Network Intrusion Detection

A report submitted in partial fulfillment of the requirements

Of

Mini-Project (ISL64)

In

Sixth Semester

By

1MS17IS040	**Dhruthick Gowda M**
1MS17IS042	**Fawaz Hussain**
1MS17IS064	**Mohammed Faisal**
1MS17IS073	**Nida Shakeb Rais**

Under the guidance of

Mrs. Lincy Meera Mathews
Assistant Professor
Dept. of ISE, RIT

RAMAIAH
Institute of Technology

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
RAMAIAH INSTITUTE OF TECHNOLOGY
(AUTONOMOUS INSTITUTE AFFILIATED TO VTU)
M. S. RAMAIAH NAGAR, M. S. R. I. T. POST, BANGALORE – 560054

2019-2020

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute Affiliated to VTU)

M. S. Ramaiah Nagar, M. S. R. I. T. Post, Bangalore – 560054

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

RAMAIAH
Institute of Technology

CERTIFICATE

This is to certify that the project work entitled **"Network Intrusion Detection"** is a bonafide work carried out by **Dhruthick Gowda M, Mohammad Faisal Abdul Khader, Fawaz Hussain and Nida Shakeb Rais** bearing **USN: 1MS17IS040, 1MS17IS064, 1MS17IS042 & 1MS17IS073** respectively in partial fulfillment of requirements of Mini-Project (ISL64) of Sixth Semester B.E. It is certified that all corrections/suggestions indicated for internal assessment has been incorporated in the report. The project has been approved as it satisfies the academic requirements in respect of project work prescribed by the above said course.

Signature of the Guide	Signature of the HOD
Mrs. Lincy Meera Mathews	**Dr. Vijaya Kumar B P**
Assistant Professor	Professor and Head,
Dept. of ISE, RIT,	Dept. of ISE, RIT
Bangalore-54	Bangalore-54

Other Examiners

Name of the Examiners:	**Signature**
1.	
2.	

Acknowledgement

This project opened the door for us to a wider world of networking and cybersecurity. Our knowledge in this field has become significantly larger than it was before we decided to base our project off of it.
We are grateful to have completed the project under the guidance and supervision of our respected guide, Mrs. Lincy Meera Mathews. We would like to thank her for her constant supervision and encouragement. We would also like to thank Dr. Vijaya Kumar B P for allowing us this detailed form of learning.

Abstract

Network attacks have become one of the most important security problems in the world today. With a high increase in network traffic, hackers and malicious users plan new and different kinds of ways of network intrusion. Hence, network intrusion detection is a field of active research with various engineers and scientists trying to identify threats that computer networks face every day. A huge effort is required in the effective examination of network packets that comprise high dimensions. Therefore, there is a need for network intrusion detection systems that can detect and differentiate malicious attacks that can occur efficiently. Such predictive systems can be built by using the concepts of networking and machine learning. There are quite a few datasets available that are for training and testing machine learning based models. We acquired the UNSW-NB15 dataset prepared at ACCS (Australian Centre for Cyber Security) includes nine different types of network-based attacks. By using the above dataset we attempt to train and build a learning model that can first perform binary classification and then multi-class classification to determine the type of the attack accurately.

Table Of Contents

List of Figures

List of Tables

1. Introduction

1.1 Motivation

The choice of our topic falls under networking. Networking is one of the core subjects we study and is therefore being emphasized here. To be more specific, here we deal with cyber security.

Cyber security is imperative because government, military, corporate, financial, and medical organizations collect and store unusual amounts of data on computers and other devices. A significant portion of that data can be sensitive information for which unauthorized access or exposure could have negative consequences. Some threats to cyber security include Malware, Phishing, Trojans and attacks like DoS (Denial of Service), worms, shellcode and so on.

This project attempts to detect these attacks as they occur, so a proper defense mechanism can be set up against them before they cause any severe loss of resources or functionalities.

1.2 Scope

The data set used has nine types of attacks, namely – fuzzers, analysis, backdoors, DoS, exploits, reconnaissance, shellcode, worms and other generic attacks. Our work attempts to differentiate these attacks from normal data flow into a server. The different types of attacks are described below.

Fuzzing – is the automated process of entering random data into a program and analyzing the results to find potentially exploitable bugs.
Backdoors – A type of malware that gives cybercriminals unauthorized access to a website.
DoS (Denial of Service) – Hackers attempt to prevent legitimate users from accessing the service.
Exploits – An attack on a computer system that takes advantage of a particular vulnerability the system offers to intruders.
Reconnaissance – An attack in which the miscreant/hacker engages with the targeted system to gather information about the system’s vulnerabilities.
Shellcode – A set of instructions that executes a command in a software to take control of a compromised system.
Worms – A computer worm is a type of vicious software program whose function is to infect other systems while remaining active on infected systems.

1.3 Objectives

Our aim is to build an effective learning model that can identify malicious attacks within a stream of connections to a server and distinguish them from benign connections. Once an attack is identified, we attempt to determine the type of attack. The system can then take the required retaliatory actions against the detected malicious connections and block them.

1.4 Proposed Model

Our proposed model uses 47 parameters that are extracted from each connection to predict whether the connection is a part of an attack or not. These 47 features are elaborated in Section 3.1.4. The model includes two Random Forest classifiers, one that performs binary classification distinguishing between a normal and an attack connection, another that performs multi-class classification to determine the type of attack.

1.5 Organization of Report

In order to explain the developed system, the following sections are covered:

- **Literature Review**
- **System Analysis and Design** provides a detailed walk through of the software engineering methodology adopted to implement the model, an overview of the system and the modules incorporated into the system
- **Modelling and Implementation** provides a deeper insight into the working of the model. The various modules and their interactions are depicted using relevant descriptive diagrams.
- **Testing** the model to ensure bug/error free model along with the **Results** obtained. **Discussion** then provides detailed analysis on quality assurance measures.
- **Conclusion** about the Results obtained after successfully running the model and **Future Scope** of the model is highlighted.

2. Literature Review

There has been an abundance of work performed on intrusion detection.

“Research on Network Intrusion Detection System Based on Improved K-means Clustering Algorithm”, by Li Tian and Wang Jianwen is a paper that gave a new model of anomaly intrusion detected based on the K-means clustering algorithm. Because of the k-means algorithm’s shortcomings about dependence and complexity, the paper puts forward an improved clustering algorithm through studying the traditional means clustering algorithm. The new algorithm learns the strong points from the k-medoids and improved relations trilateral triangle theorem. The experiments proved that the new algorithm could improve accuracy of data classification and detection efficiency significantly. The results show that the algorithm achieves the desired objectives with a high detection rate and high efficiency.

Ahmad, Iftikhar, et al., projected a new-fangled intrusion detection structure based on K-nearest neighbor (KNN) classification algorithm in a wireless sensor network. This structure can isolate anomalous nodes from regular nodes by discerning their anomalous behaviors, and the authors investigated parameter selection and inaccuracy rate of the intrusion detection system.

Aburomman, Abdulla Amin, and Mamun Bin Ibne Reaz proposed a novel collaborative structure technique that employs PSO engendered weights to produce a collaborative of classifiers with improved precision for intrusion detection. Local uni-modal sampling (LUS) technique is employed as a meta-optimizer to discover healthier behavioral constraints for PSO. Besides, the current research exploited the KDD Cup ’99 dataset for discovering the intrusion detection.

Thaseen, Ikram Sumaiya proposed an intrusion detection model consuming chi-square feature selection and multi-class support vector machine (SVM). In their paper, a constraint tuning system is implemented for optimization of Radial Basis Function kernel parameter. The NSL-KDD dataset which is an enriched version of KDD Cup ’99 dataset was practiced in this paper.

Li, Longjie, et al., a unique hybrid model was proposed with the determination of identifying network intrusion meritoriously. In the proposed model, Gini index is accustomed to picking the most excellent division of features, the gradient based decision tree (GBDT) algorithm is implemented to sense network attacks. The particle swarm optimization (PSO) algorithm is exploited to augment the parameters of GBDT. The NSL-KDD dataset was used to assess the activity of the future technique.

3. System Analysis and Design

This section deals with the purpose of studying the system or its parts in order to identify its objectives. It also deals with the process of planning a modern system or replacing an existing system by defining its components or modules to satisfy the specific requirements.

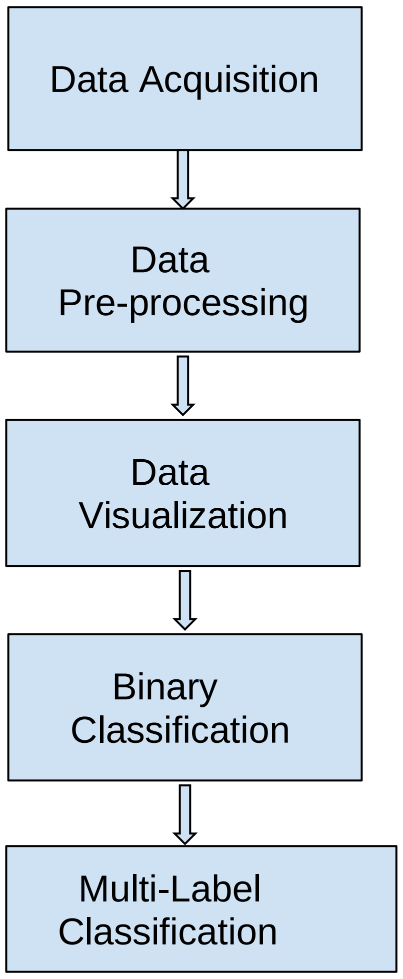

Figure 1 System Analysis & Design

The above figure explains the flow of the complete project.

3.1 Data Acquisition

The UNSW website provides datasets for network intrusion detection systems. We use machine learning techniques to classify the UNSW-NB15 dataset.

The dataset is a comprehensive network-based dataset (2.5 million in size) which reflects modern network traffic scenarios and a variety of low footprint intrusions.

In this section, the generation of UNSW-NB15 details and the description of the different features are presented.

3.1.1 An IXIA tool Testbed Configuration

According to the figure, the IXIA traffic generator is configured with the three virtual servers. The servers 1 and 3 were configured for normal spread of the traffic while server 2 formed the abnormal/malicious activities in the network traffic. Establishing the intercommunication between the servers, acquiring public and private network traffic, there were two virtual interfaces having IP addresses, 10.40.85.30 and 10.40.184.30. The servers were connected to hosts via two routers. The router 1 had 10.40.85.1 and 10.40.182.1 IP addresses, whereas router 2 was configured with 10.40.184.1 and 10.40.183.1 IP addresses. These routers were connected to the firewall device that is configured to pass all the traffic either normal or abnormal. The tcpdump tool was installed on router 1 to capture the Pcap files of the simulation uptime.

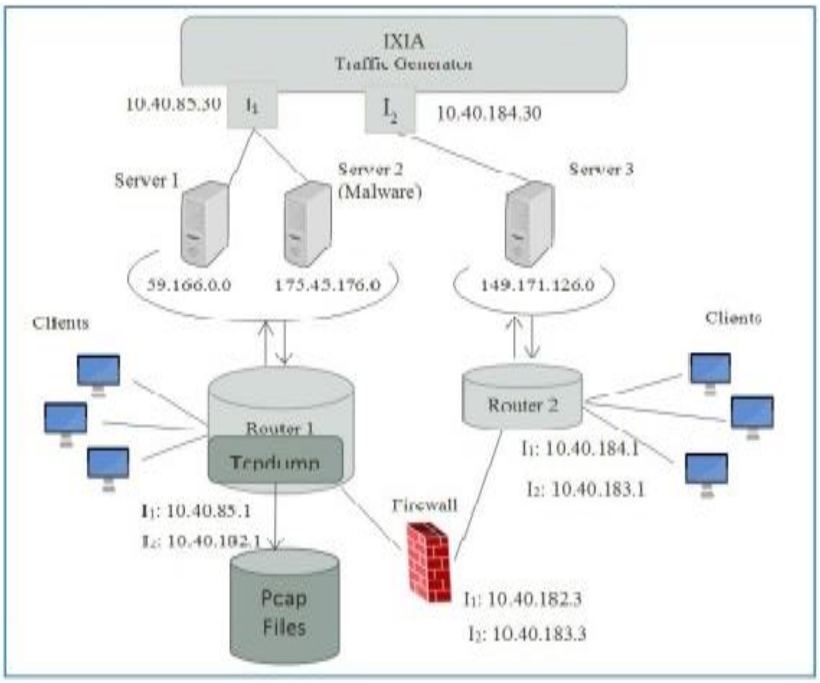

Figure 2 Testbed Visualization for UNSW-NB15

3.1.2 Architectural Framework

The whole architecture which was involved in generating the final shape of the UNSW-NB15 from pcap files to CSV files with 49 features (attributes in any CSV file) is presented in the figure. All the 49 features of the UNSW-NB15 data set are elaborated below along with the generation sequence explanation for understanding convenience.

When the simulation was running on the testbed presented in Fig. 1, the pcap files were generated by using the tcpdump tool. The features of the UNSW-NB15 dataset were extracted by using Argus, Bro-IDS tools and twelve algorithms were developed using c# programming language as shown in the figure. These tools were installed and configured on Linux Ubuntu

14.0.4. The detailed formatting description of the UNSW-NB15 data set is elaborated in the following section.

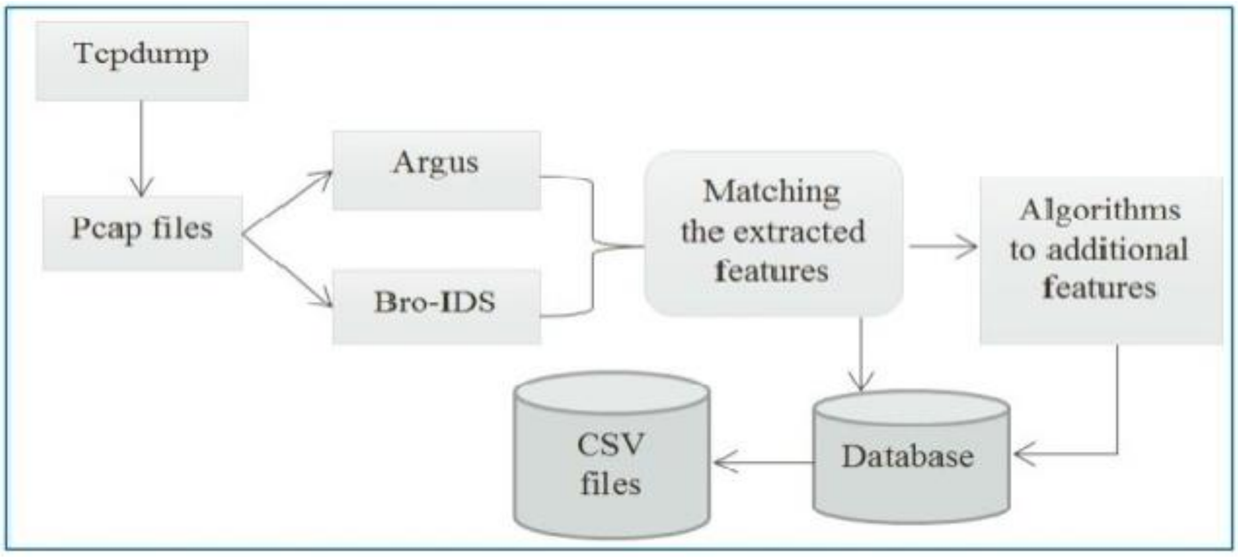
Tcpdump
Pcap files
Argus
Bro-IDS
Matching the extracted features
Algorithms to additional features
Database
CSV files

Figure 3 Framework Architecture for Generating UNSW-NB15 data set

3.1.3 The extracted features from the Argus and Bro-IDS Tools

Argus tool processes raw network packets (e.g., pcap files) and generates attributes/features of the network flow packets. The Argus tool consists of an Argus-server and Argus-clients.The Argus-server writes pcap files of receiving packets in Argus files in the binary format. The Argus clients extract the features from the Argus files.

Bro-IDS tool is an open-source network traffic analyser. It is predominantly a security monitor that inspects all network traffic against malicious activities. The Bro-IDS tool is configured to generate three log files from the pcap files. First, the conn file records all connection information seen on the pcap files. Second, the http file includes all HTTP requests and replies. Third, the ftp file records all activities of a FTP service.

3.1.4 The matched features of the Argus and Bro-IDS Tools

Finally, the output files of the two different tools, Argus and Bro-IDS were stored in the SQL Server 2008 database to match the Argus and Bro-IDS generated features by using the flow features as reflected below.

No.	Name	Type	Description

1	srcip	nominal	Source IP address
2	sport	integer	Source port number
3	dstip	nominal	Destination IP address
4	dsport	integer	Destination port number
5	proto	nominal	Transaction protocol
6	state	nominal	Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, and (-) (if not used state)
7	dur	Float	Record total duration
8	sbytes	Integer	Source to destination transaction bytes
9	dbytes	Integer	Destination to source transaction bytes
10	sttl	Integer	Source to destination time to live value
11	dttl	Integer	Destination to source time to live value
12	sloss	Integer	Source packets retransmitted or dropped
13	dloss	Integer	Destination packets retransmitted or dropped
14	service	nominal	http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service
15	Sload	Float	Source bits per second
16	Dload	Float	Destination bits per second
17	Spkts	integer	Source to destination packet count
18	Dpkts	integer	Destination to source packet count
19	swin	integer	Source TCP window advertisement value

20	dwin	integer	Destination TCP window advertisement value
21	stcpb	integer	Source TCP base sequence number
22	dtcpb	integer	Destination TCP base sequence number
23	smeansz	integer	Mean of the ?ow packet size transmitted by the src
24	dmeansz	integer	Mean of the ?ow packet size transmitted by the dst
25	trans_depth	integer	Represents the pipelined depth into the connection of http request/response transaction
26	res_bdy_len	integer	Actual uncompressed content size of the data transferred from the server's http service.
27	Sjit	Float	Source jitter (mSec)
28	Djit	Float	Destination jitter (mSec)
29	Stime	Timestamp	record start time
30	Ltime	Timestamp	record last time
31	Sintpkt	Float	Source interpacket arrival time (mSec)
32	Dintpkt	Float	Destination interpacket arrival time (mSec)
33	tcprtt	Float	TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'.
34	synack	Float	TCP connection setup time, the time between the SYN and the SYN_ACK packets.
35	ackdat	Float	TCP connection setup time, the time between the SYN_ACK and the ACK packets.
36	is_sm_ips_ports	Binary	If source (1) and destination (3)IP addresses equal and port numbers (2)(4) equal then, this variable takes value 1 else 0
37	ct_state_ttl	Integer	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
38	ct_flw_http_mthd	Integer	No. of flows that has methods such as Get and Post in http service.

39	is_ftp_login	Binary	If the ftp session is accessed by user and password then 1 else 0.
40	ct_ftp_cmd	integer	No. of flows that has a command in ftp session.
41	ct_srv_src	integer	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
42	ct_srv_dst	integer	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
43	ct_dst_ltm	integer	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
44	ct_src_ ltm	integer	No. of connections of the same source address (1) in 100 connections according to the last time (26).
45	ct_src_dport_ltm	integer	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
46	ct_dst_sport_ltm	integer	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
47	ct_dst_src_ltm	integer	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).
48	attack_cat	nominal	The name of each attack category. In this data set , nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellcode and Worms
49	Label	binary	0 for normal and 1 for attack records

Table **1**

3.2 Data Pre-Processing

In this stage we'll be converting categorical data into binary vectors and remove unwanted features like column id. The dataset contains 3 nominal features that require encoding namely proto (protocol field), state and service. We use One Hot Encoding to achieve this.

One Hot Encoding splits the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains 0 or 1 corresponding to which column it has been placed.

For example – The protocol field has 133 unique values. One-hot encoding all of them will add an additional 132 features to the dataset, so, we consider only the four most occurring values and rename the rest of them as others.

The below tables show the four most occurring values for the respective features.

Proto – denotes the transaction protocol being used.

Tcp	79946
udp	63283
unas	12084
arp	2859
Name: proto, dtype: int64	

state – The state and its dependent protocol, e.g. ACC, CLO.

INT	82275
FIN	77825
CON	13152
REQ	1991
Name: state, dtype: int64	

service – http, ftp, ssh, dns.

dns	47294
http	18724
smtp	5058
ftp-data	3995
Name: service, dtype: int64	

After performing One Hot Encoding, we split the dataset into training and test sets and then perform feature scaling using the built in StandardScaler() from sci-kit learn.

3.3 Data Visualization

Visualizing the data allows us to explore the data and uncover the deep insights.

The UNSW-NB15 dataset has a large number of variables. In other words, it has a high number of dimensions along which the data is distributed. Visually exploring the data can be challenging.

Hence we use some of the known dimensionality reduction techniques to achieve visualization. In particular we'll be using PCA and t-SNE.

3.3.1 Principal Component Analysis (PCA)

It is a statistical procedure that uses an orthogonal transformation which converts a set of correlated variables to a set of uncorrelated variables.

The below figure shows how the data is correlated using PCA and setting the number of components parameter equal to 2.

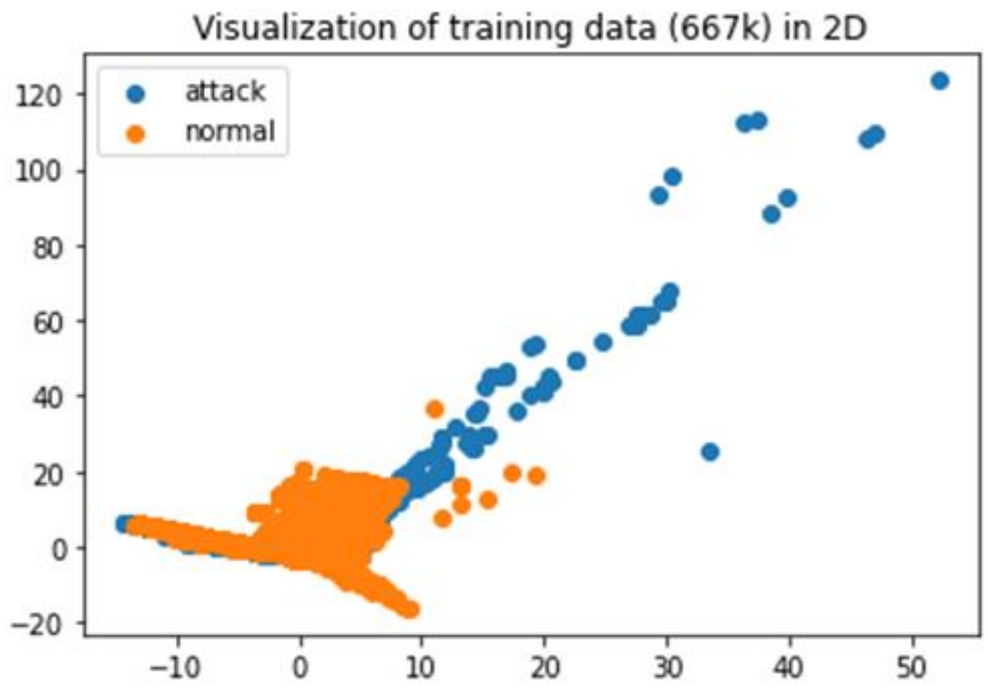

Figure 4 Visualization of training data using PCA

3.3.2 T-distributed Stochastic Neighbor Embedding (t-SNE)

It is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions.

The below figure shows the plot of our data using t-SNE (1 – attack and 0 – normal).

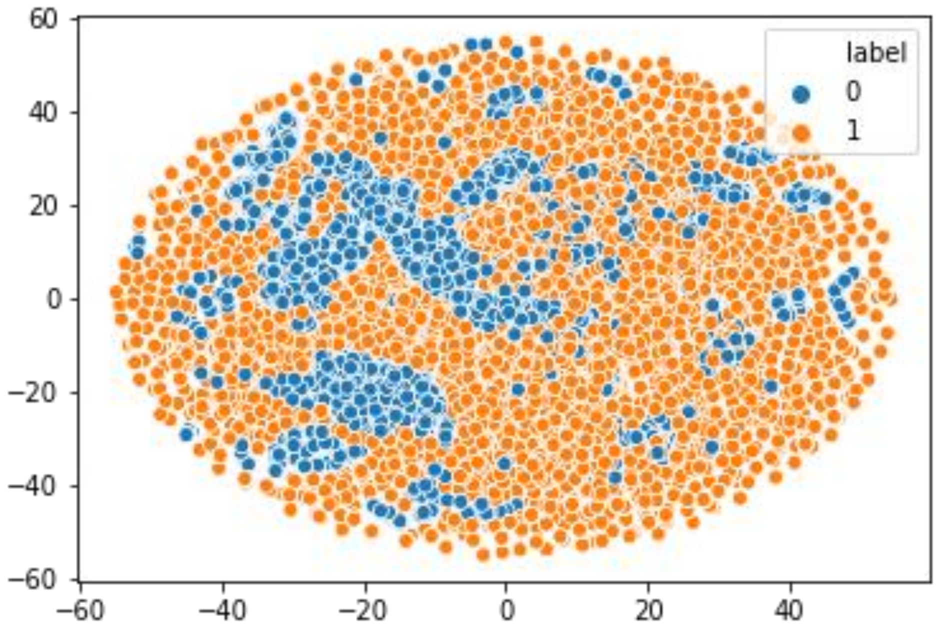

Figure 5 Visualization of training data using t-SNE

3.4 Binary Classification

In this stage we'll be classifying if the network stream is an attack or not. Since we have only two groups to classify into (1 – attack and 0-normal), we'll be using binary classification to achieve this.

Binary classification is the task of classifying the elements of a given set into two groups (predicting which group each one belongs to) on the basis of a classification rule.

3.5 Multi-label Classification

Once we detect if the network stream is an attack , the next step is to classify the type of attack.

The dataset has 9 different types of attacks –

- generic
- exploits
- fuzzers
- dos
- reconnaissance
- analysis
- backdoor
- shellcode
- worms

Therefore we use Multi-label classification to achieve this.

In Multi-label classification, the training set is composed of instances each associated with a set of labels, and the task is to predict the label sets of unseen instances through analyzing training instances with known label sets.

4. Modelling and Implementation

4.1 The Modelling Process

After successfully completing data pre-processing and cleaning we move on to labelled.

Different machine learning models exist and the choice of which one to use generally depends on the problem at hand. No machine learning model works best for all kinds of problems. So, our job in this stage is to test multiple models and fine tune parameters to squeeze out every bit of accuracy.

The first stage involves building a binary classifier to classify a connection request into attack and normal.

4.1.1 Binary Classifier

For this stage we trained and tested six different classifiers listed below

- K-nearest neighbours
- Logistic regression
- Gaussian naïve bayes
- Decision tree
- Random forest
- Gradient boost

We observed that all the classifiers gave better AUC (Area under ROC curve) scores on the cross-validation set than on the test set.

The cross-validation set was split from train data provided unlike the test set which was provided separately, it led us to believe that the training set did not contain enough records to represent the test set.

We loaded the entire raw dataset (2.5 million records) before it was sampled for training and test data.

The results of all classifiers comprehended in a table.

Classifier	Training accuracy(%)	Test accuracy(%)	Test AUC score(%)
K- Nearest Neighbours	99.32239689724369	98.9867494544554	97.73011996638872

Logistic Regression	98.90065720601954	98.91925940923714	98.34402242948844
Gaussian Naïve Bayes	89.45005864161425	89.32262491282535	92.94064893852731
Decision Tree	99.98575203745864	99.18292051922342	98.16183046183347
Random Forest	99.98440223048104	99.3651436413129	98.53840436080699

Table 2

Observing the precision and recall scores of the above classifiers, Random forest classifier gives the best performance.

4.1.2 Multi-Class Classifier

After detection of an attack, the next step is classifying the type of attack.

For this stage we trained and tested four different classifiers listed below

- K-nearest neighbours
- Gaussian naïve bayes
- Decision tree
- Random forest

Classifier	Accuracy Score(%)

Random Forest	89.45640163717572
Decision Tree	88.6502637844904
Gaussian naïve bayes	69.246930295532
KNN	85.73851876060196

Table 3

Observing the Accuracy scores of different classifiers Random Forest once again gave the best score for multi-class classification for DOS/DDOS attacks.

4.2 Use Case Diagram

This system is designed to combat DdoS attacks on the Internet in a modern collaborative way. In this approach, the system collects network traffic samples and classifies them.

After distinguishing the network request based on certain information extracted from the packet as a “bad” connection/attack or a “good’’ /normal connection.

After detection of an attack,the next step is classifying the type of attack. This system is capable of identifying 9 different types of attacks.

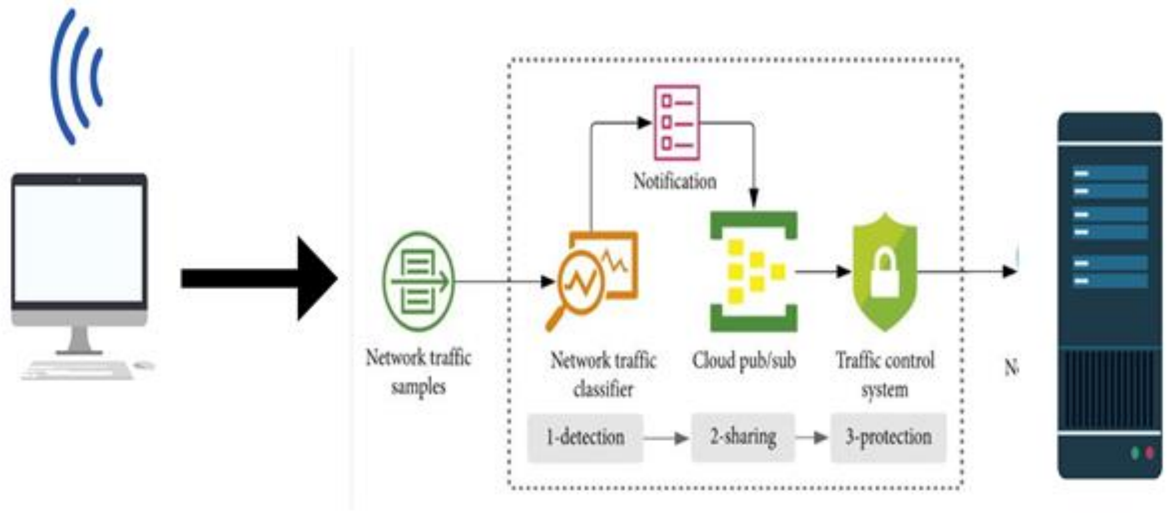

Figure 6 System to combat DdoS attacks (Use case)

4.3 Sequence Diagram

Sequence diagrams describe interactions among classes in terms of exchange of messages over time. They're also called event diagrams.

A sequence diagram is a good way to visualize and validate various runtime scenarios.

The below diagram helps to visualize the process of detecting a DOS/DDOS attack and further classifying it into a certain type of attack.

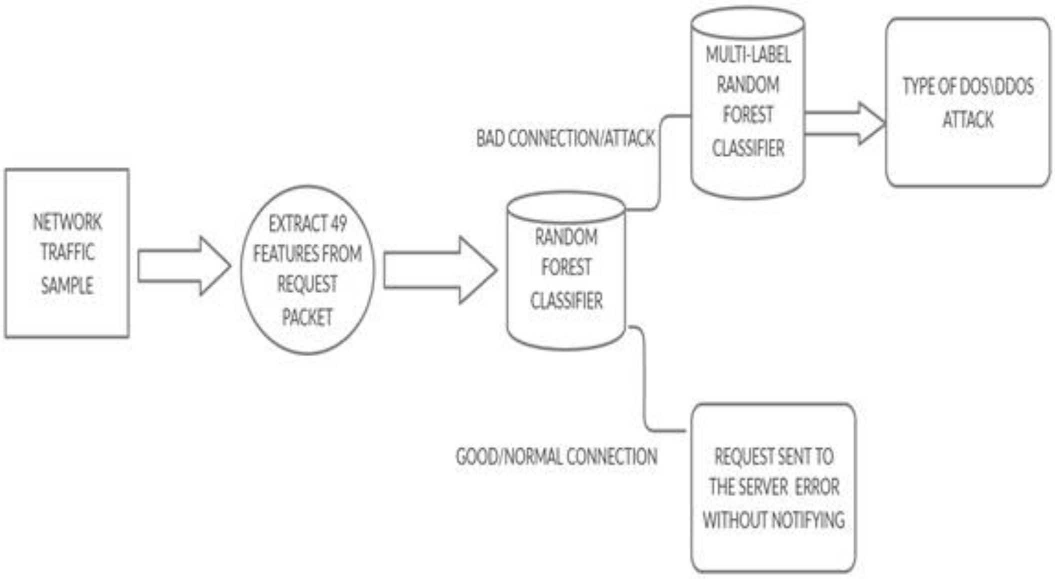
NETWORK TRAFFIC SAMPLE
EXTRACT 49 FEATURES FROM REQUEST PACKET
RANDOM FOREST CLASSIFIER
BAD CONNECTION/ATTACK
MULTI-LABEL RANDOM FOREST CLASSIFIER
TYPE OF DOS\DDOS ATTACK
GOOD/NORMAL CONNECTION
REQUEST SENT TO THE SERVER ERROR WITHOUT NOTIFYING

Figure 7 Process of classifying a DoS/DdoS attack

5. Testing, Results and Discussion

5.1 Testing

The dataset we used to train the model was given by the UNSW website and we are grateful to them.

The UNSW website provides separate datasets for training and testing that are sampled from the larger dataset (2.5 million in size) for network intrusion detection as training and test set.

Dimensions and features of the data set are:

- Training set size : (175341, 45)
- Test set size : (82332, 45)
- No. of attacks : 119341
- No. of normal connections : 56000

We observed that all the classifiers gave better AUC (Area under ROC curve) scores on the cross-validation set than on the test set. The cross-validation set was split from train data provided unlike the test set which was provided separately, it led us to believe that the training set did not contain enough records to represent the test set.

We loaded the entire raw dataset (2.5 million records) before it was sampled for training and test data.

For Multi-Class classification the same dataset was used but the models were only sampled and trained which were already 25abelled as an attack in the dataset.

Dimension of the dataset : (321283, 49)

There are 9 different types of DOS/DDOS attacks listed with their frequency below

1.	generic	215481
2.	exploits	44525
3.	fuzzers	24246
4.	dos	16353
5.	reconnaissance	13987
6.	analysis	2677
7.	backdoor	1795
8.	shellcode	1511
9.	backdoors	534
10.	worms	174

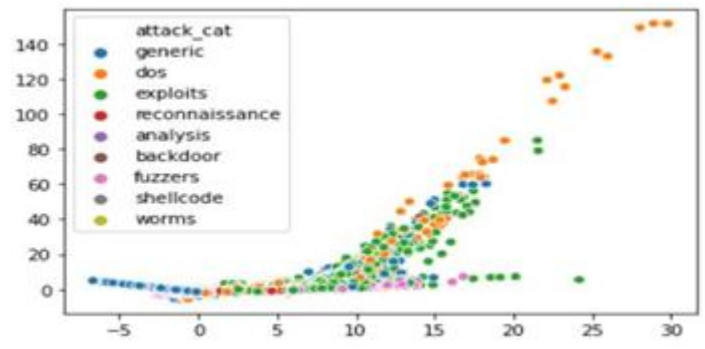

Figure 8 Visualization of attack data using PCA

The connections were classified as testing and training set with a ratio of 20:80.This was done using sklearn which consists of a library called test_train_split.

Dimensions of the training and testing set are:

- Training size: (257026, 47)
- Test set size: (64257, 47)

5.2 Results

The model is trained using sklearn 'model.fit' and 'model.predict' is used for prediction.

<u>Binary Classifier Results for Cross-validation Set and Test Set</u>

Classifier	Training accuracy(%)	Cross-Validation	Test AUC score(%)
K- Nearest Neighbours	94.68288495953937	92.28249732169042	83.08594954700574

Logistic Regression	90.54291497581939	90.84629439410352	79.66975980215635
Gaussian Naïve Bayes	82.81551509492686	82.55159075130324	78.1621018265243
Decision Tree	99.82288543088177	94.34012175086633	85.47612735827683
Random Forest	99.78912104371662	95.09403148435208	86.10135039742823
Gradient Boost	92.71607962095093	92.78187450249277	84.47216798109324

Table 4

<u>Binary Classifier Results for full UNSW-NB15 Dataset</u>

Classifier	Training accuracy(%)	Test accuracy(%)	Test AUC score(%)
K- Nearest Neighbours	99.32239689724369	98.9867494544554	97.73011996638872
Logistic Regression	98.90065720601954	98.91925940923714	98.34402242948844
Gaussian Naive Bayes	89.45005864161425	89.32262491282535	92.94064893852731
Decision Tree	99.98575203745864	99.18292051922342	98.16183046183347
Random Forest	99.98440223048104	99.3651436413129	98.53840436080699

Table 5

<u>Multi-Class Classifier Results</u>

Classifier	Accuracy Score(%)
Random Forest	89.45640163717572

Decision Tree	88.6502637844904
Gaussian naive bayes	69.246930295532
KNN	85.73851876060196

Table 6

5.3 Discussion

We observed that the Random Forest classifier gave the best results for both binary and multi-class classification. Though Logistic Regression, Decision Tree and Gaussian Naive Bayes classifiers also gave quite similar results for binary classification, they weren't as quick as the Random Forest classifier.

With a test accuracy of 99.36% and an AUC-ROC (Area under ROC curve) score of 0.985 we can say that binary classification is done quite successfully by the Random Forest Classifier. However, when it comes to multi-class classification to determine the type of attack, the test accuracy obtained is 89.46%, which is at best mediocre and can be further improved upon.

The training time depends on the number of times the classifier needs training. The training can be speeded up by removing the overlapping data and retaining only training samples adjacent to the decision boundary.

6. Conclusion and Future Work

Critical services are often badly affected by malicious attacks, in spite of the conventional deployment of network attack prevention mechanisms such as Firewall and Intrusion Detection Systems.

Some intrusion detection systems detect only attacks with known signatures. Predicting the future attacks is impossible. Hence, the system must be trained and tested in such a way that it learns by observing the aberrant patterns associated with the network traffic and classify the incoming traffic as an attack or normal.

Our proposed model can effectively differentiate between normal and malicious connections as shown above. The overhead of this entire process can be further reduced by performing feature reduction as not all 47 features might be required for classification. However, the accuracy of the system should be maintained while doing so. The multi-class classifier used in our model can be further improved upon for better identification of the type of attack. This information will help in deploying the right countermeasures to combat or block the right type of attack.

Once the detection model is well developed it can be further tested in a simulation environment. Then the required hardware can be developed which has the detection model embedded in it and can be installed in and around the parts of a network server detection and mitigation of malicious attacks.

7. References

1. L.Tian and W. Jianwen, "Research on network intrusion detection system based on improved k-means clustering algorithm," in Computer Science-Technology and Applications, 2009. IFCS T A '09. International Forum on, Dec. 2009, pp. 76-79.

2. A. A. Aburomman and M. Bin Ibne Reaz, "Survey of learning methods in intrusion detection systems," *2016 International Conference on Advances in Electrical, Electronic and Systems Engineering (ICAEES)*, Putrajaya, 2016, pp. 362-365, doi: 10.1109/ICAEES.2016.7888070.

3. I.S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of PCA and optimized SVM," *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, Mysore, 2014, pp. 879-884, doi: 10.1109/IC3I.2014.7019692.

4. Li, Longjie & Yu, Yang & Bai, Shenshen & Cheng, Jianjun & Chen, Xiaoyun. (2018). Towards Effective Network Intrusion Detection: A Hybrid Model Integrating Gini Index and GBDT with PSO. Journal of Sensors. 2018. 1-9. 10.1155/2018/1578314.

5. Ahmad, Iftikhar & Abdullah, Azween & Alghamdi, Abdullah & Hussain, Muhammad & Nafjan, Khalid. (2011). Intrusion Detection Using Feature Subset Selection based on MLP. Scientific Research and Essays. 6. 10.5897/SRE11.142.

6. N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," *2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, ACT, 2015, pp. 1-6, doi: 10.1109/MilCIS.2015.7348942.

7. J. Zhang and Z. Mohammad, "Anomaly based network intrusion detection with unsupervised outlier detection", *Communications 2006. ICC'06. IEEE International Conference on*, vol. 5.

8. R. Heady, G. Luger, A. Maccabe and M. Servilla, "The architecture of a network level intrusion detection system" in Tech. rep. Computer Science Department University of New Mexico, New Mexico, 1990.

9. M. Aydin, M. Ali, A. Halim Zaim and K. Gokhan Ceylan, "A hybrid intrusion detection system design for computer network security", *Computers & Electrical Engineering*, pp. 517-526, 2009.

10. Axelsson Stefan, "Intrusion detection systems: A survey and taxonomy", *Technical report*, vol. 99, 2000.