DDoS Attack Detection and Classification using Machine Learning Models with Real-Time Dataset Created

Harrsheetha Sasikumar

Abstract*: *Distributed Denial of Service (DDoS) attack is one of the common attack that is predominant in the cyber world. DDoS attack poses a serious threat to the internet users and affects the availability of services to legitimate users. DDOS attack is characterized by the blocking a particular service by paralyzing the victim's resources so that they cannot be used to legitimate purpose leading to server breakdown. DDoS uses networked devices into remotely controlled bots and generates attack. The proposed system detects the DDoS attack and malware with high detection accuracy using machine learning algorithms. The real time traffic is generated using virtual instances running in a private cloud. The DDoS attack is detected by considering the various SNMP parameters and classifying using machine learning technique like bagging, boosting and ensemble models. Also, the various types of malware on the networked devices are prevent from being used as a bot for DDOS attack generation.

Keywords* : *Attacks, Intrusion Detection, Malware Detection, Machine Learning Models.

I. INTRODUCTION

The main aim of the DDOS attack is to prevent the victim from utilizing its resources efficiently. DDOS attack mainly paralyzes the resources like CPU, Bandwidth as they are limited in a network. Attackers scan the network to find the machines having some vulnerability and then these machines are used as agents by the attacker. These are called zombie machines. Spoofed IP's are used by zombie machines. The design of internet gives rise to many conditions causing denial of service attacks. Security in internet is dependent on hosts. Attackers compromise the security of hosts to launch DDoS attacks and they use spoofed IP addresses making it difficult to trace attack source. There are huge number of hosts in the internet. It gives attacker huge amount of options, out of which vulnerable hosts are chosen. There are numerous types of DDOS attack possible classified according to the resource they paralyze. Depletion of bandwidth is done through flooding attacks or amplification attacks. Resource depletion is done by exploiting the vulnerability of the protocol and using the malformed packet. The attacks that we used for generation of dataset and will be predicting based on the values of SNMP are UDP Flood, ICMP Flood, TCP SYN flood, TCPSYACK Flood, Land Flood. These are the different types of flooding attack which we will be predicting using our dataset. These DDOS attack are frequent in number and there is significant case. But there is not an effective measure for predicting the possibility of these attack based on the state of the victim. So, in this work we aim to predict the possibility of an attack based on the state of various SNMP parameters. We also visualize these parameters and find effective relationship between various factors which helps in finding the possibility of an attack. The first step in generation of DDOS attack is generating an army of botnets which can be controlled remotely to generate an attack on the victim. These bots are networked devices which have been compromised by the presence of malware on them which gives control of the device to the hacker. Hackers use these devices to generate DDOS attack. This can be prevented by identifying the presence of malware on the devices. We will predict the presence of malware on the system by using machine learning algorithm. When we use both the trained classifier on their respective set of devices we will be able to predict the possibility of a DDOS attack

Table 1. Description of Attacks

Type Of Attack	Basic Description of the Attack	Target Resource
UDP	Connectionless model of UDP is exploited and bulk quantities of UDP packet are sent to the victim.	Bandwidth
ICMP	Victim is flooded with massive amounts of spoofed ICMP packets and its resource are exhausted.	Bandwidth
TCPSYN	Exploitation of three- way communication of TCP protocol. Spoofed SYN packets are send to the victim and its resources are utilized in managing session for these spoofed host creating scarcity of resources.	Network Ports
TCPSYNACK	A large amount of spoofed SYN-ACK packets is sent to the victim to exhaust an it's resources as the server tries to process this flood of requests	CPU & RAM
LAND	Spoofed SYN packets containing same IP address in host and destination is send creating an empty connection that is active until timeout is reached.	Network Ports

II. RELATED WORKS

To detect DDoS attack, SNORT IDS is used for producing alerts using DARPA dataset. The experimental results tell that PART, random forest and RIPPER is better than the other classifiers [1]. SNORT alerts are subjected to various levels such as alert fusion and alert generalization. RIPPER has less false negative rate when compared to other classifiers [2]. Application layer attacks are detected by proposing a transition matrix that observes the user browser behaviour.

Manuscript received on January 08, 2021.
Revised Manuscript received on January 15, 2021.
Manuscript published on January 30, 2021.
* Correspondence Author
Harrsheeta Sasikumar*, Computer Science and Engineering, Panimalar Engineering College, Chennai, India. Email: harrsheetha161999@gmail.com

The transition probability of the user and bot is clearly investigated [3]. A neuromorphic cognitive technique is used for detecting the network intrusions by deep learning. It provides 90% accuracy when tested on the neuromorphic chip [4]. An offline detection system is proposed to detect DDoS attack by proactive response against cyber-attacks through international collaborative exchange algorithm. It detects the network attacks based on the behaviour of the nodes [5]. Alert system is designed using artificial neural networks and support vector machines. The accuracy is compared for the rule based and threshold method techniques [6]. DoS detection in cloud using machine learning algorithms is proposed. The detection system provides greater detection accuracy compared to the existing classifiers [7]. An intelligent security system consists of the monitoring system and change point detection system that discriminates the attack and the legitimate pattern [8]. SYN flooding attacks are detected by using switch programming [9]. SDN also plays a major role in defeating the DDoS attack in cloud [10]. Feature selection which influences the effectiveness of machine learning (ML) IDS is discussed to explain the role of feature selection in the classification and training phase of ML Intrusion Detection System[11-14].

III. GENERATION OF REALTIME DATASET

The DDOS attack dataset is generated using open source cloud which is an open source software and was used to create a virtual environment for generation of Dataset. Open stack private cloud was deployed in the cloud security testbed. There were four virtual instances running on the cloud as in Figure 1. Three of the virtual instances are acting as zombies from different public and private network of the Open Stack private cloud.

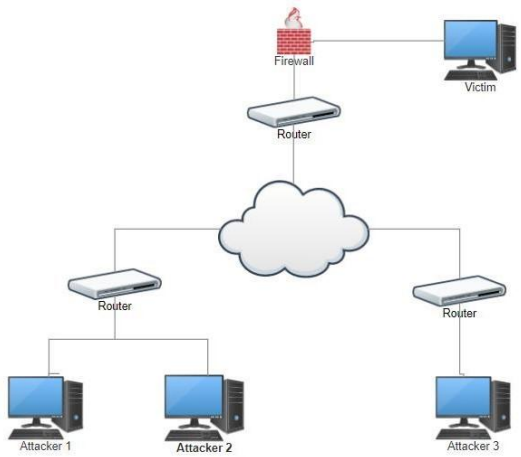


Figure 1. Basic Architecture of the System

One of the virtual instance was a victim of the attack and all the zombies targeted this victim. Traffic data was collected at the victim site using network analyser tool. The total duration of the attack was half an hour. The types of attack conducted on victim were ICMP Flood, UDP Flood, SYN Flood, SYNACK Flood, and Land Flood. SNMP MIB variables were collected at the victim side by the network analyser tool. Dataset contains SNMP variables collected during both attack and normal mode. The Dataset has ten classes based on the normal and attack mode and the class is decided by the values of the SNMP Parameters. The ten classes in the dataset are based on the normal and flood mode of each attack. The data was collected during normal mode. In normal mode most of the SNMP variables showed a linear relationship and there was no unexpected rise in number of packets. Attack generation was done using scripts and SNMP variables were collected during the attack mode. Most of the SNMP variables showed abnormal behavior when the victim was under attack. Based on the type of attack conducted the values of specific SNMP variable showed irregular trend and the linear relationship was not followed.Traffic Data was collected using Wireshark which is a network analyser tool. Files captured by the Wireshark is used to create the dataset and classes were assigned based on the value of the SNMP parameters. The SNMP parameters are discussed in brief in the table below. There were ten classes in the dataset-ICMP Normal, ICMP Flood, UDP Normal, UDP Flood, Land Normal, Land Flood, TCPSYNACK Normal, TCPSYNACK Flood, TCPSYN Normal, TCPSYN Flood. The Malware Dataset is also generated using virtual instances running in a virtual environment. The malware samples are collected from various repositories. These malware samples are executed in these virtual machines and their impact on the virtual machines is recorded by using different parameters. The parameters used by us to judge the impact of malware on the machine are debug size (Files contain an optional debug directory that indicate what form of information present and where it is. The field that is considered is size), latRVA (Relative virtual address in an image file that address of an item after it is loaded into memory., with the base address of the image subtracted from it. The RVA of an item almost always differs from its position within the file on disk), Export size (Export directory table information size), Image version (The version of the image that is used for processing the operating system), Resource size (Resource directory table has the format of offset, size and field. The field that is used is resource size), Virtual size (Virtual size occupied by the particular sample), Number of sections (The basic unit of code or data within a portable executable file), Virtual size (virtual size occupied by the particular sample). The impact on these parameters by the malware were noted. There are 4125 instances of these parameters in the dataset along with the category of the malware with which the system was affected. The different classes of malware used are virus, Trojan, adware, backdoor, muldrop, Sdbot, spam, Rbot and ransomware. This dataset will be used for training the classifier using various machine learning algorithms as in Figure 2.

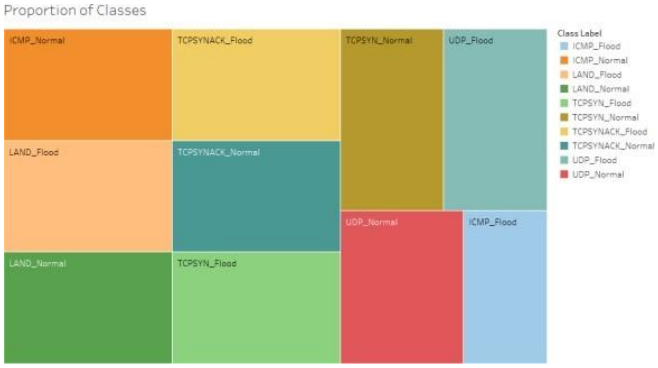


Figure 2. Class distribution for Intrusion Detection Dataset

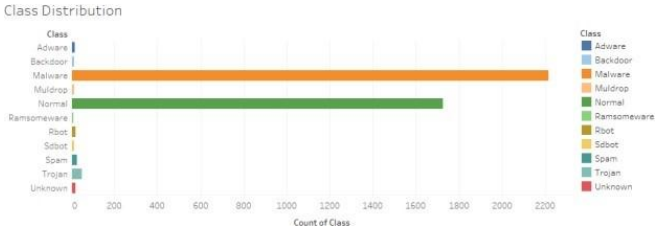


Figure 3. Class distribution for Malware Detection Dataset

Most of the classes have 250 instances in the dataset except ICMP_Flood which has 171 instances only as in Figure 3.

IV. ARCHITECTURE

The black path denotes the architecture followed for Intrusion detection and the red path if followed for Malware detection. This overlap in the architectures is due to the fact that both of these detection systems use machine learning to learn from the dataset and detect the intrusion or malware. Hence both of these architectures follow machine learning paradigm as in Figure 4. The first step in the proposed architecture is Data Capturing. The network flux at the victim's side in the cloud is monitored using network monitor tool Wireshark. The tool will track all the network traffic and store it in the raw form. The raw data from the network tool contains all details about the traffic but we need details about the specific SNMP parameters from the raw data because we will be using these SNMP parameters to build our intrusion detection system. So the raw data captured by the tool is used to extract only the SNMP parameters and projected in a tabular representation with all the proper notations so that it can be used in the later stages of the architecture. The data captured has some anomalies and null values due to various physical constraint during the data capturing phase. These values will affect the later decision making as they are not proper values and are recorded due to some technical error or some other constraint. We need to remove these values so we will pass the data through the pre- processing phase which will make the data clean and convert it into a form that can be used to train the Intrusion Detection System (IDS) classifier.

The dataset generated after pre-processing contains a large number of attributes and some of these attributes are not significant for training our IDS classifier. Training the classifier on a number of attributes is computationally high and may result in less accuracy of the system as less significant attributes will hamper its performance. So important features will be extracted from these set of features. We will use different strategies to extract important features like Principal Component Analysis (PCA) or t-distributed stochastic neighbour embedding or factor analysis techniques.

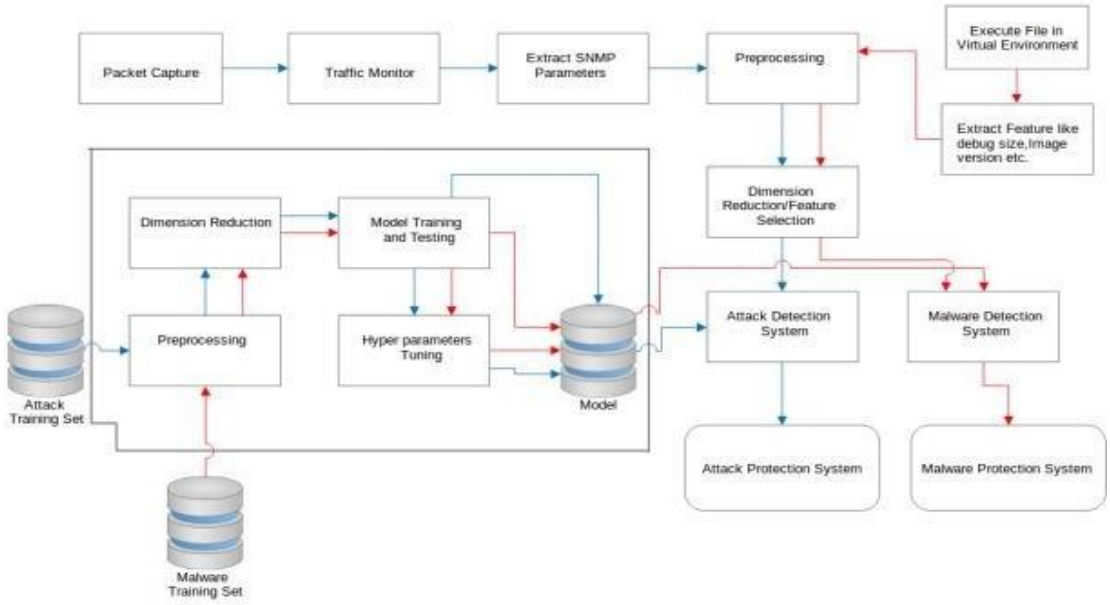


Figure 4. Block Diagram for the System

These techniques arrange the features according to their importance. PCA forms principal component vectors from the attributes and these vectors are used to decide the important features. Now these extracted features will be used to create a reduced dataset containing only the important features. This dataset after pre-processing and feature extraction is used to train the machine learning models. The machine learning models depend on various hyper parameters. These hyper parameters are responsible for the efficiency and accuracy of the model. They are used to define the model. Models with good hyper parameter give good result. Each model has a different set of parameters based on its characteristics.

We need to manually decide the best set of hyper parameters for our model. So, we will be fine tuning the model using hyper parameter tuning. After the whole training is done and the machine learning models are trained on the training datasets. We will use cross fold validation to validate our trained model on the test datasets prepared by cross fold validation. On using cross fold validation, we will get an idea about the performance of our trained model on the test data and then we will optimize these models. The Intrusion Detection System is now ready to classify the live packet stream based on the rules generated by the machine learning algorithm. These rules are generated during the training phase of the model and are used by machine to classify the new instances of data into classes. So, these set of rules will decide the class of the new instance of the data. The live stream data is collected in the same way as our training data so before feeding live data into the Intrusion Detection System (IDS), this data will again go through the above- mentioned stages in order to make the data ingestible to the classifier. Using the trained model, the classifier will classify the state of machine into one of the ten classes. Based on the class predicted by the classifier we can use other effective measured to tackle an emergency situation. If the class predicted by the classifier suggests that DDOS attack is taking place, then we can implement other methods to tackle this situation. For Malware detection, most of the architecture remains same. The difference occurs in the dataset generation where we establish a virtual environment for this setup and extract features like Debug size, image version, etc.

V. EXPERIMENT

The dataset that was analysed for this experiment on IDS to enhance cloud security was generated by using virtual instances running in a cloud security testbed. OpenStack private cloud was deployed in the cloud security testbed along with virtual instances running on them. The attack generation is done using virtual instances as zombies and the data collection is performed at the victim with network monitoring software. Throughout the experiment, the following classes were considered for classification of the state of the victim: 'ICMP_Normal', 'LAND_Normal', 'TCPSYN_Normal', 'TCPSYNACK_Normal', 'UDP_Normal', 'ICMP_Flood', 'LAND_Flood', 'TCPSYN_Flood', 'TCPSYNACK_Flood' and 'UDP_Flood'. In this dataset, there were around sixty-six attributes, which would have made the classification task more prone to overfitting due to redundant values. Also, due to the existence of highly correlated attributes which would lead to increase in training time without giving a significant improvement in the machine learning models accuracy.

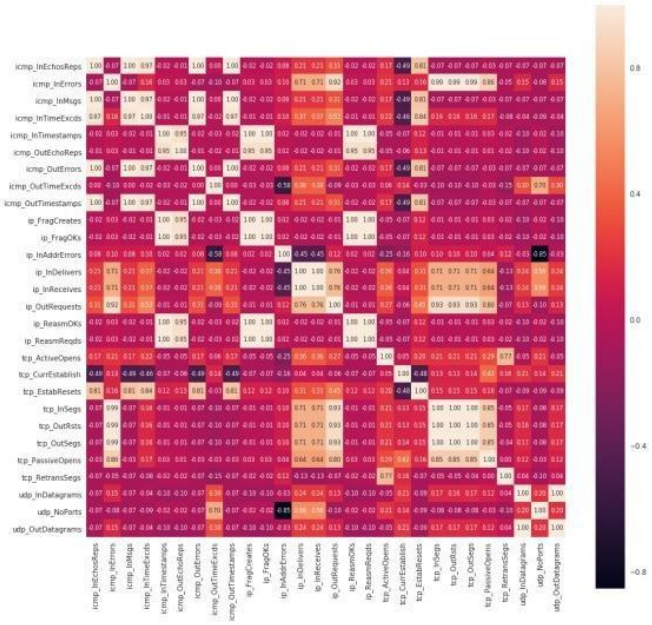


Figure 5. Correlation Heat map for Intrusion Detection Dataset

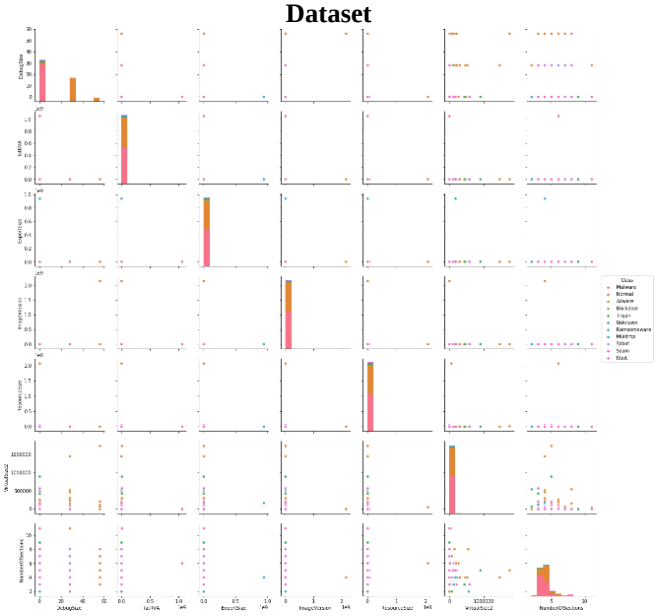


Figure 6. Pair plot for Malware Detection Dataset

Therefore, we used some dimension reduction techniques to extract the most meaningful features from this set of attributes. First, we eliminated those columns that have constant values for all the instances. All the SNMP parameters were integer values. After this, we converted all the attribute values to float as that would facilitate the machine learning calculation. Then, we used correlation matrix to get the correlation between attribute and represent highly correlated features as a single feature. This brings the features down to twenty-nine and did the same experiment on Malware Detection Dataset as in Figure [5-7].

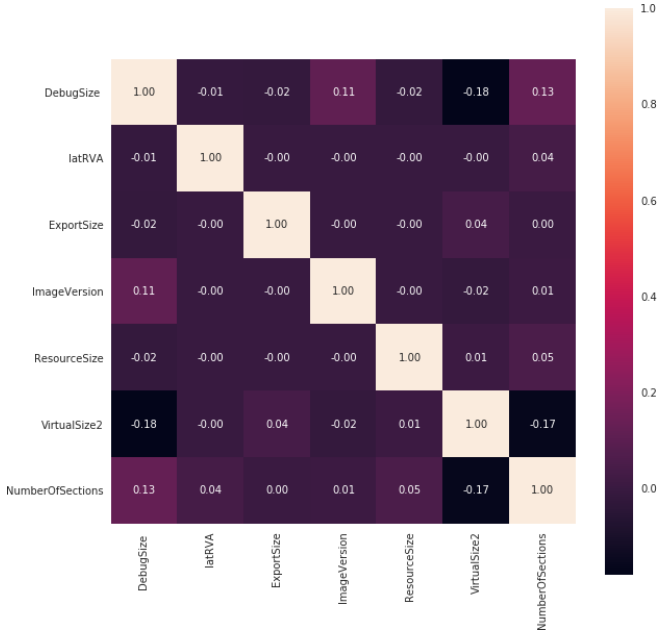


Figure 7.Correlation Heat map for Malware Detection Dataset

After this, we split the dataset into train and test data for learning and used machine learning algorithms. We used a combination of bagging and boosting models for this classification which are KNN, Xgboost, Decision Tree, Random Forest, Extra Trees and Support Vector Machine. After doing some hyper tuning for all these models we trained the models with the hyper parameters mentioned in the Table [2-6].

VI. RESULT AND ARCHITECTURE

We have visualized the Intrusion Detection dataset using Professional Data Visualization Tool: Tableau and we have inferred the following results.

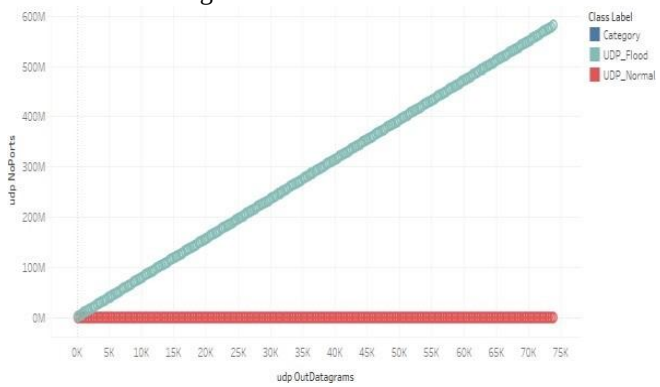


Figure 8. UDP Out_Datagram v/s No_Ports

The No_ports axis is a logarithmic axis, thus the linear relation between No_Ports and Out_Datagrams during UDP_Flood can be considered as an exponential relation in normal axis. This shows that during UDP_Normal the No_Ports and Out_Datagrams were somewhat comparative having a ratio of 1:3 whereas during UDP_Flood the ratio went up to 7300:1 as in Figure 8. The In_Datagram axis is a logarithmic axis, thus the linear relation between No Ports and In Datagrams during UDP_Flood can be considered as an exponential relation in normal axis. This shows that during UDP_Normal the In_Datagram and No_Ports were somewhat comparative having a ratio of 3:1 whereas during UDP_Flood the ratio dropped up to 0.00013 as in Figure 9

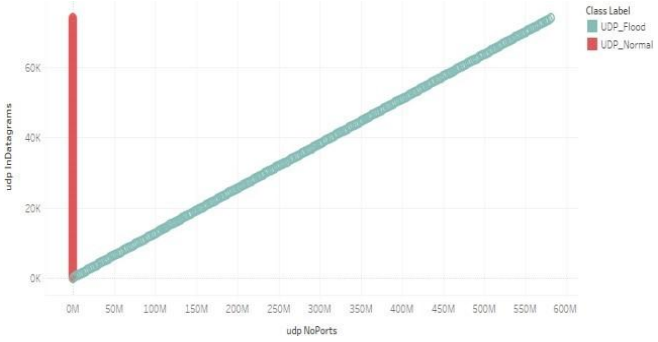


Figure 9. UDP In_Datagram v/s No_Ports

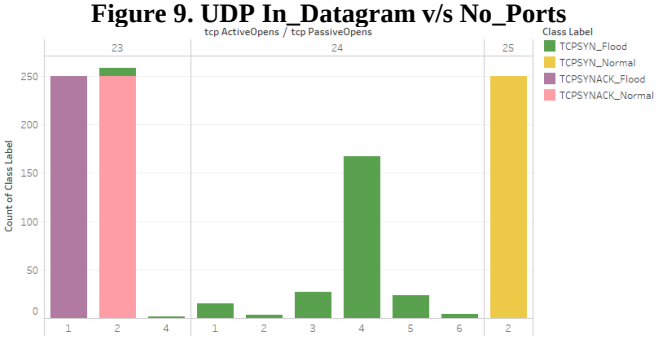


Figure 10. TCP Active and Passive Opens

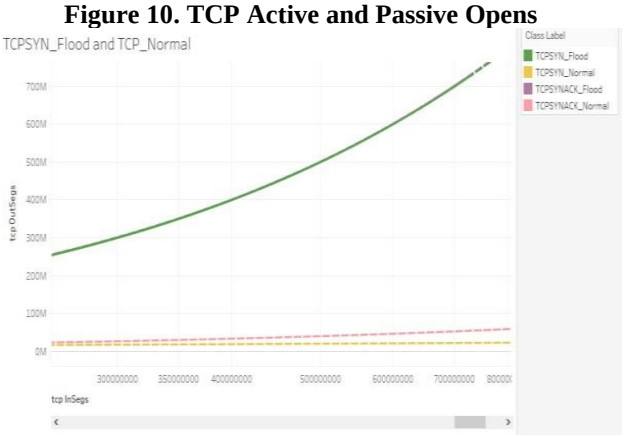


Figure 11. TCP Floods and Normals

We can infer that for TCP Active Opens = 24 we are always getting TCPSYN_Flood. Similarly, during TCPSYN_Normal we are getting 25 Active Opens. Furthermore, we can also judge the classes using the combination of Active Opens and Passive Opens as for Active opens= 23 and Passive Opens= 1 we are getting TCPSYNACK_Flood as in Figure 10. We can see that during TCP_Normal(s) the In_segs and Out_segs were somewhat comparative having a ratio of 5:2 (approx.) whereas during TCP_Flood(s) the ratio dropped up to 7300:1 (approx.) as in figure 11.

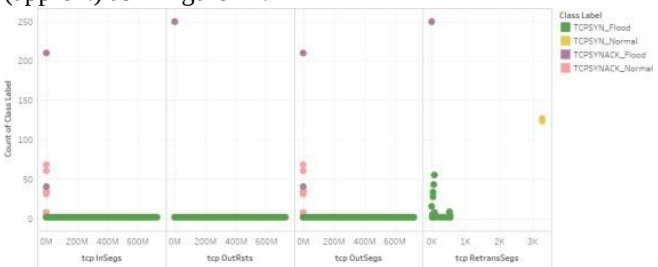


Figure 12. TCP Normal(s) and Flood(s) w.r.t. In, Out & Retrans Segs and Out Rsts

We can see that during TCPSYN_Flood the In_Segs, Out_Segs and Out_Rsts were reaching up to millions whereas while TCPSYN_Normal the Retrans_Segs was close to 3K. We can also see that the count of TCPSYNACK_Flood instances was larger than the TCPSYNACK_Normal for the same values of above mentioned parameters as in Figure 12.

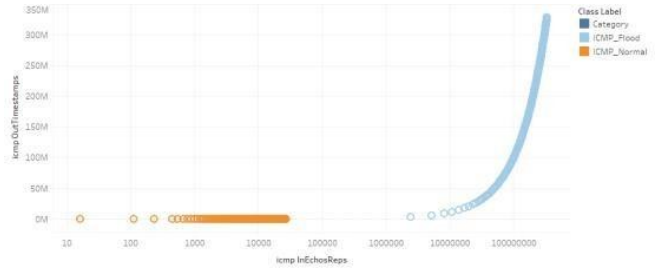


Figure 13. ICMP Flood w.r.t. OutTimeStamp and

The In_Echo_Reps axis is a logarithmic axis, thus the exponential relation between Out_Time_Stamps and In_Echo_Reps during ICMP_Flood would be a curve with much greater hike in normal axis. This shows that during ICMP_Normal the In_Echo_Reps and Out_Time_Stamps were somewhat comparative whereas during ICMP_Flood the ratio dropped went upto 5600:1 as in Figure 13.

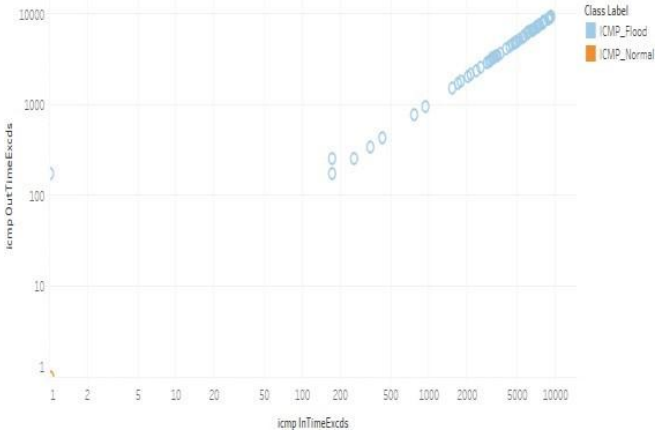
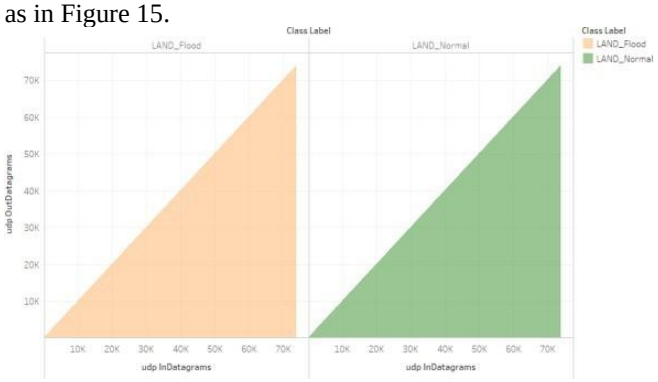
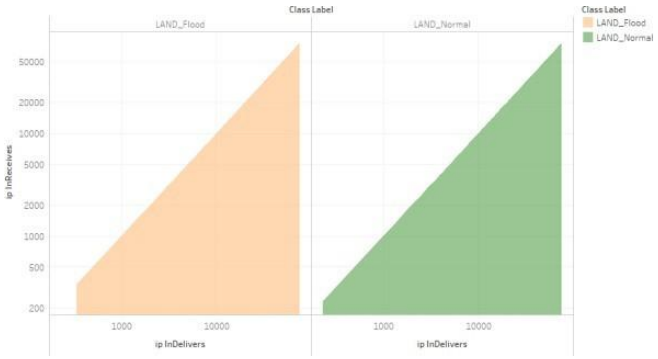


Figure 14. ICMP_Flood w.r.t. In & Out_Time_Excds

This shows that during ICMP_Normal the In and Out_Time_Excds were close to zero whereas during ICMP_Flood the values went up to 10,000 as in Figure 14. We can see that the Land Normal and Flood does not depend upon neither the UDP Datagrams nor the IP Receives and Deliveries. Therefore, this is verified that LAND Normal and Flood states can't be differentiated from the given attributes. It only depends on the Source and Destinations IP Addresses as in Figure 15.



(a)



(b)

Figure 15. Land Flood: (a)UDP IN and OUT Datagrams (b)IP Receives and Deliveries

Figure 16. ICMP Flood w.r.t. In_Msgs and Out_Errors

This shows that during ICMP_Normal the In_Msgs and Out_Errors were within thousands whereas during ICMP_Flood the In_Msgs were close to millions as in Figure 16. After this visualization, we trained and tested the already mentioned machine learning models on the Intrusion dataset and we are getting a state-of-the-art accuracy of 99.2% from SVM (linear kernel). Furthermore, the following tables show a comparative analysis of the prediction done by these algorithms for Intrusion Detection dataset with different hyper-parameters as in Table [2-7].

Table 2: KNN Results

KNN	Accuracy	Precision	Recall	F-Score
n=5	0.657	0.653	0.645	0.646
n=10	0.629	0.627	0.617	0.620
n=15	0.645	0.643	0.632	0.634
n=20	0.660	0.662	0.647	0.653
n=25	0.647	0.649	0.635	0.640
n=30	0.645	0.652	0.632	0.638

Table 3. XG Boost Results

Xg Boost	Accuracy	Precision	Recall	F-score
0.001 , 5 , 100	0.990	0.990	0.989	0.989
0.1 , 5 , 100	0.990	0.989	0.989	0.989
0.01 , 3 , 100	0.988	0.988	0.987	0.988
0.01 , 5 , 10	0.990	0.990	0.989	0.989
0.01 , 5 , 50	0.990	0.990	0.989	0.989
0.01 , 5 , 100	0.990	0.990	0.989	0.989

Table 4. Decision Tree Results

Decision Tree	Accuracy	Precision	Recall	F-Score
max_depth=2	0.272	0.157	0.3	0.184
max_depth=5	0.781	0.725	0.789	0.741
max_depth=7	0.993	0.992	0.992	0.992
max_depth=10	0.991	0.991	0.990	0.991
max_depth=12	0.991	0.991	0.990	0.991
max_depth=12	0.991	0.991	0.990	0.991

Table 5. Random Forest Results

Random Forest	Accuracy	Precision	Recall	F-score
depth=5, n = 100, features=7	0.987	0.987	0.986	0.986
depth=10, n = 100, features=7	0.991	0.991	0.990	0.991
depth=5, n =50, features=5	0.987	0.987	0.986	0.986
depth=5, n = 100, features=5	0.988	0.988	0.987	0.988
depth=10, n = 100, features=5	0.991	0.991	0.990	0.991
depth=10, n = 100, features=7	0.987	0.987	0.986	0.986

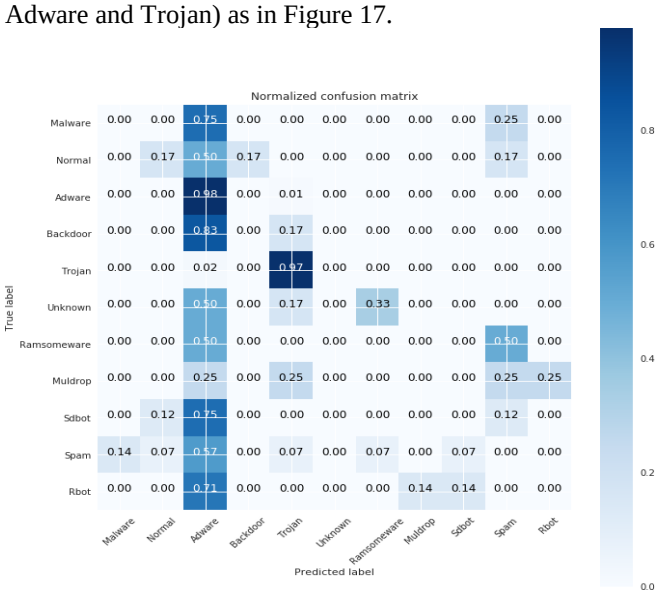
Table 6. Extra Trees

Extra Trees	Accuracy	Precision	Recall	F-score
depth=5, n = 100, features=7	0.883	0.833	0.880	0.847
depth=10, n = 100, features=7	0.987	0.987	0.985	0.986
depth=5, n =50, features=5	0.770	0.724	0.759	0.719
depth=5, n = 100, features=5	0.793	0.727	0.786	0.738
depth=10, n = 100, features=5	0.795	0.726	0.787	0.738
depth=10, n = 100, features=7	0.793	0.727	0.786	0.738

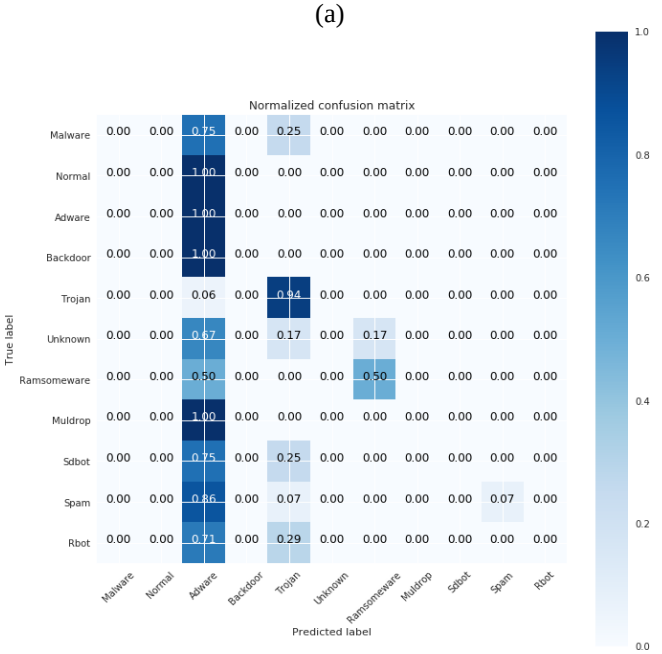
Table 7. Performance Comparison for Intrusion Dataset

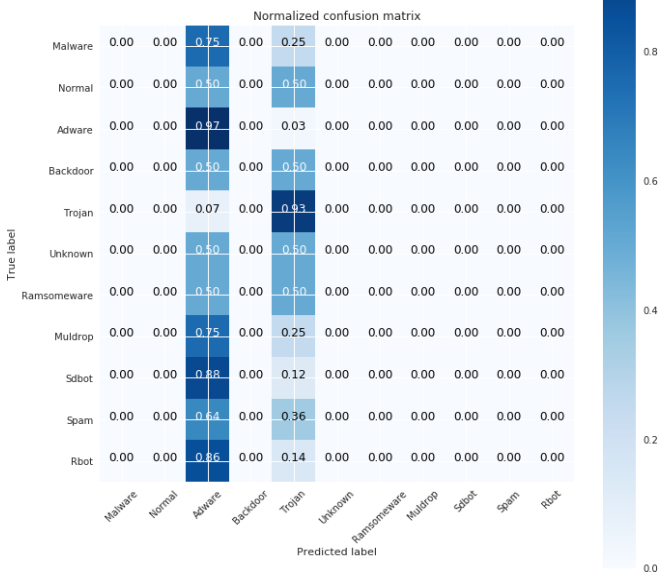
Model	Accuracy	Precision	Recall	F - Score
K Nearest Neighbours	0.613	0.647	0.645	0.641
Xgboost	0.958	0.994	0.995	0.994
Decision Tree	0.990	0.991	0.991	0.991
Random Forest	0.983	0.985	0.985	0.985
Extra Trees	0.858	0.840	0.878	0.848
SVM (linear)	0.991	0.992	0.992	0.992

After the Intrusion dataset, we trained and tested the already mentioned machine learning models on the Malware dataset and we are getting an accuracy of 92.9% from Extra Trees. Furthermore, the following figure and table shows a comparative analysis of the prediction done by these algorithms for Intrusion Detection dataset and their confusion matrices which shows that these models can classify the dataset in the following classes (majorly in Adware and Trojan) as in Figure 17.

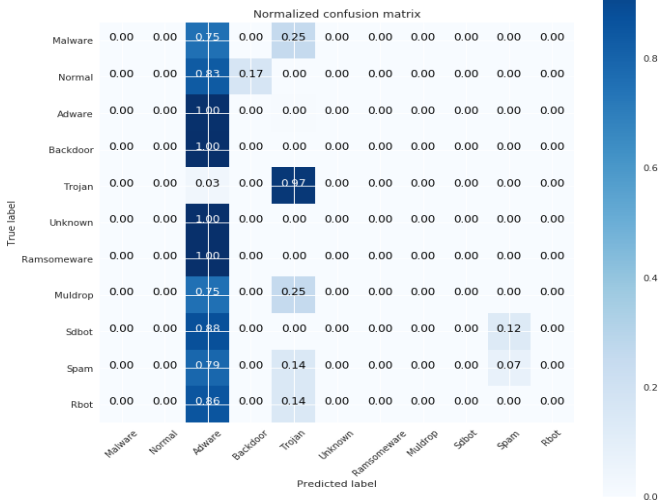


(a)

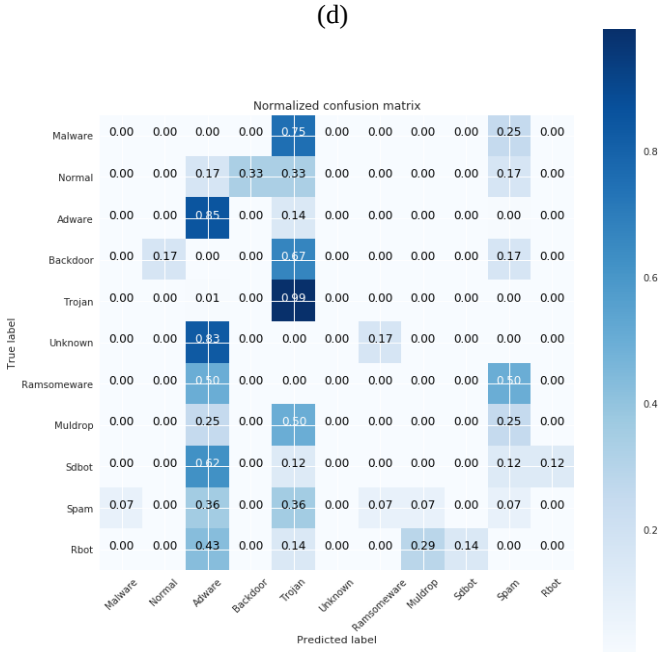




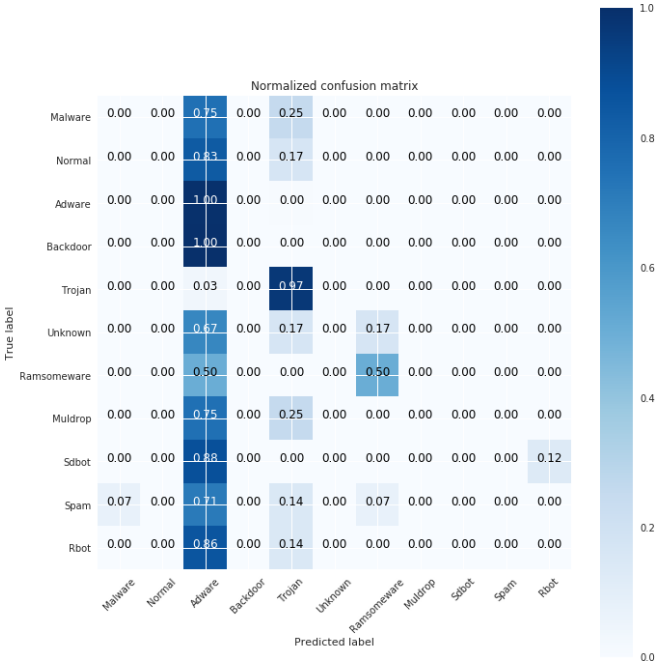
(c)



(d)



(e)



(f)

Figure 17. Confusion Matrix for Malware Detection Dataset: (a) Decision Tree (b) Extra Trees (c) KNN (d) Random Forest (e) SVM (RBF kernel) (f) Xgboost

Table 8. Performance Comparison for Malware Dataset

Model	Precision	Recall	F-score	Accuracy
KNN	0.1657	0.1727	0.1690	0.908
Xgboost	0.2024	0.2239	0.2115	0.938
Decision Tree	0.1919	0.1927	0.1921	0.932
Random Forest	0.2174	0.1849	0.1864	0.938
Extra Trees	0.3069	0.2283	0.230	0.929
SVM (RBF)	0.1712	0.1742	0.1715	0.873

VII. CONCLUSION

In this work we used the open source cloud and performed a DDOS attack on the open source cloud. The datasets are generated using virtual instances running in a cloud security testbed. We then visualized the data using Tableau: Data Visualization Tool and inferred a lot of results. Then we performed analysis to predict the possibility of a DDOS attack based on the instances of various SNMP parameters using various machine learning technique like bagging, boosting and ensemble models and achieved state-of-the-art performance using SVM (linear kernel). Decision Tree also performed really well. We also used various machine leaning models on the Malware Dataset for the malware detection system and achieved 92.9% accuracy using Extra Trees. Hence, using the proposed architecture, we can detect and prevent any upcoming DDoS Attack.

REFERENCES

1. T. Subbulakshm and S. M. Shalinie, “Real Time Alert Classification System for Minimizing the False Alarm rate”, International Journal of Information and Communication Technologies, vol.2, pp. 117-122, 2009.
2. T. Subbulakshmi and S. M. Shalinie, “Real Time Classification and Clustering of IDS alerts using Machine Learning Algorithms”, International Journal of Artificial Intelligence & Applications, vol.1, pp. 1-9, 2010.
3. Y. Chengxu and Z. Kesong, “Detection of application layer distributed denial of service”, In: Proc. of International Conference on Computer Science and Network Technology: 6181964, 2011.
4. A.M.Zahangir and M. T Tarek, “Network Intrusion Detection for Cyber Security on Neuromorphic Computing System”, In: Proc. of International Joint Conference" on Neural Networks (IJCNN): 7966339, 2017.
5. F.Yaokai, H. Yoshiaki and S. Kouichi, “A Detection System for Distributed DoS Attacks Based on Automatic Extraction of Normal Mode and Its Performance Evaluation”, In: Proc of the 10th International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage, SpaCCS- Guangzhou, China 461-473. 10.1007/978-3-319-72389-1_37, 2017.
6. T. Subbulakshmi and S. M. Shalinie, “Detection and Classification of Intrusions using Machine Learning Algorithms”, European Journal of Scientific Research, vol.47, pp. 334-346, 2010
7. Z.He, T.Zhang and R. B. Lee, “Machine Learning Based DDoS Attack Detection from Source Side in Cloud”, IEEE, 10.1109/CSCloud.2017.58, 2017.
8. S.Murat, C. Ali and S. Bulent, “An Intelligent Cyber Security System Against DDoS Attacks in SIP Networks. Computer Networks”. 10.1016/j.comnet.2018.02.025, 2018.
9. M. Safaa, M. Oussama and H. Abdelkrim, “Scalable and Dynamic Network Intrusion Detection and Prevention System”. 318-328. 10.1007/978-3-319-76354-5_29, 2018
10. S. Nasrin, C. Naveen and P. Wei & Alhadad, Rabei,” Survey on SDN based network intrusion detection system using machine learning approaches”, 2018.
11. N. Udayakumar, V. J. Saglani, A. V. Cupta and T. Subbulakshmi, "Malware Classification Using Machine Learning Algorithms," 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, 2018, pp. 1-9, doi: 10.1109/ICOEI.2018.8553780.
12. N. Udayakumar, S. Anandaselvi and T. Subbulakshmi, "Dynamic malware analysis using machine learning algorithm," 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, 2017, pp. 795-800, doi: 10.1109/ISS1.2017.8389286.
13. Udayakumar, N. "Subbulakshmi. T, Ayush Mishra, Shivang Mishra and Puneet Jain, Malware Category Prediction using KNN Classifier and SVM Classifiers." International Journal of Mechanical Engineering and Technology 10.2 (2019): 787-797.
14. N. Udayakumar, A. Khera, L. Suri, C. Gupta and T. Subbulakshmi, "Bandwidth Analysis of File Transfer Protocol," 2018 International Conference on Communication and Signal Processing (ICCSP), Chennai, 2018, pp. 0791-0795, doi: 10.1109/ICCSP.2018.8524583.

AUTHORS PROFILE

Harrsheetha Sasikumar is a final year student currently pursuing her Bachelors in Computer Science Engineering at Anna University. She did her 10th grade at S.B.O.A. School and Junior College and graduated with an excellent score of 10 CGPA and she graduated senior secondary education (12th grade) from DAV Matriculation School. She is a network security, data science and machine learning enthusiast. She did a project on “ heart disease prediction” using random forest algorithm during her third year at college. She is a former software engineering intern at Mindgate Solutions Pvt. Ltd. She is an active member of the following technical clubs at her college- Mozilla club, cyber security club and Study monk. She is also a member of NCC(National Cadet Corps) at her college. She is an aspiring data scientist with strong foundations in Python, dot net, Java, neural networks, Statistics and ML.