COMP8325

Applications of Artificial Intelligence in Cyber Security

Assignment-2

Final Report

Port scanning/Probe attack detection using Machine learning

Group members:

Mohan Kumar Padmanabha - 45758875

Shiva Prasad Aggayala - 46233865

Srisai Ravi Teja Marupeddi - 45756546

Table of Contents

Abstract

In recent years, the number of attacks against computer networks and their components has increased. To defend against these attacks, several intrusion detection techniques have been proposed. Intrusion Detection systems (IDS) is a system that gathers and analyses data from the network to determine various attacks performed against the network component [1]. Preventing any form of cyber-attack is integral because a small or single attack may compromise the security of computer and network systems [2]. The detection of such assaults is completely reliant on them. Machine learning techniques are being used in cybersecurity in greater numbers than ever before. We are going to examine a probing data set that contains network activity during an attack and normal network activity and apply Machine learning algorithms on this data set. In this paper, we employed some machine learning techniques to detect the probing attack, such as Isolation forest, KNN classifier, Random Forest, and SVM.

Introduction

The internet has become an integral part of people's lives all across the world. Businesses, in particular, make extensive use of the internet as part of their overall business strategy. Furthermore, internet technologies, such as the web and email, are being used to generate income and communicate with consumers. Important and confidential information is frequently stored on these devices that can be accessed via the internet [3]. The fast growth of computer networks, particularly the internet, has resulted in several security issues. The number of attacks on networks in recent years has dramatically increased. As a result, network security and resources are important when it comes to probing attacks.

Attacks such as DOS (Denial of service), U2R (User to root attacks), and R2U (Remote to the user) are based on these attacks. Each attack has its own behavior of using network resources in order to meet the attacker's aim of rendering the network unavailable for its users [2]. As a result, protecting a system against such attacks is crucial. Here the focus is on probing if this type of attack occurs when an attacker scans network devices to find flaws in topology design or open ports, then using them in the future for unauthorized access to sensitive information[1]. These attacks do not participate in active activities, although most of their time is spent passively, such as determining which machines are operational or which services are used by the users within the network. Most intruders use a variety of methods to learn about bugs, parameters, or techniques that might be useful to gain access to resources. The primary objective of a probing attack is to know about the company resources that are currently in use of the internet or its information system-based activities [2]. A single probing attack could result in a significant loss of a company that serves as the backbone of many networks [1]. As a result, ensuring

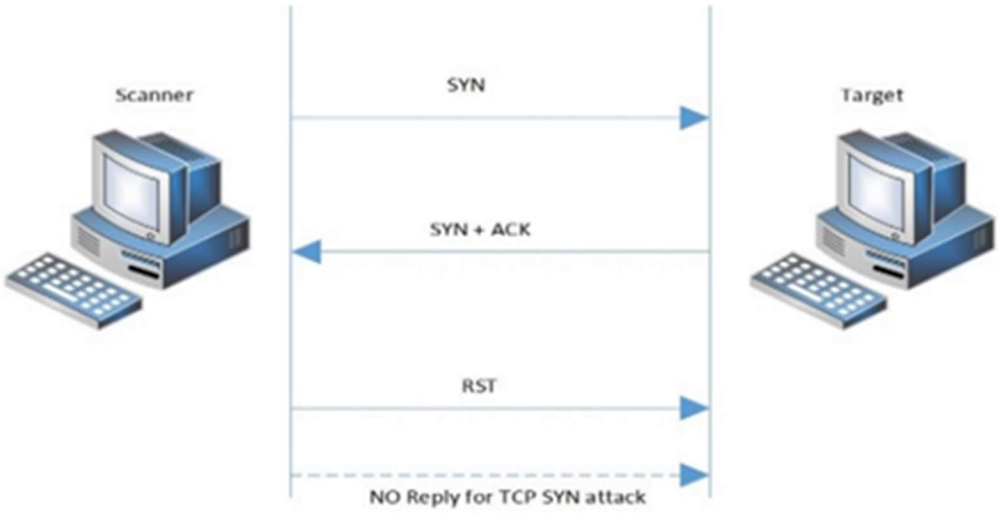
company assets (servers, etc.) are essential from these attacks. In this report, we presented an approach that determines the probing attack using machine learning techniques such as Isolation forest, KNN classifier, Random Forest, and SVM. We also present the visualizations using confusion matrix and AUC-ROC curve in the supervised machine learning algorithms.

Port scanning or probing attack

A probe attack scans the network to determine the vulnerabilities when a port is being scanned; it uses scanning tools [4] to identify open ports in the target.

The goal of port scanning is to find insecure areas of a target resource where exploitation might be feasible. The anatomy of a typical threat comprises five steps: reconnaissance, enumeration, penetration, exfiltration, and sanitation[4].

During the reconnaissance phase, an attacker obtains required information such as IP scheme, data center locations, and target profile[4]. Port scanning is involved in the enumeration step as an early step. NMAP is the most often used port scanning tool. Scanning for ports that handle Transmission Control Protocol/Internet Protocol (TCP/IP) communication is a common example. The synchronization (SYN) signal is sent from a scanner to the intended target if the target responds with Synchronization & Coordination. The acknowledgment (SYN+ACK) signal indicates that the message has been received. This indicates the port is open[4]. The scanner may now send a rest (RST) signal to terminate the connection after finding that the port is open. But on the other side, leaving the connection open may lead to denial of service, also called a TCP SYN attack[4].



Probing attacks can steal sensitive information from computers or networks, which the intruder can then misuse. These attacks have the potential to do considerable damage. Many organizations have lost time and money [1] as a result of this.

Different types of probing attacks

Ipsweep: It probes the network to find out what services are accessible. The first attacker discovers a machine that might be used to attack[1].

Portsweep: It probes a host to see what services are made available on that host. If a service is well-known on the system, a network attacker can readily target it[1].

Nmap: It's a comprehensive and adaptable tool for scanning a network in a random or sequential manner. As a result, attackers often use these tools to scan network settings. This might facilitate them in their attack on the system[1].

Satan: It's a network administration tool that collects data. An attacker might make use of this data[1].

Literature Review

Machine Learning for Cyber Security Applications

Machine Learning has significantly increased the efficiency of solving cybersecurity issues, including intrusion detection by analyzing networks and tackling malicious activities. Machine Learning models require elegant theoretical and methodological handling to deal with sophisticated cybercrimes. The quality of the datasets used to train the machine learning models is the key to achieving better predictions and responding to attacks. But, companies, in general, do not share their data because it is sensitive, especially which is related to cybersecurity. But, we can overcome this problem by removing sensitive data from the datasets using anonymization and share it with the community.[5]

Machine learning models track the behavior of the traffic flow and other metrics like changes in user authentication and authorization to prevent insider threats for an organisation and respond to them accordingly in an automated way. It is equally important to secure the company from outside threats as well as insider threats. Some of the other significant areas of cybersecurity where Machine learning can be implemented are:[6]

- Detection of Spam emails and Phishing links
- Malware identification
- Anomaly Detection
- DDOS attack detection
- Detection of Software vulnerabilities

We'll briefly describe Anomaly detection and detection of software vulnerabilities from the above-mentioned challenges.

Security flaws or vulnerabilities in a software system are leveraged by an attacker if they are not fixed in a timely manner which could compromise the entire system or network. Malicious code injections can be done by the attackers, which escalates their privileges in the network and eventually lead to a data breach or any other attacks. Machine learning methods like anomaly detection and pattern recognition can be implemented in the syntax of the code and analyze patterns in the large codebases for vulnerability detection.[6]

Anomaly detection includes the implementation of API usage patterns, lack of access controls and input validation, etc., and classification is done by using techniques like k-nearest neighbors.

➢ Vulnerable lines of the code are analyzed by machine learning algorithms in the Pattern recognition using features like API calls and system calls which are invoked by syntax trees, etc. The classification of vulnerabilities is done by using Machine learning techniques like random forests, logistic regression, neural networks, etc.[6]

Anomaly detection involves finding unexpected or abnormal patterns in a dataset which are known as anomalies or outliers. Machine Learning techniques can be implemented to detect these anomaly patterns and categorize them into various intrusions and attacks. Although, not all anomalies are actual attacks for which different algorithms are used to detect and separate them from actual malicious attacks. They are:[6]

➢ K-means clustering in which objects are grouped on the feature vector basis into disjoint clusters. In the case of network traffic, anomalous traffic in a training dataset is differentiated from the normal traffic by using the Network Data Mining approach along with a k-means clustering algorithm.
➢ EM clustering algorithm checks the weight of the probability of membership and assigns the objects to a cluster and the new mean of this cluster is computed. This algorithm is proven to outperform the K-means algorithm.
➢ A hybrid approach combining supervised and unsupervised learning is preferred to overcome the problems that arise in individual approaches. This is achieved by combining entropies of network features which results in high accuracy detection.

Adversarial Machine Learning

Adversarial Machine Learning is a technique that attempts to fool the machine learning models into malfunction by feeding deceptive data to them. Automation of various Cybersecurity tasks can be done extensively by using various Machine Learning techniques in the real world. Malicious entities called Adversaries may deliberately manipulate the training data of the models to exploit specific vulnerabilities to perform certain Cyber-attacks. Adversarial Learning is characterized along three dimensions which are influence, specificity, and security violation. We focus on influence as it is the most researched dimension of Adversarial Learning, which is most relevant for probing attacks, and it specifies two types of adversarial attacks, namely causative and exploratory (also known as probing). To develop appropriate learning strategies, the adversary is characterized based on their behavior as they try to discover vulnerabilities by observing the output of learner's classifiers of various data in the exploratory attacks.[7]

Deep Learning or Machine Learning algorithms can be implemented in Intrusion Detection Systems to tackle security-critical issues like port scans and DDOS attacks, which evade the IDS consequently. They can be classified into Supervised Learning and Unsupervised Learning where port scanning falls under the category of Supervised Learning[8]. Port scans are usually flagged by the IDS as they log the sender's IP address. Apart from that, the Intrusion Detection system focuses on attack patterns that require knowledge of domain and network behavior analysis. This increases the dependency on domain knowledge and large sets of data collection and analysis. The Adversaries can attack training and testing datasets in an effort to reduce the accuracy and effectiveness of these models.[9]

One of the experimental strategies that the IDS systems can implement is, the SDN switch deployed in the network system monitors Open-Flow Control protocol and initiates Packet-In message to the controller when it flags the arriving packets which do not match the Flow Table. The controller then extracts pre-defined features from the deep learning detecting system like packet length, UDP & TCP packet count, port counts, ICMP packet count, etc. Static messages are used by the controller to monitor the switch frequently of its ports and send this report along with the Packet-In messages to the deep learning models for processing and training them. It then performs an attack-detecting process to determine whether the packets are attack messages based on its knowledge gained previously.[9]

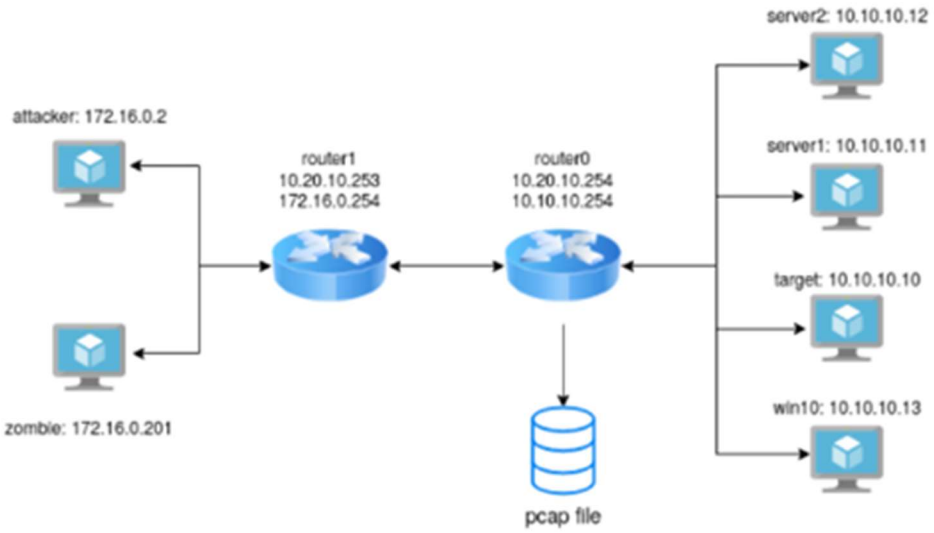
Another approach is to use a combination of Supervised and Unsupervised Learning methods known as Deep Belief Networks, which detect signatures of port scanning attacks by processing network data. It is a multilayer neural model and its network layers

generate output with clear expression of data from the same class of the complex structured data. It has a two-phase training algorithm where the first phase is unsupervised learning with unrestricted training mode based on Restricted Boltzmann Machine (RBM) and the second phase which is also known as refining phase is Supervised Learning process.[10]

Data set description

Probing data set and its anomaly detection using the different anomaly detection methodologies.

Background on the data set: This data set is a popular probing data set that has been quoted multiple times in multiple places. Out of all the given data sets, this data set was clearer to us and more understandable as we have a strong background in Networking. On Inspecting the data set, we can see that the file has the network capture pcap file transcribed into a csv file. This is a very systematic data set as it was generated by the network devices which work on similar protocols, and they have to be in proper order to work in real life, so this data set was very precise for the data analysis task. The data which is being used in our project is the MAWIlab dataset, and the malicious data set is generated by the owner of the dataset using the below network architecture and is targeted on the server with the IP address 10.10.10.10. The pcap file generated on the internal router (router0) is the dataset that is being used in this project by the machine learning models to learn the features of the malicious network traffic.



Code Implementation

Below are the steps that have been followed to achieve the anomaly detection on the probing data set that has been provided.

Step 1: Import all the libraries required for the execution of the code and put it in one place, which is the first 2-3 blocks of the code.

Step 2: As this code is being executed in google colab, all the data sets have been imported from the drive for ease of execution.

Step3: The data set that has been selected is the probing attack data set. This data set has two sub data sets. One of which is the malicious data set, and the other is the normal data set. We are merging both the data sets to form one big data set, and the data in the normal data set has been labeled as normal_traffic for supervised learning models to get a deeper understanding of the background traffic or the normal traffic.

Step 4: Apart from the data set, there is one more file that has labels that can be attributed to the data set as labels that are provided in GitHub. We are using this label for the supervised learning models in the code.

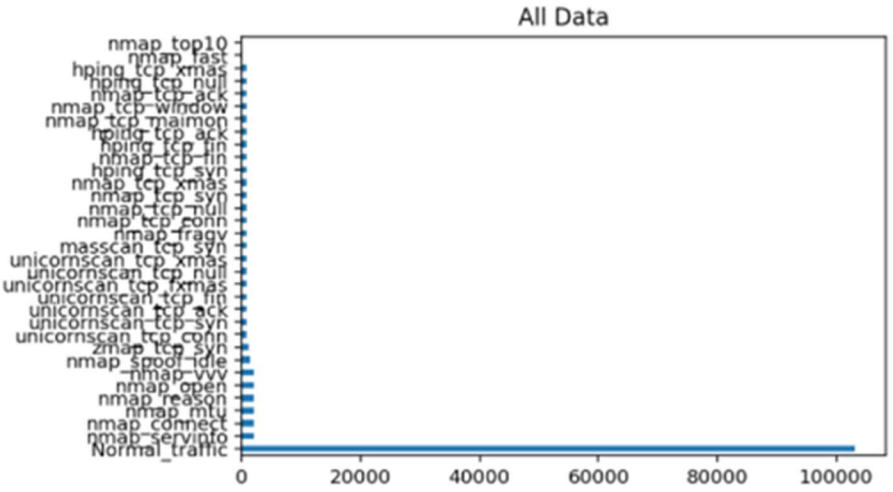
Step 5: After merging the file which has labels with the original data set, we get a file that has all the labels.

Step 6: We are then loading this data set for modification using panda's library. After loading the data sets in the code, we are shuffling the data to get uniformity and randomness in the data.

	ip.src	frame_info.encap_type	frame_info.time	frame_info.time_epoch	frame_info.number	frame_info.len	frame_info.cap_len	eth.type	ip.version	ip.hdr_len	ip.to
13848	172.16.0.16	1	Dec 31, 1969 21:35:25.379795000 -03	2.125380e+03	26890	78	78	0x00000800	4.0	20.0	NaN
135151	92.192.62.116	1	Nov 21, 2019 02:01:14.036651000 -03	1.574312e+09	193348	54	54	0x00000800	4.0	20.0	NaN
78582	203.180.205.36	1	Nov 21, 2019 02:00:30.787588000 -03	1.574312e+09	81261	1436	66	0x00000800	4.0	20.0	NaN
76409	13.35.206.114	1	Nov 21, 2019 02:00:29.319744000 -03	1.574312e+09	77018	1440	66	0x00000800	4.0	20.0	NaN
50287	88.162.108.220	1	Nov 21, 2019 02:00:10.229634000 -03	1.574312e+09	25741	58	58	0x00000800	4.0	20.0	NaN

Step 7: After this, we are looking at the data set, and we are reducing the features of the data set. The features which remain unchanged throughout the data set and have null values are removed from the data frame to enhance the execution speed.

Step 8: After making adjustments to the data frame, we are plotting a bar graph to show the distribution of the labels throughout the data. From the graph, we can see that the label named nmaptop10 has the least number of appearances in the data, and the highest number of packets is from the Normal_Traffic from the data set, which consists of normal data or background data. This is followed by labels nmap_servinfo, nmap_connect, nmap_reason, nmap_mtu, etc, as shown in the graph below. We also printed the label counts to get the exact counts of appearances in the data. The screenshot is attached below.




```
Normal_traffic          103094
nmap_servinfo             2016
nmap_connect              2008
nmap_mtu                  2006
nmap_reason               2006
nmap_open                 2006
nmap_vvv                  2002
nmap_spoof_idle           1417
zmap_tcp_syn              1250
unicornscan_tcp_conn      1026
unicornscan_tcp_syn       1017
unicornscan_tcp_ack       1014
unicornscan_tcp_fin       1014
unicornscan_tcp_fxmas     1014
unicornscan_tcp_null      1014
unicornscan_tcp_xmas      1014
masscan_tcp_syn           1003
nmap_fragv                1003
nmap_tcp_conn             1002
nmap_tcp_null             1001
nmap_tcp_syn              1001
nmap_tcp_xmas             1001
hping_tcp_syn             1001
nmap_tcp_fin              1001
hping_tcp_fin             1000
hping_tcp_ack             1000
nmap_tcp_maimon           1000
nmap_tcp_window           1000
nmap_tcp_ack              1000
hping_tcp_null            1000
hping_tcp_xmas            1000
nmap_fast                  206
nmap_top10                  26
Name: label, dtype: int64
```

Step 9: Once the data frame is reduced by removing the features, we are going to encode the labels and the ip.src column for efficient handling in the Anomaly detection models. Also, we are going to convert the Hex and other strings in the data to numerical values for the analysis so that the data is ready and fully prepared for Anomaly detection. Also, this is followed by replacing the null values and NaN (Not a Number) value from the data frame to 0, so we have uniformity in the data frame.

Step10: Now, we will split the data into a training and testing split. After this, we are using the standard scalar function to transform the training data so that the distribution of the values has a mean value of 0 and a standard deviation value as 1.

Step 11: Next, we will be implementing Isolation forest anomaly detection. This algorithm helps us detect global outliers in our dataset. On running the code with multiple values on the hyperparameter (random_state=) such as 0,3and 5 we found that the accuracy of the model was around 77%. Hence, we decided to keep the value at 3 in the code.

```
#Isolation forest
from sklearn.ensemble import IsolationForest

clf = IsolationForest(random_state=3)# hyperparameter tuning
clf.fit(X_train)

# predictions
y_pred_train = clf.predict(X_train)
y_pred_test = clf.predict(X_test)
#y_pred_outliers = clf.predict(X_outliers)

#print(list(y_pred_test).count(1), y_pred_test.shape[0])
print("Accuracy:", list(y_pred_test).count(1)/y_pred_test.shape[0])
```

```
Accuracy: 0.7714479363607177
```

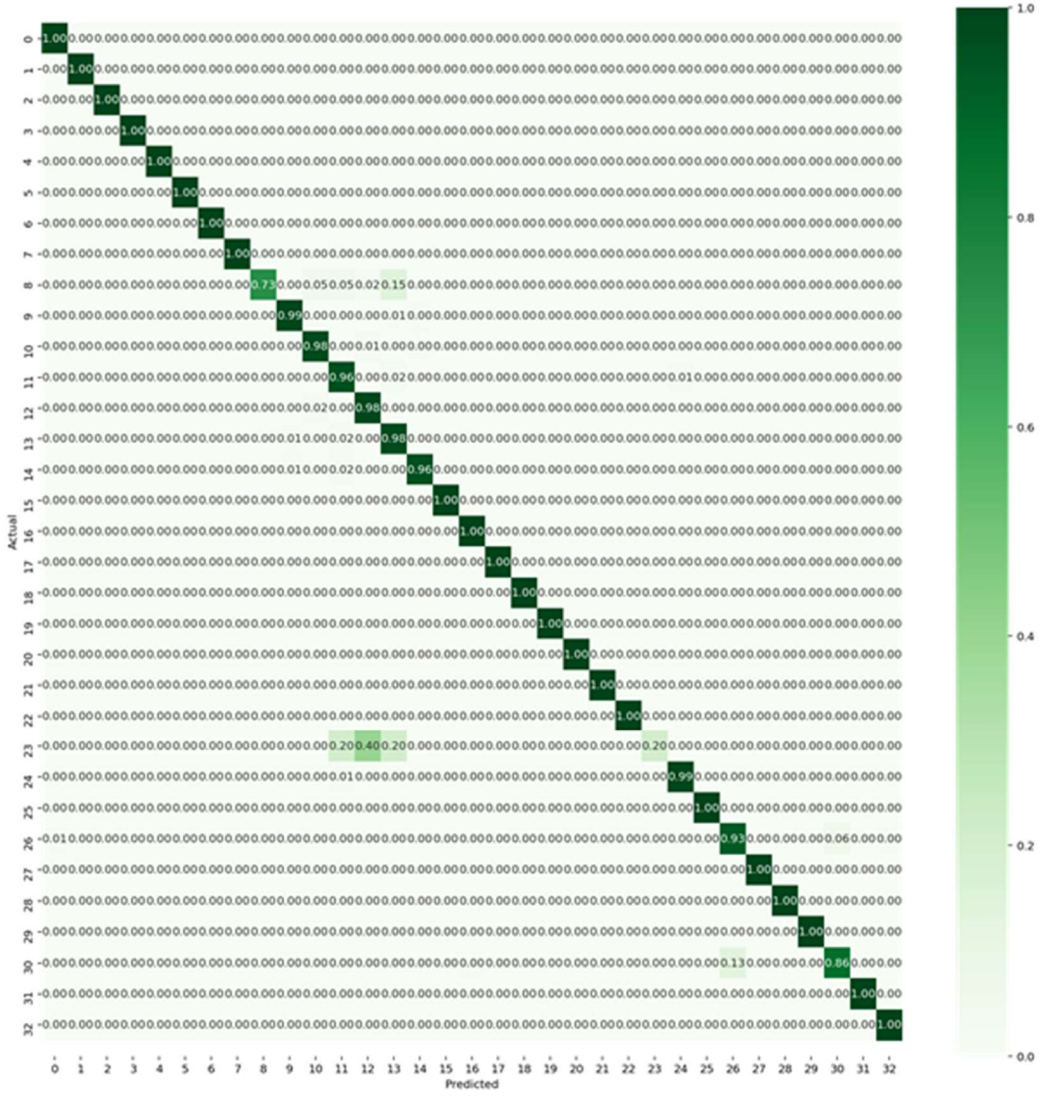
Step 12: In the next step, we are implementing the KNN classifier on the data set we have; even in this algorithm, we have only one hyperparameter which can be tuned, which is n_neighbours as shown in the screenshot below. I used multiple values like 3,5, 7, etc., and the answer for the accuracy for this algorithm stayed around 99 %, as shown in the below screenshot. But when considering the decimal values, it remained highest for the value 3. Hence the code has the value 3. We Also generated the Confusion matrix to show the correct and incorrect predictions broken down by each class label. Since we have 33 labels, 32 of which are malicious probing sets, and the last one is the normal traffic. We can see that in the confusion matrix below. We have also generated the AUC

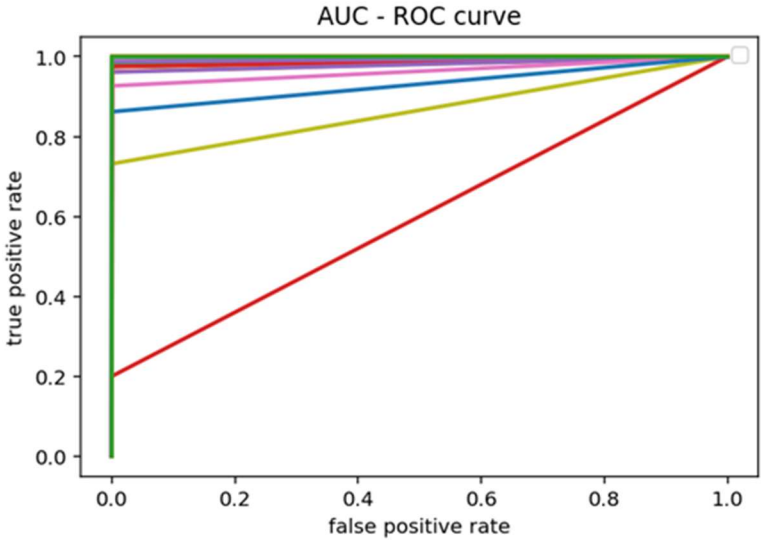
-ROC curve that shows the performance measurement for this classification problem using the KNN classifier.

```
# KNN implementation, Accuracy for the model, confusion matrix AUC-ROC curve for the same .
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=7)# hyperparameter tuning
neigh.fit(X_train, train_Y)
y_pred = neigh.predict(X_test)
from sklearn.metrics import accuracy_score

print(accuracy_score(val_Y, y_pred))
from sklearn import metrics
y_test1 = val_Y
y_pred1 = y_pred
```

```
0.9957193307887133
[[20619     0     0 ...     0     0     0]
 [    0   200     0 ...     0     0     0]
 [    0     0   200 ...     0     0     0]
 ...
 [    0     0     0 ...   175     0     0]
 [    0     0     0 ...     0   203     0]
 [    0     0     0 ...     0     0   250]]
```


Actual
Predicted
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
1.0
0.8
0.6
0.4
0.2
0.0

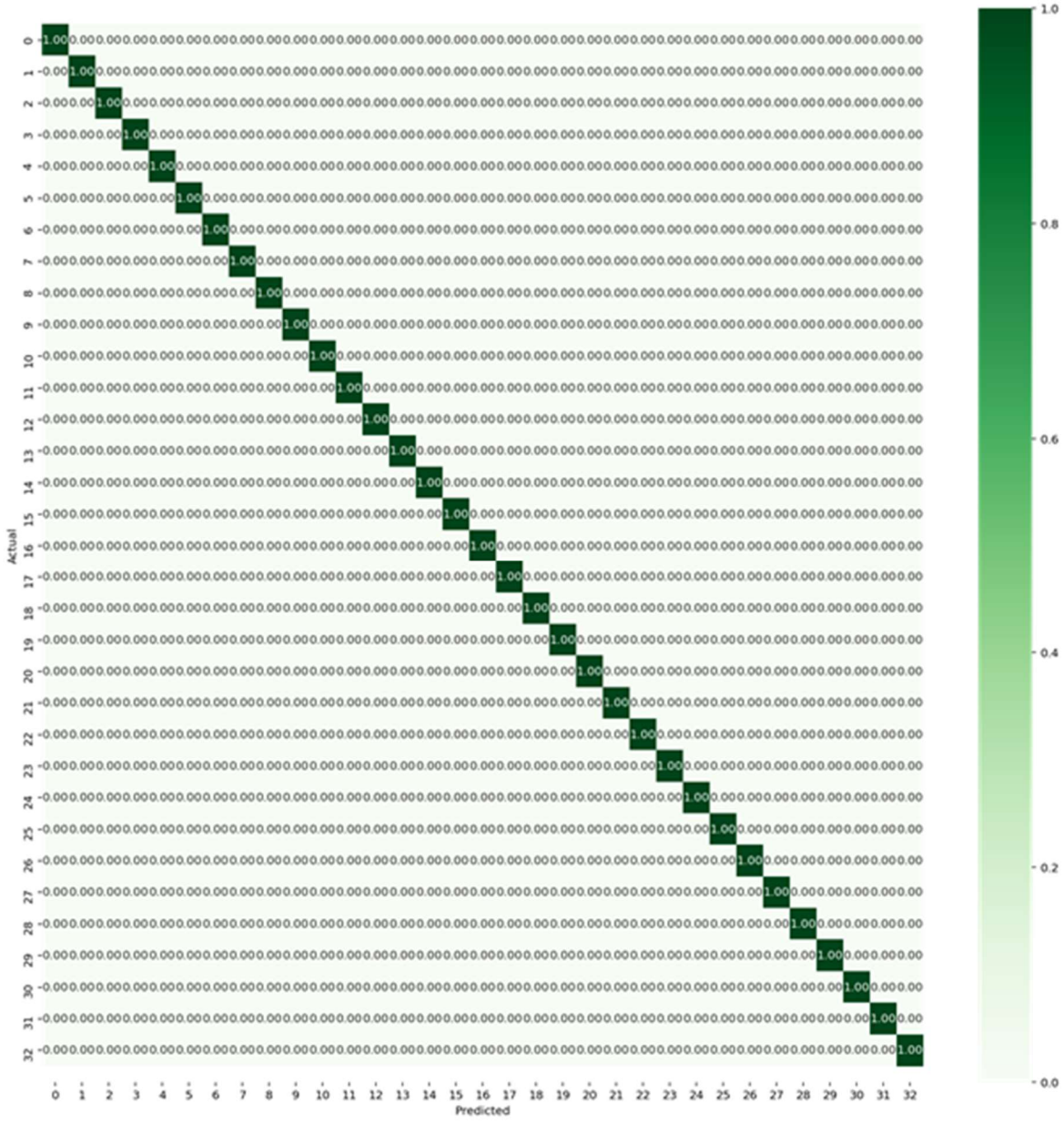

AUC - ROC curve
true positive rate
false positive rate
0.0
0.2
0.4
0.6
0.8
1.0

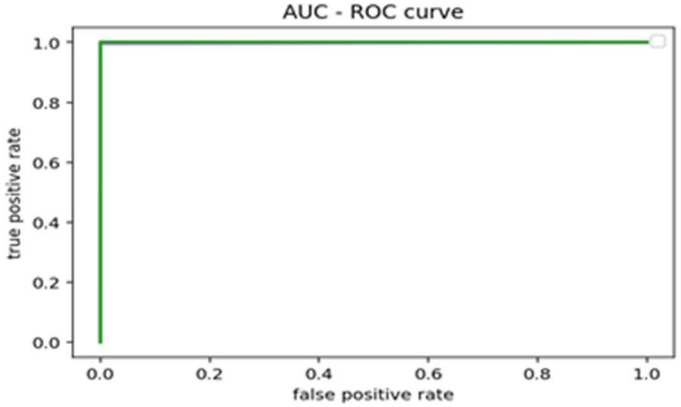
Step 13: Now, we are going to implement the Random Forest algorithm for our data set. The Hyperparameters for the algorithm are n_estimators criterion and random_state. On applying different value to the parameters, the accuracy didn't change much. Hence we considered the below-shown values for the code as it was giving us 99% accuracy, which is a really good accuracy rate when compared to the Isolation forest model. We Also generated the Confusion matrix to show the correct and incorrect predictions broken down by each class label. Since we have 33 labels, 32 of which are malicious probing sets, and the last one is the normal traffic. We can see that in the confusion matrix below. We have also generated the AUC -ROC curve that shows the performance measurement for this classification problem using Random forest.

Classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 42)

```
plt.show()
```

```
[ 0  0  0 ... 30 11  0]
[ 0  0  0 ... 30 11  0]
0.9999286555131452
```

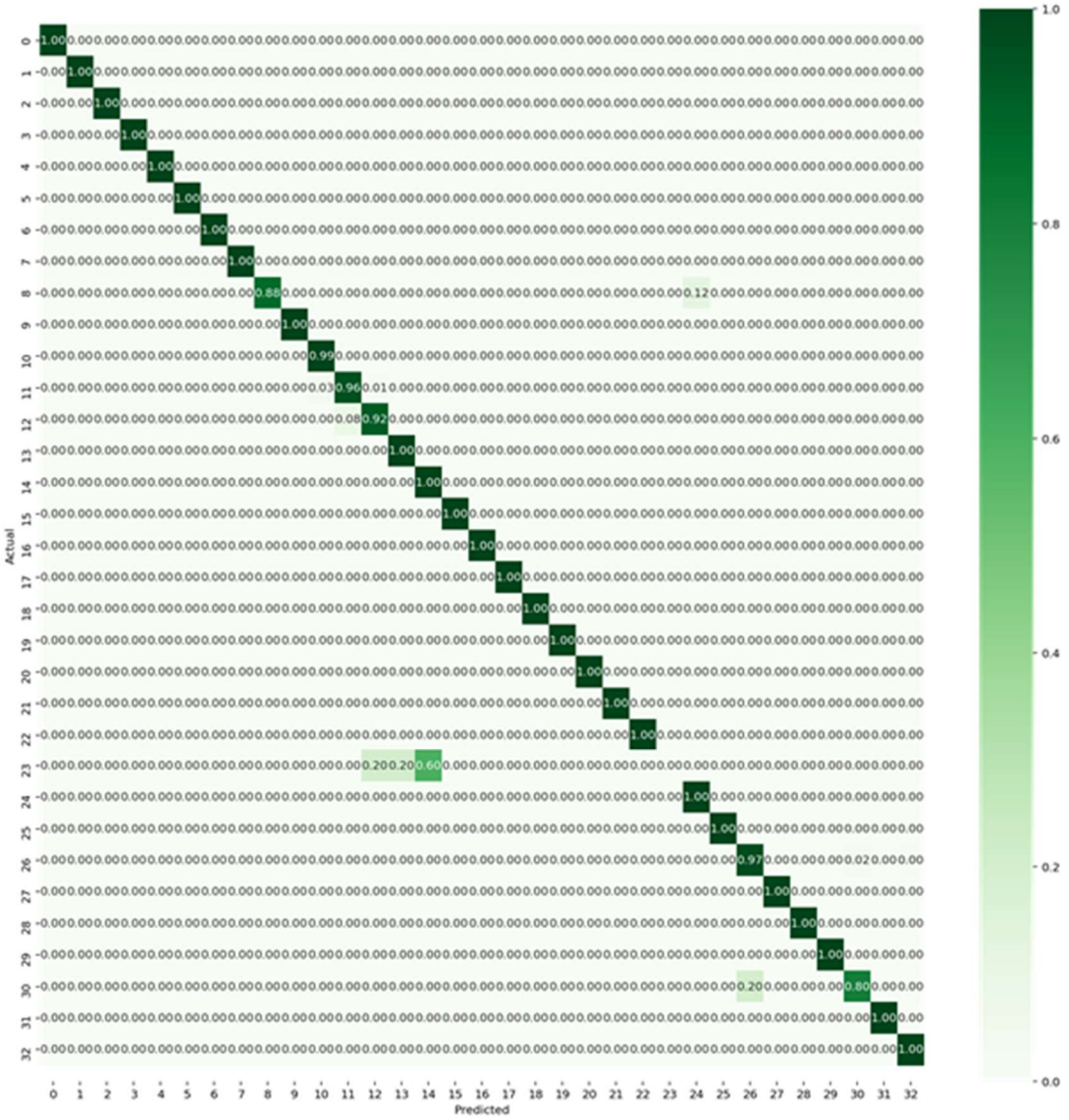

Actual
Predicted
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
1.0
0.8
0.6
0.4
0.2
0.0


AUC - ROC curve
true positive rate
false positive rate
0.0 0.2 0.4 0.6 0.8 1.0

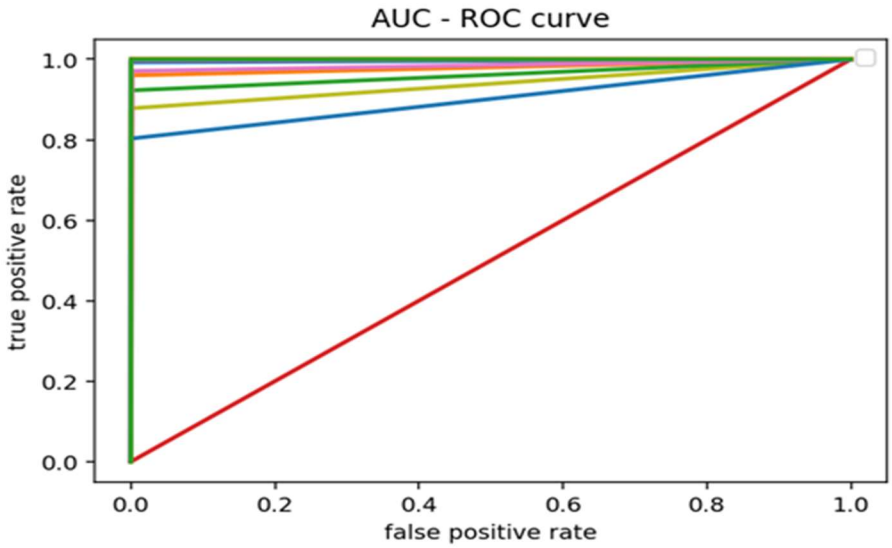
Step 14: We have also implemented SVM Algorithm just to check if the accuracy using this Machine learning technique would give us any different results, but this algorithm also gives us the same results as the Random forest and the KNN classifier the results of the SVM algorithm's Accuracy score, confusion Matrix and the AUC-ROC curve is pasted below.

```
plt.show()

0.996147397709842
[[20619     0     0 ...     0     0     0]
 [    0   200     0 ...     0     0     0]
```



Confusion Matrix for the SVM Algorithm.



Conclusion

On comparing the accuracy results of each of the algorithms, we see that the Random Forest algorithm has the highest accuracy, which is above 99.99%, for this type of data set. The analysis of the network traffic is best predicted using supervised Machine learning models like the Random Forest, SVM, and KNN, which gives us the accuracy in the ballpark of 99.5% and above, which is a very efficient accuracy. At the same time, unsupervised models like the isolation forest will not be able to give us a high degree of accuracy. The isolation forest algorithm is able to detect anomalies that are deviating heavily from the clusters which it forms, and the local outliers have gone unnoticed. Hence in this domain of detection of probing attacks, it can be inferred that supervised machine learning models will give us the desired results and best performance.

This project on implementing a live scenario can easily detect any type of probing attack that is being performed on the network with an accuracy of 99.99% and will help the organizations to defend themselves when under attack.

References

[1]] "Detection of Probe Attacks Using Machine Learning Techniques", www.academia.edu, 2021. [Online]. Available: https://www.academia.edu/download/38386389/7.pdf. [Accessed: 03- Jun- 2021].

[2] I. Ahmad, A. B. Abdullah and A. S Alghamdi, "Application of artificial neural network in detection of probing attacks," 2009 IEEE Symposium on Industrial Electronics & Applications, 2009, pp. 557-562, doi: 10.1109/ISIEA.2009.5356382.

[3] Taeshik Shon, Yongdae Kim, Cheolwon Lee and Jongsub Moon, "A machine learning framework for network anomaly detection using SVM and GA," Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop, 2005, pp. 176-183, doi: 10.1109/IAW.2005.1495950.

[4] AAMIR, Muhammad et al. Machine Learning Classification of Port Scanning and DDoS Attacks: A Comparative Analysis. Mehran University Research Journal of Engineering and Technology, [S.l.], v. 40, n. 1, p. 215- 229, jan. 2021. ISSN 2413-7219. Available at: <https://publications.muet.edu.pk/index.php/muetrj/article/view/1999>. Date accessed: 03 june 2021. doi: http://dx.doi.org/10.22581/muet1982.2101.19.

[5] Amit, I, Matherly, J, Hewlett, W, Xu, Z, Meshi, Y, & Weinberger, 'Machine Learning in Cyber-Security - Problems, Challenges and Data Sets'. Available: arXiv:1812.07858

[6] T. Thomas, A. P. Vijayaraghavan and S. Emmanuel, "Machine Learning and Cybersecurity", *Machine Learning Approaches in Cyber Security Analytics*, pp. 37-47, 2019. Available: 10.1007/978-981-15-1706-8_3.

[7] P. Dasgupta and J. Collins, "A Survey of Game Theoretic Approaches for Adversarial Machine Learning in Cybersecurity Tasks", *AI Magazine*, vol. 40, no. 2, pp. 31-43, 2019. Available: 10.1609/aimag.v40i2.2847.

[8] M. Aamir, S. Rizvi, M. Hashmani, M. Zubair and J. Usman, "Machine Learning Classification of Port Scanning and DDoS Attacks: A Comparative Analysis", *January 2021*, vol. 40, no. 1, pp. 215-229, 2010. Available: 10.22581/muet1982.2101.19.

[9] C. Huang, T. Lee, L. Chang, J. Lin and G. Horng, "Adversarial Attacks on SDN-Based Deep Learning IDS System", *Lecture Notes in Electrical Engineering*, pp. 181-191, 2018. Available: 10.1007/978-981-13-1059-1_17.

[10] H. Viet, Q. Van, L. Trang and S. Nathan, "Using Deep Learning Model for Network Scanning Detection", *Proceedings of the 4th International Conference on Frontiers of Educational Technologies - ICFET '18*, 2018. Available: 10.1145/3233347.3233379.