DDOS Attack Simulation and Machine Learning Model for Intrusion Detection

M.Sc. Computer Science (Specialization: Cyber Security)

by

A Rajesh, Dhruti Ranjan Mohanty, Kamala Kannan.P,

Mahija .M.S, Naveen Kumar S, Nihala MN, Santhosh P, Thanigai Vel.R

Enrollment no.: MSCS21R002, MSCS21R004, MSCS21R006,

MSCS21R007, MSCS21R008, MSCS21R009, MSCS21R010, MSCS21R012

Under the guidance of

Dr.P.Thiyagarajan

Associate Professor and Head

Department of Computer Science (Cyber Security)

DEPARTMENT OF COMPUTER SCIENCE (CYBER SECURITY)

RAJIV GANDHI NATIONAL INSTITUTE OF YOUTH DEVELOPMENT

SRIPERUMBUDUR – 602 105

Department of Computer Science
Rajiv Gandhi National Institute of Youth Development Sriperumbudur, Tamil Nadu India - 602105

CERTIFICATE

This is to certify that I have examined the project entitled "**DDOS Attack Simulation and Machine Learning Model for Intrusion Detection**", submitted by **A. Rajesh, Dhruti Ranjan Mohanty, Kamala Kannan .P, Mahija.M.S, Naveen Kumar S, Nihala M N, Santhosh P, Thanigai Vel.R** *(Roll Number: MSCS21R002, MSCS21R004, MSCS21R006, MSCS21R007, MSCS21R008, MSCS21R009, MSCS21R010, MSCS21R012)*, the postgraduate students of **Department of Computer Science** in partial fulfilment for the award of degree of Master of Computer Science with specialization of CyberSecurity. I hereby accord my approval of it as a study carried out and presented in a manner required for its acceptance in fulfilment for **CSNS301 – Intrusion Detection System** project course for which it has been submitted. The project has fulfilled all the requirements as per the regulations of the institute as well as course instructor and has reached the standard needed for submission.

Supervisor
Dr.P.Thiyagarajan
Department of Computer Science
(Cyber Security)
RGNID, Sriperumbudur

Place: Sriperumbudur
Date: 10/02/2023

DDOS Attack Simulation and Machine Learning Model for Intrusion Detection

A.Rajesh[*], Dhruti Ranjan Mohanty[*], Kamala Kannan.P[*], Mahija.M.S[*], Naveen Kumar.S[*]

Nihala.M.N[*], Santhosh.P[*], Thanigai Vel.R[*]

Dr. P.Thiyagarajan[1]

Students, Dept. of Computer Science (Cyber Security), RGNIYD, Tamil Nadu, India[*]

Associate Professor & Head, Dept. of Computer Science (Cyber Security), RGNIYD, Tamil Nadu India[1]

Abstract: The study of intrusion detection has gained significant importance over the past few years. To increase system accuracy and lower the false positive rate, researchers now focus on a variety of datasets. Only in the presence of a useful dataset can build intelligent intrusion detection systems. An intrusion detection system can only be trained and tested using a data set that contains a large volume of high-quality data that resembles real-time events. During the last decade the analysis of intrusion detection has become very important, the researcher focuses on various dataset to improve system accuracy and to reduce false positive rate based on DAPRA 98 and later the updated version as KDD cup 99 dataset which shows some statistical issues, it causes the performance of the security analysis to suffer, which results in the replacement of the KDD dataset with the NSL-KDD dataset. It also weakens the evaluation of anomaly detection. This research comprises of two parts. One part focuses on simulation of DDOS attack in a safe environment and another part focuses on a thorough analysis of the NSL-KDD dataset, which only comprises chosen records. This particular dataset offers a useful examination of several machine learning methods for intrusion detection.

Keywords: Intrusion Detection System, DDOS, NSL-KDD dataset, Machine learning, Data analysis

I. Introduction

The use of communication systems is essential to the everyday lives of most people. Computer networks may be utilised efficiently for processing corporate data, teaching and learning, teamwork, acquiring large amounts of data, and entertainment. With the goal of making it transparent and user-friendly, the computer network protocol stack that is currently in use was created. A solid communication protocol stack was eventually created as a result. The protocol is prone to being attacked by outsiders because of its flexibility, which are difficult and costly to be solved by manufactures. There is no denying that as technology advances, the frequency of hacking and intrusion events rises. The protocol's versatility has rendered it susceptible to attacks by intruders, highlighting the need for continuous monitoring and protection of computer networks. The automation of the monitoring process is facilitated by the use of an Intrusion Detection System (IDS), which is a combination of hardware and software components. This system plays a crucial role in maintaining the security and integrity of computer networks. The heavy traffic generated by clients visiting a web server at any given time presents a need for efficient processing of the resulting voluminous database. To accomplish this, machine learning techniques can be employed to classify the logged traffic data into normal and abnormal traffic based on its attributes, which can be visualized as a set of network connections.

One approach to mitigate potential risks and improve the security posture of computer systems is through the use of safe simulation environments. The use of operating systems such as Parrot OS and Linux provides a platform for conducting secure simulations and testing of various security measures. In this research work, we aim to explore the effectiveness of using these operating systems for simulating DDOS attack using various tools.

Data extraction from large data sets using machine learning techniques is referred to as Data Mining. The NSL-KDD dataset is subjected to a thorough

analysis to identify and categorize the various types of cyber-attacks. The data is divided into test and training datasets and four distinct attack clusters are identified. An in-depth study of the datasets is conducted to better understand the nature of the attacks. The rest of the paper is organized as follows: Section II presents simulation of DDOS attack and capturing. Section III describes briefly about the NSL-KDD dataset. Section IV provides an in-depth analysis of the dataset through the application of various classification techniques and graphical view of various scoring methods. Section V deals with conclusion and future work.

II. Simulation of DDOS attack and Capturing Network traffic

A DDoS (Distributed Denial of Service) attack is a type of cyber-attack in which multiple compromised computers are used to flood a target system or network with traffic, rendering it unavailable to users. The goal of a DDoS attack is to overload the target with so much traffic that it becomes difficult or impossible to access. This type of attack is often launched from a large network of infected computers, known as a botnet, making it difficult to trace the source of the attack and defend against it.

- *Safe Environment*

  There is no specific operating system that is solely designed for conducting DDoS attacks. DDoS attacks can be launched from a variety of operating systems, including Windows, Linux, and macOS. The operating system used to launch a DDoS attack is usually a secondary factor, as the attacker typically relies on specialized tools and malware to carry out the attack.

  That being said, some operating systems may be more susceptible to compromise by attackers, making them easier targets for infection and recruitment into a botnet. It is important to keep all operating systems and software up-to-date with the latest security patches and to practice good cyber security habits, such as using strong passwords and avoiding suspicious emails and attachments, to minimize the risk of compromise.

  Here in this paper ParrotOS is used to simulate a DDoS attack. Parrot OS provides a huge arsenal of tools, utilities and libraries that IT and security professionals can use to test and assess the security of their assets in a reliable, compliant and reproducible way. Parrot OS is a Debian-based Linux distribution that is commonly used for ethical hacking and penetration testing. While Parrot OS can be used to carry out various types of security assessments, it is not designed or intended for conducting DDoS attacks.

- *Tools for DDoS Attack*
  - Xerxes
  - GoldenEye
  - Slowloris
  - LOIC (Low Orbit Ion Cannon)
  - HOIC (High Orbit Ion Cannon)
  - THC-SSL-DoS
  - HULK (http Unbearable Load King)
  - Pyloris
  - TOR's Hammer
  - XOIC
  - RUDY (R U Dead Yet?)
  - DAVOSET
  - OWASP HTTP POST

  Here in this paper Xerxes is used to perform DDoS attack.

- *Execution of Xerxes*

  Xerxes is a free and open source tool available on GitHub. You can install and download the tool free of cost. A denial of service attack can be performed by using this tool. Xerxes is written in C. The framework works by maintaining a full TCP connection. After making full TCP Connection it only requires a few hundreds of requests at long term in regular intervals. As a result, Xerxes doesn't need to spend lots of traffic requests to exhaust all the available connections on a server. Using xerxes any remote machine can be taken down or any server can be taken down easily. The tool use perfectly legitimate HTTP traffic. The tool is very useful for security testing.

Steps to perform Xerxes:

1. Clone the Xerxes from Github and save it in a folder.
2. Navigate to the cloned folder
   cd xerxes
3. Compile with GNU GCC compiler.
   gcc xerxes.c –o xerxes
4. Lunch the attack
   /xerxes <example.com> 80

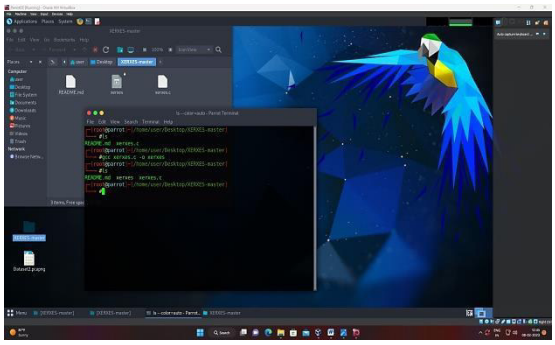

Figure 1: Compile with GNU GCC compiler

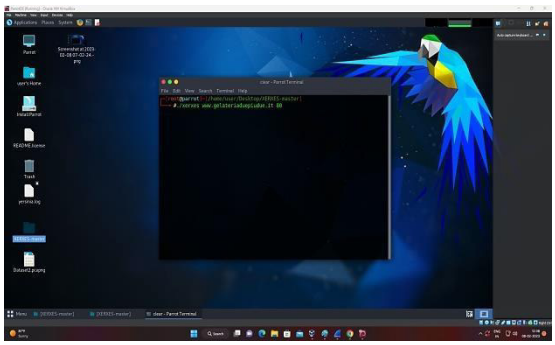

Figure 2: Launch the attack

- *Network Traffic Capturing*
  The attack is done by using the Parrot OS using the tool Xerxes and the attack are captured by using the Wireshark tool that will capture the network packets in Windows 10 OS.

  The capturing method in Wireshark can be performed in the following ways:
  - Live Capture: Wireshark can capture packets in real-time by monitoring network interfaces.
  - Offline Capture: Wireshark can also read and analyze captured packet data from a saved file.
  - Remote Capture: Wireshark can capture packets from a remote machine by using protocols such as TCP dump or Remote Capture protocol (RCP).

Note: To capture packets, Wireshark must be run with sufficient permissions, such as administrator or root privileges.

STEPS TO CAPTURE THE ATTACK

a. Run the Wireshark in the target machine parallel while performing attack in Virtual Machine.

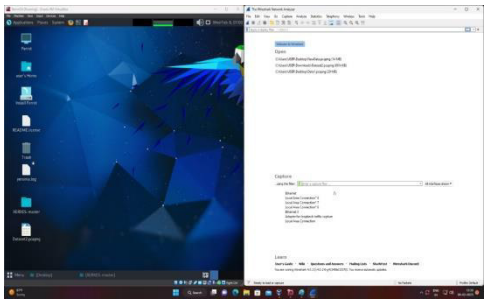

Figure 3: Wireshark is running on the target machine

b. Capturing the network packets and saving file.

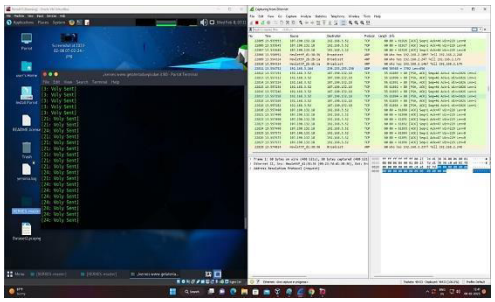

Figure 4: Capturing the network traffic

c. Exporting the file into .csv file.

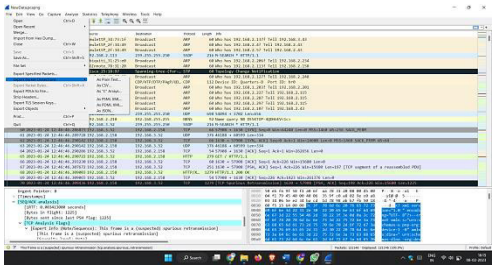

Figure 5: Saving the file into .csv file for further examination

d. Confirming whether the attack is captured in the collected data.

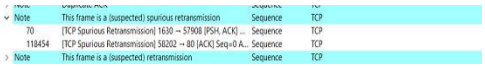

Figure 6: Confirming that the attack is captured in the saved file.

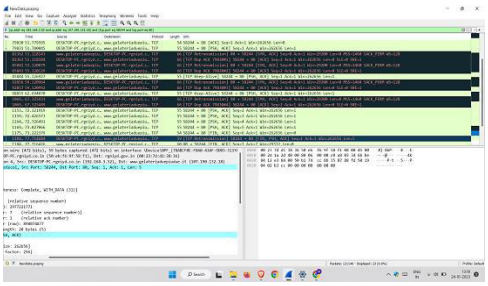

Figure 7: Details of the attacked website as per the expert information.

III. Dataset Description

The NSL-KDD (NSL-KDD Intrusion Detection System (IDS) Evaluation Data Set) is a widely used data set for the evaluation of intrusion detection systems (IDSs). It was created by taking the KDD Cup 1999 dataset and transforming it into a more realistic and challenging dataset. The NSL-KDD dataset is widely used in the field of cyber security as it provides a large and diverse collection of network traffic patterns, including both normal and intrusion traffic. This data set is also commonly used for the development and testing of machine learning algorithms for intrusion detection.

The NSL KDD dataset encompasses comprehensive records and is made available to researchers through a collection of downloadable files, as listed in Table I.

TABLE I: LIST OF NSL-KDD DATASET FILES AND THEIR DESCRIPTION

Sl. No.	Name of the File	Description
1	KDDTrain+.ARFF	The full NSL-KDD train set with binary labels in ARFF format.
2	KDDTrain+.TXT	The full NSL-KDD train set including attack-type labels and difficulty level in CSV format.
3	KDDTrain+_20Percent.ARFF	A 20% subset of the KDDTrain+.arff file.
4	KDDTrain+_20Perce nt.TXT	A 20% subset of the KDDTrain+.txt file.
5	KDDTest+.ARFF	The full NSL-KDD test set with binary labels in ARFF format.
6	KDDTest+.TXT	The full NSL-KDD test set including attack-type labels and difficulty level in CSV format.
7	KDDTest-21.ARFF	A subset of the KDDTest+.arff files which does not include records with difficulty level of 21 out of 21.
8	KDDTest-21.TXT	A subset of the KDDTest+.txt files which does not include records with difficulty level of 21 out of 21.

The advantage of NSL KDD dataset is

1. No redundant records in the train set, so the classifier will not produce any biased result.

2. No duplicate record in the test set which have better reduction rates.

3. The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD data set.

The NSL-KDD dataset comprises of 41 attributes that depict various features of a network connection flow and a label assigned to each record, either as normal or an attack type. The attributes and their descriptions, along with sample data, are listed in Tables II, III, IV, and V. Table VI provides information about the data type of all 41 attributes. The 42nd attribute comprises data on the five classes of network connection vectors, including one normal class and four attack classes. The attack classes are grouped into DoS, Probe, R2L, and U2R categories and have a detailed description provided in Table VII.

TABLE II: BASIC FEATURES OF EACH NETWORK CONNECTION VECTOR

Sl. No.	Attribute Name	Description
1	Duration	Length of time duration of the connection.
2	Protocol_type	Protocol used in the connection.
3	Service	Destination network service used.
4	Flag	Status of the connection – Normal or Error.
5	Src_bytes	Number of data bytes transferred from source to destination in single connection.
6	Dst_bytes	Number of data bytes transferred from destination to source in single connection.
7	Land	if source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0.
8	Wrong_fragment	Total number of wrong fragments in this connection.
9	Urgent	Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated.

TABLE III: CONTENT RELATED FEATURES OF EACH NETWORK CONNECTION VECTOR

Sl. No.	Attribute Name	Description
10	Hot	Number of "hot" indicators in the content such as: entering a system directory, creating. programs and executing programs.
11	Num_failed _logins	Count of failed login attempts.
12	Logged_in	Login Status : 1 if successfully logged in; 0 otherwise.
13	Num_compromised	Number of "compromised " conditions.
14	Root_shell	1 if root shell is obtained; 0 otherwise.
15	Su_attempted	1 if "su root" command attempted or

		used; 0 otherwise.
16	Num_root	Number of "root" accesses or number of operations performed as a root in the connection.
17	Num_file_creations	Number of file creation operations in the connection.
18	Num_shells	Number of shell prompts.
19	Num_access_files	Number of operations on access control files.
20	Num_outbound_cmds	Number of outbound commands in an ftp session.
21	Is_hot_login	1 if the login belongs to the "hot" list i.e., root or admin; else 0.
22	Is_guest_login	1 if the login is a "guest" login; 0 otherwise.

TABLE IV: TIME RELATED TRAFFIC FEATURES OF EACH NETWORK CONNECTION VECTOR

Sl. No.	Attribute Name	Description
23	Count	Number of connections to the same destination host as the current connection in the past two seconds.
24	Srv_count	Number of connections to the same service (port number) as the current connection in the past two seconds.
25	Serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23).
26	Srv_serror_rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24).
27	Rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (23).
28	Srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24).
29	Same_srv_rate	The percentage of connections that were to the same service, among the connections aggregated in count (23).
30	Diff_srv_rate	The percentage of

		connections that were to different services, among the connections aggregated in count (23).
31	Srv_diff_host_rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count (24).

TABLE V: HOST BASED TRAFFIC FEATURES IN A NETWORK CONNECTION VECTOR

Sl. No.	Attribute Name	Description
32	Dst_host_count	Number of connections having the same destination host IP address.
33	Dst_host_srv_count	Number of connections having the same port number.
34	Dst_host_same _srv_rate	The percentage of connections that were to the same service, among the connections aggregated in dst_host_coun t (32).
35	Dst_host_diff_ srv_rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_coun t (32).
36	Dst_host_same _src_port_rate	The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_c ount (33).
37	Dst_host_srv_ diff_host_rate	The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_c ount (33).
38	Dst_host_serror_rat e	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_coun t (32).
39	Dst_host_srv_serror _rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections. aggregated in dst_host_srv_c ount (33).
40	Dst_host_rerror_rat e	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in

		dst_host_count (32).
41	Dst_host_srv_rerror_rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_count (33).

TABLE VI: ATTRIBUTE VALUE TYPE

Type	Features
Nominal	Protocol_type(2), Service(3), Flag(4).
Binary	Land(7), logged_in(12), root_shell(14), su_attempted(15), is_host_login(21), is_guest_login(22).
Numeric	Duration(1), src_bytes(5), dst_bytes(6), wrong_fragment(8), urgent(9), hot(10), num_failed_logins(11), num_compromised(13), num_root(16), num_file_creations(17), num_shells(18), num_access_files(19), num_outbound_cmds(20), count(23), srv_count(24), serror_rate(25), srv_serror_rate(26), rerror_rate(27), srv_rerror_rate(28), same_srv_rate(29) diff_srv_rate(30), srv_diff_host_rate(31), dst_host_count(32), dst_host_srv_count(33), dst_host_same_srv_rate(34), dst_host_diff_srv_rate(35), dst_host_same_src_port_rate (36), dst_host_srv_diff_host_rate(37), dst_host_serror_rate(38), dst_host_srv_serror_rate(39), dst_host_rerror_rate(40), dst_host_srv_rerror_rate(41).

TABLE VII: MAPPING OF ATTACK CLASS WITH ATTACK TYPE

Attack Class	Attack Type
Dos	Back, Land, Neptune, Pod, Smurf, Teardrop,Apache2, Udpstorm, Processtable, Worm (10)
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint (6)
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named (16)
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps (7)

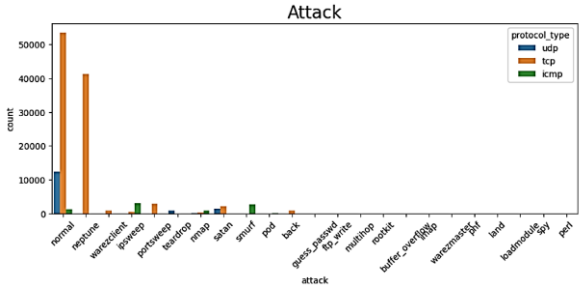

Figure 3: Relationship between protocol type and occurrence of attack in KDDTrain+.txt dataset.

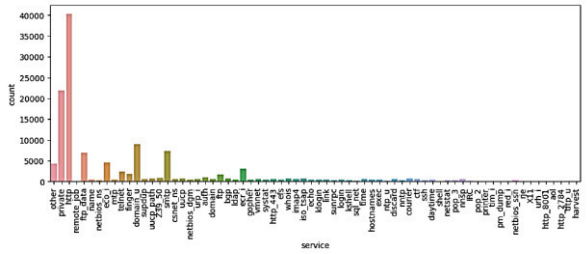

Figure 4: Most service used in KDDTrain+.txt dataset

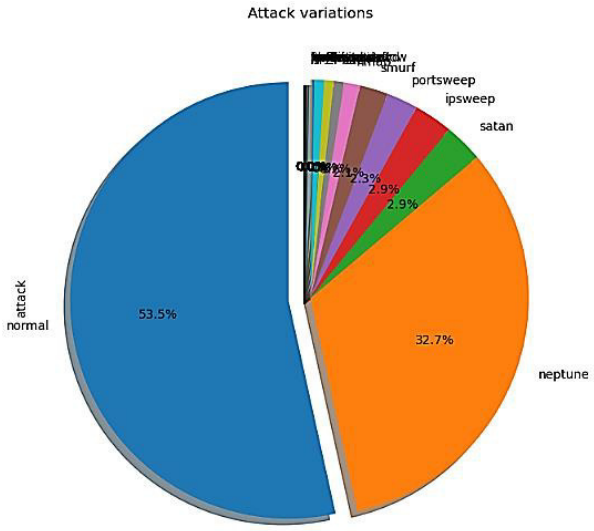

Figure 5: Attack Variation in KDDTrain+.txt dataset

IV. Analysis of dataset and Classification Techniques

This section focuses on the analysis of dataset and classification techniques used for machine learning model.

A. Experimental Setup

The purpose of this is basic exploration of the NSL-KDD dataset. Here are the goals of this exploration:

- Gain a basic, understanding of the dataset
- Examining the potential application of the data set in predicting network anomalies or security breaches
- Walk through some fundamental concepts of building machine learning models

B. Pre-processing and Feature Selection

Pre-processing refers to a set of techniques applied to raw data before feeding it into a machine learning algorithm. The goal of pre-processing is to prepare the data in a suitable format for the algorithm, improve the quality of the data, and increase the performance of the model. Common pre-processing techniques include: Data cleaning, Data transformation, Data reduction, Data discretization.

Feature selection is the process of selecting a subset of relevant features from a larger set of features to use in a machine learning model. The goal of feature selection is to reduce the dimensionality of the data, increase the interpretability of the model, improve the accuracy of the model, and reduce the computation time required to train the model. There are several techniques for feature selection, including: Filter methods such as chi-squared or mutual information, and select the top-k features based on this test. In this paper mutual information is used.

Mutual information is a measure of the statistical dependence between two variables. In feature selection, mutual information is used to evaluate the relationship between each feature and the target variable. Features with high mutual information values are considered to be more informative and relevant for the problem at hand.

K-best feature selection is a simple feature selection method that selects the top K features based on a ranking criterion, such as mutual information. This method is particularly useful when the number of features is large and it is not feasible to evaluate all features in terms of their relationship with the target variable. By selecting only the K best features, this method can reduce the dimensionality of the data and improve the performance of machine learning models.

C. Classification Techniques

Classification is a machine learning technique used to predict a categorical output (i.e., a label or class) based on input features. There are several different classification techniques: including:

- Logistic Regression: A linear model used for binary classification problems. It predicts the probability of a certain class and outputs the class with the highest probability.
- K-Nearest Neighbors (k-NN): A non-parametric method that uses the k-nearest neighbors of a given sample to make predictions.
- Naive Bayes: A probabilistic method that makes predictions based on Bayes'

Theorem and the assumption of independence between features.

- Decision Trees: A tree-based model that splits the data based on feature values until each leaf node represents a unique class.
- Random Forest: An ensemble method that combines multiple decision trees to make predictions.
- Support Vector Machines (SVM): A linear or non-linear model that finds the hyper plane that maximally separates the classes.

Each classification technique has its own strengths and weaknesses, and it's important to choose the right technique for a given problem based on the characteristics of the data and the requirements of the problem.

For example, decision trees can be prone to over fitting and can take a long time to build the tree when dealing with large datasets. The k-Nearest Neighbor method can be computationally expensive when the size of the dataset grows, as it requires comparing each new sample to every other sample in the dataset. Neural networks are best suited for numerical data and require pre-processing to convert text data into numerical form.

It's also worth noting that there is no one-size-fits-all solution in machine learning, and combining multiple techniques can often lead to improved performance compared to using a single method. Figure 6 and 7 provides information about the test result with top 10 and 20 features respectively. Figure 8 and 9 provides visual representation of ROC curve.

	Accuracy	Precision	Recall
Logistic Regression	0.717030	0.943569	0.611111
Support Vector Machines linear	0.723861	0.955617	0.615630
Decision Trees	0.823005	0.840902	0.769579
Random Forest	0.758151	0.970343	0.645986
Naive Bayes	0.546245	0.440634	0.471463
K-Nearest Neighbor	0.788937	0.962002	0.680358

Figure 6: Test result with top 10 features

	Accuracy	Precision	Recall
Logistic Regression	0.842257	0.916795	0.764140
Support Vector Machines linear	0.832187	0.918031	0.749034
Decision Trees	0.862884	0.906498	0.801293
Random Forest	0.835248	0.970240	0.733401
Naive Bayes	0.608038	0.427247	0.558938
K-Nearest Neighbor	0.838265	0.970240	0.737303

Figure 7: Test result with top 20 features

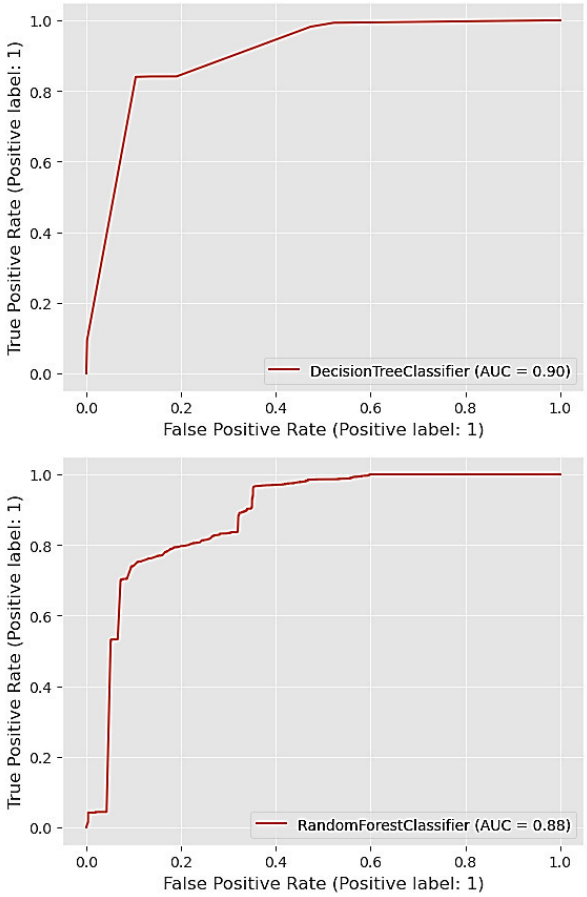

Figure 8: ROC curve of Decision Tree and Random Forest with top 10 features.

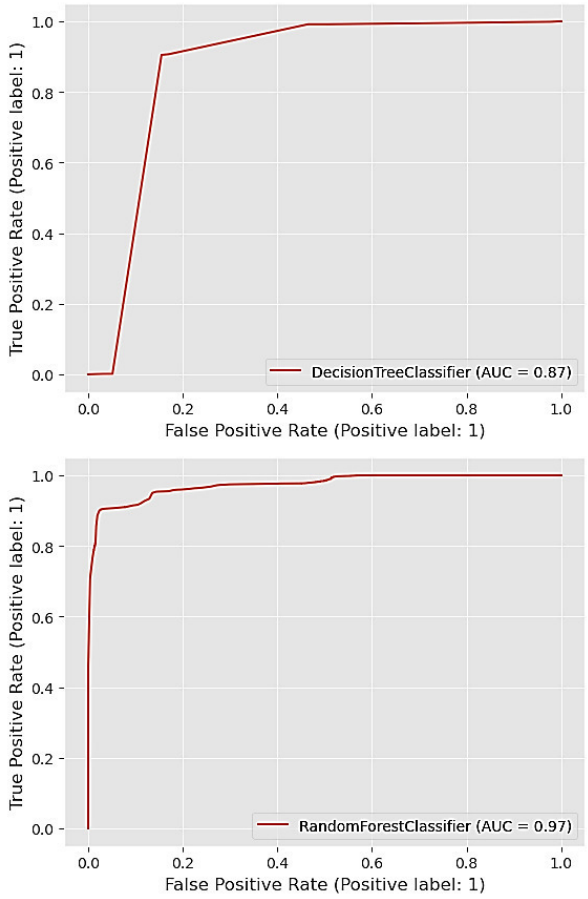

Figure 9: ROC curve of Decision Tree and Random Forest with top 20 features.

V. Conclusion and Future Work

The results of the analysis on the NSL-KDD dataset indicate that it is a suitable dataset for testing the performance of intrusion detection systems (IDSs). The use of the mutual information based feature selection method for dimensionality reduction leads to a reduction in detection time and improvement in accuracy. The analysis, through the use of figures and tables, provides a clear understanding of the dataset and highlights that Random Forest and Decision Tree are best model for this problem. Random Forest is considered the best model for several reasons such as robustness, ability to handle large datasets, feature importance, accuracy and easy to interpret. Decision trees can be prone to over fitting, especially when the tree is deep or has many branches. In contrast, random forests address the over fitting issue by creating multiple decision trees and combining their predictions. Random forests generally perform better with high dimensional and complex datasets, while decision trees are easier to interpret and can be used for small datasets.

Further research is planned to investigate the potential of using optimization techniques to enhance the accuracy rate of the intrusion detection model and to implement machine learning model in real network traffic to prevent the attack.

VI. Team Work

- *A.Rajesh*[*] & *Santhos.P*[*] collects the information about DDoS attack, how it works, what are the characteristics of the attack and why the attack happens in a system.
- *Nihala M N*[*] research about various types of tools to simulate the DDoS attack in a safe environment.
- *Mahija M S*[*] research about how to capture and analyse the network traffic using wireshark.
- *Naveen Kumar*[*] performs the data pre-processing on NSL-KDD Training dataset.
- *Kamal Kannan*[*] handle the outliers in the dataset.
- *Thanigai Vel.R*[*] research about feature engineering and feature extraction methods.
- *Dhruti Ranjan Mohanty*[*] performs the data analysis, modelling and overall coding to find out the best machine learning model.

VII. Challenges

- In the simulation part, installation and coding for all other tools were done successfully. But the site is still able to access. We tried in both Windows and Linux, and it’s not working. While using Slowloris or GoldenEye Tools for performing DDoS, its detected that suspicious request is transferred to the victim's site in the software 'EtherApe'. But due to legal issues still the site is not unavailable. Even though we are using xerxos tool for attacking, it’s not possible to perform attack on https sites. It’s only possible to attack on http sites and only for study purpose.
- The first difficulty we faced is to understand the dataset specifically their columns. Some

of the columns are there which we are not aware of that.

- Then second we faced difficulty to analyse the outliers through boxplot, it's may be we have lack of domain knowledge. We can't able to remove the outliers from the dataset.
- Third, we got confused while we were dealing with features; we didn't able get the exact reason of choosing top k features.
- Last we faced difficulty to analyse the confusion matrix and the scoring methods.

VIII. Reference

1. Sapna S. Kaushik, Dr. Prof.P.R.Deshmukh," Detection of Attacks in an Intrusion Detection System", International Journal of Computer Science and Information Technologies, Vol. 2 (3), 2011, 982-986
2. "Nsl-kdd data set for network-based intrusion detection systems." Available on: http://nsl.cs.unb.ca/KDD/NSLKDD.html, March 2009
3. https://www.kaggle.com/code/timgoodfellow/nsl-kdd-explorations#Data-profiling
4. H. Alazzam, A. Sharieh and K.E. Sabri, "A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer", Expert systems with applications, vol. 148, pp. 1-14, 2020.
5. https://www.kaggle.com/code/farazfatahnaiee/attack-detection#Data-Analysis
6. "Intrusion Detection System Classification Using Different Machine Learning Algorithms on KDD-99 and NSL-KDD Datasets - A Review Paper" International Journal of Computer Science & Information Technology (IJCSIT) Vol. 11, No 3, June 2019
7. https://towardsdatascience.com/mitigating-ddos-attacks-with-classification-models-aa75ea813d85
8. Iram Abrar, Zahrah Ayub, Faheem Masoodi, Alwi M Bamhdi "A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset" IEEE 2020 International Conference
9. D. Wang and G. Xu, "Research on the Detection of Network Intrusion Prevention with SVM Based Optimization Algorithm", Informatica, vol. 44, pp. 269-273, 2020.
10. S. Thaseen, B. Poorva and P. S. Ushasree, "Network Intrusion Detection using Machine Learning Techniques", 2020 International Conference on Emerging Trends in Information Technology and Engineering (icETITE), pp. 1-7, 2020.
11. S. Waskle and L. Parashar, "Intrusion Detection System Using PCA with Random Forest Approach", Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020), pp. 803-808, 2020.
12. Ismail, Muhammad Ismail Mohmand, Hameed Hussain, Ayaz Ali Khan, Ubaid Ullah, Muhammad Zakarya, Aftab Ahmed, Mushtaq Raza, Izaz Ur Rahman, Muhammad Haleem, "A Machine Learning-Based Classification and Prediction Technique for DDoS Attacks", IEEE Access, vol.10, pp.21443-21454, 2022.
13. S. Revathi, Dr. A. Malathi, "A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection", International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 2 Issue 12, December – 2013