neighbor is the most stable of all algorithms that achieve precision, recall, F-measure, and accuracy up to 98.10% and high speed (50 ms) [15].

Wielogorska et al. introduced a prototype of botnet detection system to detect botnets on a local area network with passive DNS traffic analysis, then warns network administrators of early-stage botnet presence on local area networks [16]. They proposed and tested the Naïve Bayes classification method to distinguish between benign and malicious DNS traffic flows. They use this method to extract features from DNS traffic to detect botnet-infected networks. The results of trials conducted by researchers using Weka's tools obtained a classification accuracy of 65%.

III. Basic theory

DNS translates a website address into an IP address, which will then be processed by the system when we access a website. DNS infrastructure is divided into three types of components, Client Resolver, local DNS server (LDNS), and authorative DNS server (ADNS) [17]. For the lookup process, the resolver client will request the DNS server, which has been informed repeatedly to provide the querries the DNS server needs to complete the name-to-address mapping request. Of all these processes, there can be loopholes for attackers to embed a botnet.

DNS traffic can be defined as traffic on a DNS infrastructure where there is a lot of information about querying domains from multiple hosts. Each host that requests the IP address of the intended domain has information that the admin can use for many purposes. There is a term traffic flow in DNS traffic to describe the data exchange transaction flow between the host and the DNS server. With this traffic flow, we can analyze and detect malicious traffic caused by a Botnet.

Botnets or robot networks are one of the threats to a network and data security that are currently occurring. Botnet is a malicious software that can be controlled remotely by utilizing a group of computers that already have a botnet inside. This is because the botnet can provide a platform distributed to illegal activities such as spam, phishing, password theft, and Distributed Denial of Service (DDOS) attacks. The botnet has a unique ability that distinguishes it from other malwares, i.e. the ability to control the botnet remotely by a botmaster under an infrastructure called the Command and Control (C&C) channel. Computers infected with botnets or so-called bots can cause a computer network to be blocked for its public domain and IP where data can only be circulated in a local or internal network.

Anomaly-based detection is one of many techniques used for detecting botnets. It works by finding anomaly activities based on the traffic query pattern and failed queries request. A system that has been infected with a bot will behave differently and is very easy to be detected by an anomaly-based technique. Parameters that are often used to see this strange behavior such as low TTL value, failed DNS queries, agile DNS- IP mappings, etc [6].

The approach of machine learning can be used to detect botnets and suspicious activity. Machine learning consists of two types: supervised learning and unsupervised learning. Supervised learning is a learning process with supervision for labelling data. A label is a tag added in the machine learning model. Supervised learning is often used both for multiple linear regression analysis and for logistics. Supervised learning algorithm trains a model to make predictions and classifications. Examples of supervised learning algorithms are linear regression, logistic regression, decision tree, random forest, naïve bayes, artificial neural network, support vector machine, etc.

Unsupervised learning does not use labels to predict feature/variable targets but utilizes the similarity of their attributes. If the extracted feature data's attributes and properties have similarities, the data will be grouped into a cluster. Examples of unsupervised learning are K-Nearest Neighbor, Hierarchical Clustering, DBSCAN, Fuzzy C-

Means, and so on. These steps apply for both supervised and unsupervised learnings: collecting data, feature extraction, building a learning model, and finally validating / testing the model.

In this study, we used a supervised learning algorithm (decision tree) that works by dividing large amounts of data into sub-data. The decision tree structure consists of a root, which is part of the topmost node in the tree, the branches node, which serves as a representation of the test results, and each node represents a class label. The decision tree process is carried out recursively and starts from the root node to the leaf node. In other words, the decision tree is used to dismantle a process for making complex decisions into simpler ones so that decision making will be easier and can interpret solutions to existing problems.

IV. Problem Specification

DNS traffic is often used by attackers who use DNS as a medium of attacks such as DDOS attack, spam, data theft, and access to devices connected to the internet network. An efficient way to detect the botnet and other malicious activity is required. Several approaches have been taken in the detection of botnets, but there are still some limitations. In [10], the authors propose an architecture using a machine-learning algorithm to detect malware to confront the mobile device's rapid malware growth. They used sample data from the Android Malware Genome Project as datasets. A study in [16] proposed the Naïve Bayes classification method to distinguish between benign and malicious DNS traffic flows on a local area network with passive DNS traffic analysis. This research focuses on using machine learning methods to label or classify DNS traffic for early detection of suspicious and regular traffic based on traffic characteristics.

V. Proposed Method

We propose a machine-learning algorithm to perform detection on DNS traffic. Machine learning is used to predict a traffic based on the attribute/feature. These aspects are chosen as they possess dominant roles in detection traffic. Figure 1 show the architecture diagram of this research.

We divide our proposed Botnet detection method into 4 stages: data collection, pre-processing, classification, and tuning. The first stage is data collection. Data collection is a dataset of DNS traffic that is collected and labeled as normal and malicious traffic. Then, preprocessing stage focuses on labeling data, string removal, handling missing values, data normalization, and feature extraction. Then the classification stage focuses on applying the decision tree algorithm. The last stage performs algorithm tuning to find the best setting of the algoritm parameters.

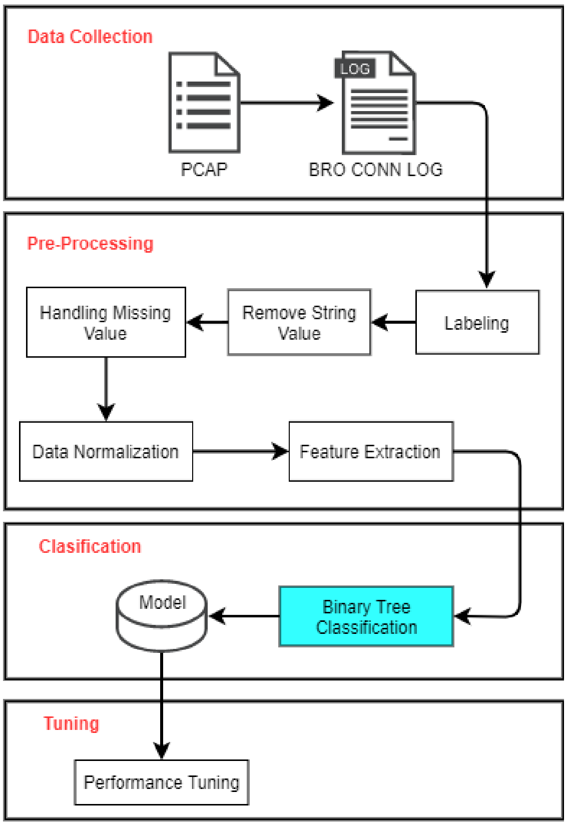

Figure 1. Proposed method.

A) Data Collection

The ISOT HTTP Botnet Dataset is a DNS traffic dataset issued by the Information Security and Object Technology (ISOT) Research Lab from Victoria University, Canada. This dataset contains malicious and benign DNS traffic. The malicious traffic dataset contains nine botnet packages and normal traffic from 19 commonly used applications. The data format that the researchers obtained was in the form of a pcap extension. The file cannot be processed directly to the preprocessing stage and must be converted into a standard connection log bro. Bro is a Network-based Intrusion Detection System (NIDS) that is also open source and popular developed by Vern Paxson in 1998. Bro can perform multi-layer analysis, behavioral monitoring, policy enforcement, etc. Bro uses anomaly-based detection that applies a statistical model to detect attacks. This method has the advantage that the detection does not stick to a predefined signature to detect new types of attacks. Besides, Bro will generate logs of each data packet based on its protocol and also supports customization of standalone log files.

B) PreProcessing

Preprocessing is the initial process in data mining used to process raw data to be processed using a specific approach. In this study, we use several preprocessing stages such as *labeling, String Value Removal, Missing Value Handling, Data Normalization*, and *Features extraction.*

1) Labeling

At this stage, the dataset labeling was carried out with two categories of malicious and normal. The malicious category refers to the identified traffic from the source of the nine botnets exploit hosts. In contrast, the normal traffic is the traffic category of 19 hosts with commonly used applications installed.

2) String Value Removal

Removing string values from this dataset is to reduce unused data in calculations. The features are timestamp, id.orig_h, id.resp_h, proto, service, duration, conn_state, history, tunnel_parents.

3) Missing Value Handling

Missing Value Handling is a method used to fill in the blanks of empty data. In this dataset, the researchers replaced the blank values with 0.

4) Data Normalization

Data normalization is the process of equalizing the range/scale on a dataset so that each feature in the dataset has the same percentage load or that all features have the same contribution in contributing their value to the mining process.

5) Feature Extraction

Feature Extraction is a technique of taking a feature from a form in which the expected value will be analyzed for the next process.

C) Classification

After the raw data were done with a preprocessing process, the data is ready to be processed at the classification stage using a decision tree. The accuracy of our classification is measured using a confusion matrix.

VI. Experiment and Analytics

We used the dataset from https://www.uvic.ca/engineering/ece/isot/datasets/botnet-ransomware/index.php [18]. There are two different datasets: a botnet dataset consisting of malicious DNS traffic generated by other botnets and a benign dataset consisting of DNS traffic generated by various known software applications. The botnet dataset contains full DNS packets of nine exploit kits collected in a virtual environment. Each bot was deployed in a Windows XP virtual machine run for several days. The virtual environment was fully monitored from the DNS server to the router (Figure 2).

The ISOT application dataset was collected from individual known (benign/normal) applications to profile their DNS behavior. This method allowed us to passively classify DNS traffic and differentiate malicious traffic vs normal traffic. The data was collected in a virtual environment. Each software application was installed on a virtual machine that was running Windows 7. The collected data is considered normal traffic since it's coming from known applications. After the data collection process is done, we divided the data into two parts, data that are identified as normal and malicious. The distribution of the dataset can be seen in Figure 3.

The data set contains 2,399,094 data with a total of 20 features, which have been divided into 2 classes: Normal and Malicious (Figure 3). Before entering the classification stage, the data will be preprocessed first. The dataset features must be extracted to reduce the number of data dimensions that the model will look for later. With this feature extraction method, new features will be obtained by combining and transforming original features [19].

Feature extraction is the main stage that must be performed in data processing. The extracted feature is a feature from reading and analyzing packet header and payload packet data. The process of extracting features from the pcap is converted to produce a CSV file extension. The dataset will be labeled, and perform calculations and feature analysis which is related to one another. This stage aims to facilitate the data normalization process. After the features are extracted, the next process is to determine the best features for the normalization process. Table I describes the attributes of the extracted files as determinants of the selected features for detection.

The transformed attribute cannot be used directly for the feature selection process. It is necessary to add new attributes to the extraction process. The addition of new attributes is done by combining the formulas from the data transformed attributes such as orig_pkts, resp_pkts, orig_ip_bytes, resp_ip_bytes. The attributes that are formed must be unique to describe the characteristics of each botnet. Table II shows the formula used to generate attributes in the extraction process.

From the total of 2,399,094 data, we divided the data into 2 portions, 80% for training data and 20% for testing data. The result of the confusion matrix can be seen in Figure 4. The classification that we created can predict malicious traffic with an accuracy of up to 96.36% from this data. The accuracy is measured using Equation 1.

$$ACC = (TP + TN) / (TP + TN + FP + FN), \tag{1}$$

where,

TP= True Positive,
TN = True Negative,
FN = False Negative,
TN = True Negative.

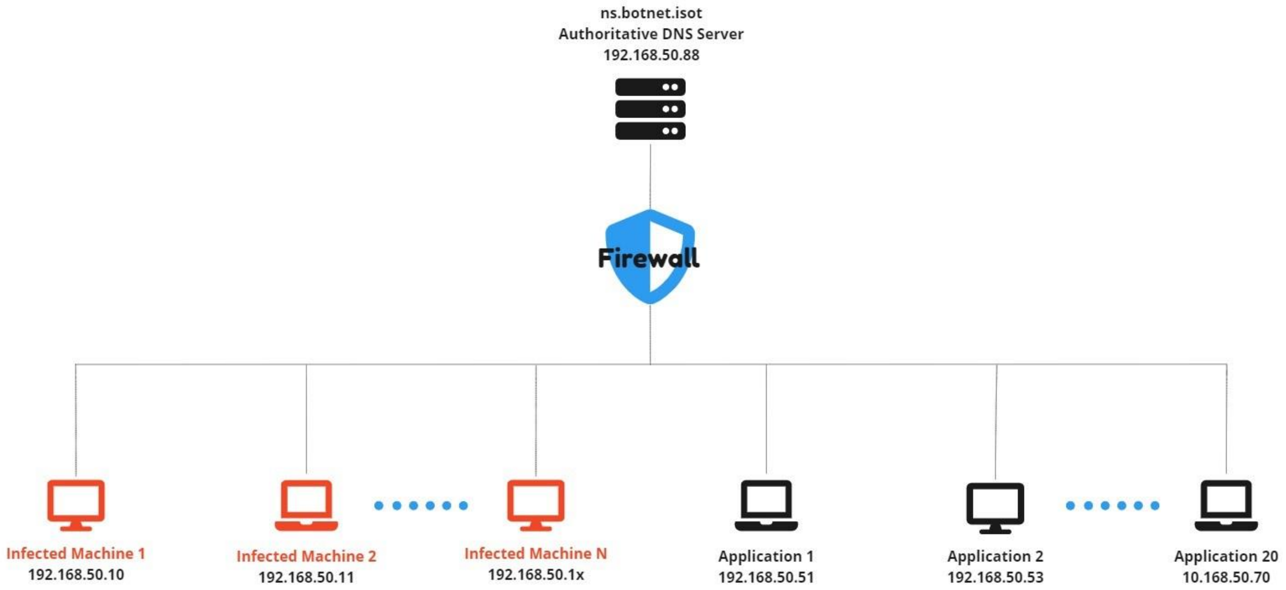

Figure 2. Dataset specification.

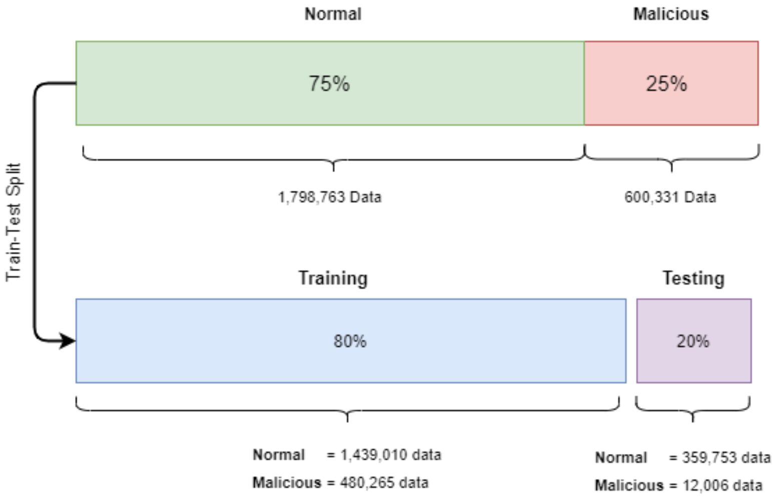

Figure 3. Data distribution analysis and splitting.

TABLE I
FEATURE EXTRACTION.

No	NAME	Description
1	id_orig_p	Source Port
2	id_resp_p	Destination Port
3	orig_bytes	The number of byte payload sent by the sender.
4	resp_bytes	The number of byte payload sent by the respondent
5	missed_byt es	Number of bytes missed in the content gap
6	orig_pkts	Number of packages sent by the sender. Only set if use_conn_size_analyzer = T.
7	orig_ip_byt es	The number of IP level bytes sent by the sender
8	resp_pkts	Number of packages sent by respondents
9	resp_ip_byt es	Number of IP level bytes sent by the respondent
10	px	Total number of packet exchange
11	nnp	Number of null packet exchange(packet with zero payload size)
12	nsp	Number of small packet echange
13	psp	Precentage of small packet exchange(have lenght between 63 and 400 bytes)
14	iopr	Rasio between number of incoming packet over number of outgoing packets
15	reconnect	Number of reconnection
16	fps	The length of the first packet
17	tbt	The length of the first packet
18	apl	Average packet length per flow
19	pps	Total number of bytes of all the packets over the total number of packets in the same flow
20	label	Traffic label name

TABLE II
FEATURE EXTRACTION FORMULAS.

No	Name	Description
1	PX	orig_pkts + resp_pkts
2	NNP	If (orig_pkts + resp_pkts = 0) then nnp = 1, else nnp = 0
3	NSP	If (orig_pkts + resp_pkts) >= 63 && (orig_pkts + resp_pkts) <= 400, then nsp = 1, else nsp = 0
4	PSP	NSP / PX
5	IOPR	Orig_pkts / resp_pkts
6	Reconnect	If history like 'Sr%', then reconnection = 1
7	FPS	Orig_ip_bytes / orig_pkts
8	TBT	Orig_ip_bytes + resp_ip_bytes
9	APL	(Orig_ip_bytes + resp_ip_bytes) / total_packet
10	DPL	The same packet length that shared by all packages in the group
11	PV	(√((1/sum data group) x ((orig_ip_bytes + resp_ip_bytes)- APL)^2))
12	BS	((orig_ip_bytes + resp_ip_bytes)x8) / duration
13	PS	(orig_pkts + resp_pkts) / duration
14	AIT	(sum(orig_pkts+resp_pkts))/(lower limit time - upper limit time)
15	PPS	(Orig_pkts + resp_pkts) / duration

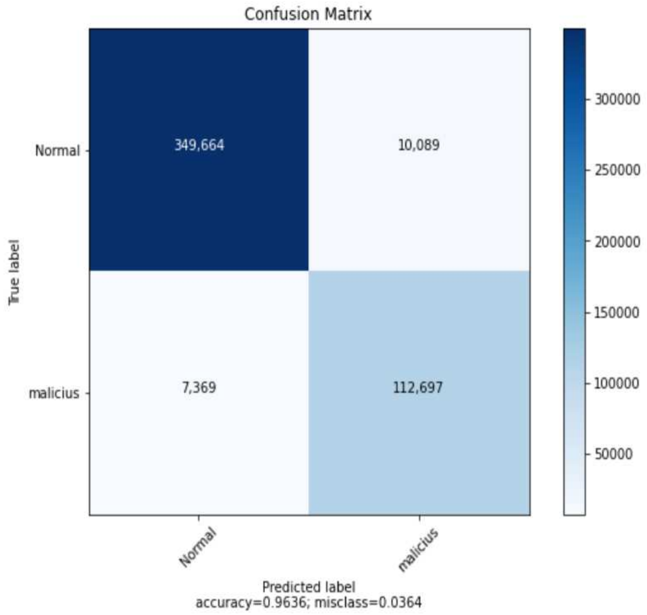

Figure 4. Confusion matrix analysis.

In this confusion matrix, it can be seen that the model we created with the decision tree can predict 349,664 data normal traffic (TP) with prediction errors on malicious data of 10,089 (FP). The False Negative (FN) score is 7,369 and True Negative (TN) is 112,697.

VII. Conclussion

We proposed a method to predict the botnet on the DNS based on the traffic. The traffic is used as input parameters to predict the malicious and benign in DNS traffic. The more data used, the more accurate the prediction. We showed that the decision tree-based botnet detection method performed well with 96% detection accuracy.

The sample data we used in this research was constrained by location, scale, and time. These types of samples are continuously occuring every time and location. Therefore, it is essential to collect the existence of a malware model at every time to analyze and improve the security system in the future. For future work, we plan to continue this research by classifying DNS traffic on real-time data.

References

[1] L. Watkins *et al.*, “Using semi-supervised machine learning to address the Big Data problem in DNS networks,” *2017 IEEE 7th Annu. Comput. Commun. Work. Conf. CCWC 2017*, no. January, 2017, doi: 10.1109/CCWC.2017.7868376.

[2] S. S. C. Silva, R. M. P. Silva, R. C. G. Pinto, and R. M. Salles, “Botnets: A survey,” *Comput. Networks*, vol. 57, no. 2, pp. 378–403, 2013, doi: 10.1016/j.comnet.2012.07.021.

[3] X. Li, J. Wang, and X. Zhang, “Botnet detection technology based on DNS,” *Futur. Internet*, vol. 9, no. 4, pp. 1–12, 2017, doi: 10.3390/fi9040055.

[4] S. Miller and C. Busby-Earle, “The role of machine learning in botnet detection,” *2016 11th Int. Conf. Internet Technol. Secur. Trans. ICITST 2016*, pp. 359–364, 2017, doi: 10.1109/ICITST.2016.7856730.

[5] X. Dong, J. Hu, and Y. Cui, “Overview of botnet detection based on machine learning,” 2018, doi: 10.1109/ICMCCE.2018.00106.

[6] A. Feizollah, N. B. Anuar, R. Salleh, F. Amalina, R. R. Ma’arof, and S. Shamshirband, “A study of machine learning classifiers for anomaly-based mobile botnet detection,” *Malaysian J. Comput. Sci.*, vol. 26, no. 4, pp. 251–265, 2013.

[7] M. Singh, M. Singh, and S. Kaur, “Issues and challenges in DNS based botnet detection: A survey,” *Comput. Secur.*, vol. 86, pp. 28–52, 2019, doi: 10.1016/j.cose.2019.05.019.

[8] M. Stevanovic and J. M. Pedersen, “An analysis of network traffic classification for botnet detection,” in *2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 2015, pp. 1–8.

[9] M. Stevanovic, J. M. Pedersen, A. D’Alconzo, S. Ruehrup, and A. Berger, “On the ground truth problem of malicious DNS traffic analysis,” *Comput. Secur.*, vol. 55, pp. 142–158, 2015.

[10] H. R. Zeidanloo, A. B. Manaf, P. Vahdani, F. Tabatabaei, and M. Zamani, “Botnet detection based on traffic monitoring,” in *ICNIT 2010 - 2010 International Conference on Networking and Information Technology*, 2010, pp. 97–101, doi: 10.1109/ICNIT.2010.5508552.

[11] S. Y. Yerima and M. K. Alzaylaee, “Mobile Botnet Detection: A Deep Learning Approach Using Convolutional Neural Networks,” *arXiv*. 2020.

[12] J. Wu, “Artificial Neural Network Based DGA Botnet Detection,” 2020, doi: 10.1088/1742-6596/1578/1/012074.

[13] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, “Towards effective feature selection in machine learning-based botnet detection approaches,” 2014, doi: 10.1109/CNS.2014.6997492.

[14] S. Saad *et al.*, “Detecting P2P botnets through network behavior analysis and machine learning,” 2011, doi: 10.1109/PST.2011.5971980.

[15] Y. M. Mahardhika, A. Sudarsono, and A. R. Barakbah, “An implementation of Botnet dataset to predict accuracy based on network flow model,” 2017, doi: 10.1109/KCIC.2017.8228455.

[16] J. Pang, R. De Prisco, J. Hendricks, B. Maggs, A. Akella, and S. Seshan, “Availability, usage, and deployment characteristics of the domain name system,” *Proc. 2004 ACM SIGCOMM Internet Meas. Conf. IMC 2004*, no. January, pp. 1–14, 2004, doi: 10.1145/1028788.1028790.

[17] A. Alenazi, I. Traore, K. Ganame, and I. Woungang, “Holistic Model for HTTP Botnet Detection Based on DNS Traffic Analysis,” 2017, doi: 10.1007/978-3-319-69155-8_1.

[18] M. Abedini *et al.*, “A generalized framework for medical image classification and recognition,” *IBM J. Res. Dev.*, vol. 59, no. 2/3, p. 1, 2015.