International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

Network Intrusion Detection System Using Single Level Multi-Model Decisiontrees

Ishitva Verma
Bachelor of technology in computer science and engineering with specialization in informationsecurity (Vellore Institute of Technology)

Submitted: 02-02-2021 Revised: 18-02-2021 Accepted: 22-02-2021

CHAPTER- I

1.1 Introduction to Area of Study

Intrusion is considered to compromise the integrity, confidentiality, or availability of valuable assets on the computer systems. An intrusion detection system (IDS) audits the traffic flowing in the network for suspicious activity. It then alerts the system administrator when any malicious activity is discovered in the network. The primary functions of intrusion detection systems are discovering anomalies and producing detailed reports for the intrusions discovered . In some cases, if malicious activity or intermittent traffic is detected in the network, intrusion detection systems are programmed to take actions like deterring traffic sent from incredulous IP addresses. An IDS is a programmable software that is developed to detect intrusions within the network. It is employed to hunt and pinpoint the intruders causing chaos within the network. The main principles of IDS are integrity, availability, confidentiality and accountability. An IDS is built using both software and hardware. It can detect highly dangerous intrusions within the network. [6]The main purpose of IDS is to detect unauthorized packets and malicious communications that happen in computer systems and networks. The most vital ingredient for the success of intrusion detection systems is feature selection. We can classify the Intrusion Prevention Systems into 4kinds:

a. **Network Based Intrusion Prevention System**– It detects and identifies any suspicious or unwanted traffic flowing within the network by evaluating the activity of various protocols for the network.

b. **Wireless Intrusion Prevention System**– It detects and identifies any suspicious or unwanted traffic flowing within the network by evaluating the activity of the wireless protocols within thenetwork.

c. **Network Behaviour Analysis**– It evaluates the entire network to spot for vulnerable threats. For example Remote Code Executions, Denial ofService.

d. **Host Based Intrusion Prevention System**– In this type of system a software is installed on the host to monitor for any pernicious activity. This is done by evaluating the hostsystem.

1.2 Relevance to Practical Field :

Intrusions are considered as a sequence of steps taken to compromise the integrity, confidentiality, or availability of valuable assets on the computer systems. Intruders gain unauthorized access to the resources available on the system. They use all kinds of techniques to gain access to confidentialinformation and manipulate the data available on the system. This can sometimes damage the system and render it worthless.[9] An IDS can be considered to be a blend of software and hardware units that can be used to identify and pinpoint unauthorized experiments to gain access to the network. All the network related activities can be audited by an IDS which in turn can be used to suspect the traces of invasions within the network. The end goal of an IDS is to trigger alerts when a suspicious activity has occurred by notifying the System Administrator. Intrusion detection techniques can be classified into 2 types :

A) **Anomaly Detection**: In this kind of detection the system alerts malicious tasks by identifying deviations that is how differently are the network activities occurring as compared to regularpatterns.

B) **Misuse Detection**: In this kind of system intrusions are detected on the basis of already known patterns i.e. previously occurred malicious activity. This method can be used to identify and pinpoint known attack patterns more accurately. Misuse Detection has a disadvantage over Anomaly Detection i.e. only those kinds of intrusions can be identified

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

which can correlate to already known patterns of attack. An ideal IDS will monitor all the happenings within the network and then decide whether those happenings are malicious or normal. The decision is based on system availability, confidentiality and integrity of the information resources. The various aspects to be considered while building and IDS are the following: data pre-processing, collection of data, detecting intrusions and curating reports with appropriateresponses.[10]

An Intrusion Detection System works in the following manner: Collecting Data,SelectingFeatures,AnalysingtheData,andtheActionstobePerformed.

a. **Collecting Data**: We need to gather reports on the traffic flowing in the network like hostsa live, protocols used and the various forms of traffic flowing.
b. **Selecting Features**: After collecting a huge amount of data, the next step is to pick all those required features which we want to workupon.
c. **Analysing the Data**: In this step the data about the features which are selected data is evaluated to help us determine if the data is unnatural ornot.
d. **Actions to be Performed**: When a malicious attack has taken place the system administrator is alarmed or notified by the IDS. The details about the type of attack are also provided by the IDS. The IDS closes the unnecessary network ports and processes to further mitigate the attacks fromhappening.

1.3 Importance of the study proposed :

Intrusion detection is one of the key interests in network administration and security. There is a need to protect the networks from known vulnerabilities and at the same time take steps to identify new and unknown but potential device abuses by creating more robust and effective systems for intrusion detection.

An intrusion detection system (IDS) audits the traffic flowing in the network for suspicious activity and signals when any malicious activity is discovered. For most Intrusion Detection Systems discovering anomalies and generating reports for the intrusions discovered are the elementary responsibilities. In some cases,if a malicious activity or intermittent traffic is detected in the network, intrusion detection systems are programmed to take actions like deterring traffic sent from incredulous IP addresses. An IDS is a software that is developed to detect intrusions within the network. It is employed to hunt and pinpoint the intruders within thenetwork.

NETWORK-BASED IDS: The data fed into this type of IDS is obtained from the flow of packets in the network having data.

MISUSE DETECTION: This type of IDS has the functionality to trace previously recognized patterns of pernicious activity going on in the network and spotintrusions.[8]

CHAPTER- II

2.1 Brief Background of the Problem:

Intrusion Detection System (IDS) is a security mechanism that behaves like a layer that protects the internal structure. Over many years, IDS technology has been improved tremendously to be at par with the cyber crime advancements. Since the technology came into picture in the mid-’80s, developers have performed experiments and research to boost the ability of identifying attacks without affecting the performance of the network.

A very major threat to the security of a network is an intrusion which can be anything such asdenial of service , brute force, or a network-internal infiltration. As the network topologies change and update, it has become crucial to shift to a dynamic techniquein order to detect and avoid intrusions [5]. Various sorts of research has been a part of this field, and it has been accepted universally that traffic compositions and interventions are not captured by static datasets.

We need the changeable, revamp-able and expandable datasets to study and handleexpertintruders who can easily bypass normal intrusion detection systems (IDS). This is where we need Machine Learning algorithms. We will use machine learning technique like the decision tree that can be deployed to make robust IDS and detect different types of intruder attacks on our system.

2.2 Earlier Works on this type of problem:

A recent study presents a taxonomy of contemporary IDS, a comprehensive review of notable recent works, and an overview of the datasets commonly used for evaluation purposes. It also presents evasion techniques used by attackers to avoid detection and discusses future research challenges to counter such techniques so as to make computer systems more secure. [7]

There are numerous methodologiesthat are being used in intrusion detection systems, but due to some reasonsall these methodshave not been considered ideal till date. The systems may generate false alarms in several anomaly based intrusion detection systems. The false alarm rate in

denoting intrusive activities can be decreased with fuzzy logic.A group of useful fuzzy rules can interpret the normal and abnormal changes in communication networks. Hence some trick is necessary for efficientand reliable security to keep a lookout for the anomalous change in the network. A not so recent study has presented the methodologies and good fuzzy classifiers using genetic algorithm which are prioritizing current development efforts and the solution of the problem of Intrusion Detection System to offer a realscenario of intrusion detection.[4]

Honey pots are effective detection tools to sense attacks such as port or email scanning activities in the network. Some features and applications of honey pots are explained in a recent paper. HP is mainly a heuristic approach and is based on the concept of bait and trap. Nevertheless, industry sector is very attracted to this concept. There are a number of products available that use the HP to trap undetected intrusion attempts. Generally speaking, HP is a deception based approach to detect actions of a deceitful enemy (the intruder). [1]

Well known IDS system Snort and the new coming IDS system Suricata were tested and analysed in a recent research. Both Snort and Suricata were implemented on three different platforms (ESXi virtual server, Linux 2.6 and FreeBSD) to simulate a real environment. [3]

A recent article presents several specific aspects which make it challenging for an IDS to monitor and detect web attacks. The article also provides a comprehensive overview of the existing detection systems exclusively designed to observe web traffic. Furthermore, various dimensions have been identified for comparing the IDS from different perspectives based on their design and functionalities. A framework has been proposedfor a web IDS with a prevention mechanism to offer systematic guidance for the implementation of the system. [2]

2.3 Objectives of the proposed study:

The IDS we will be developing is a network-based IDS that is programmed to detect any misuse of the network resources (misuse detection) i.e. it detects malicious packets flowing in a network.

We will be building a classifier using Decision Trees. They help in increasing the accuracy at which intrusions are detected. Before building the classifier wewill be required to select the most optimal features using Feature selection. We will be employing the concept of Recursive Feature Elimination(RFE). Because of RFE the attacks are detected more efficiently even in highly congested networks. It also leads to a lesser number of false positives and therefore a lower rate of false alarms. Also, The time taken to detect attacks can be cut down significantly by using databases for storage and the concept of Dynamic multi-boosting. The proposed network intrusion detection system can classify packets in real-time based on the packets collected from the networkflow.

2.4 Identification and exact Problem Definition:

Lately, intrusion detection has become a major concern in the field of network security and administration. Considering intrusion as a security threat, a network needs a system which protects it from known vulnerabilities and at the same time take measures to detect unknown ones for efficient functioning of the network. The detection system must be accurate upto some extent in detecting attacks with a possible minimum number of falsepositives.

In this Project we aim to create an Intrusion Detection System, using an ML model that has maximum accuracy. Now, there are a variety of ML models that have been used to detect intrusion attacks, but most of them are not industry-ready, as in the accuracy is not more than 98 percent. After going through some research papers we came across this algorithm, known as Multimodal decision trees to build a classifier and to detect which particular type of attack will occur on which protocol with extreme accuracy behind the detection of the intrusion. Multimodal is a decision tree having several modes or maxima, that uses the multimodal web mining framework. Mainly this system will be a network-based IDS and programmed to detect any misuse of the network resources.

CHAPTER- III

3.1 IDS Methodology adopted for the AI Model:

The primary goal is to design a plan for detecting intrusions within the system with the least possible number of features within the dataset. Based on the data from previous papers published, we can tell that only a subdivision of features in the dataset are derivative to the Intrusion Detection System. We have to cut back the dimensionality of the dataset to build an improved classifier in a justifiable amount of time. The approach we are going to use has a total of 4 stages : In the first stage, we pick out the significant features for every class using feature selection.In the next we combine the various features, so that the final cluster of features are optimal and relevant for each

attack class. The third stage is for building a classifier. Here, the optimal features found in the previous stage are sent as input into the classifier. In the last stage, we test the model by employing a testdataset.

The proposed approach consists of 4 fundamental stages:

1. Featureselection
2. Combining the optimal features
3. Building a classifier
4. Evaluation

3.1.1 Feature Selection:

We already know that network intrusion systems require bulk amounts of raw data to be dealt with. The most crucial building block of developing such a sophisticated system is Feature Selection. One thing which has a major effect on the dimensionality of the dataset is the complexity of the model. It results in obtaining high computational time and costs and also low classification accuracies. A significant number of techniques and approaches are used to get rid of repetitive and insignificant features.

These methods are chosen in such a way that we select only those features which are optimal and can be used to enhance the performance of the model. The 2 basic methods used for feature selection are: wrapper methods and filter methods.

Wrapper algorithms- In these types of algorithms a learning algorithm is used to make a decision for the features. In the

Filter algorithms - In these types of algorithms a measure which is not dependent on the others is employed. Some of the measures are consistency, information and distance. A relationship between the set of features can be estimated using these measures.

We will be using Information Gain (IG) to select the subset of relevant features in this project. Information Gain often costs less and is faster as compared to the wrapper methods. We calculate Information gain for all the attributes present in the training dataset. It is calculated for each class separately. In the next step the values of the information gain are ranked i.e. the feature with the highest information gain being at rank 1. It means that this particular feature can distinctively classify for the particular class. If the value of the Information gain is less than the fixed threshold value for a particular feature, that feature can be eliminated from the feature space. A better and optimal threshold value is obtained by examining the distribution of Information Gain for each attribute tested with different values of fixed threshold on the trainingdataset.

In the next stage the feature selection for each category can be broken down into 4 stages:

Stage 1: We divide the training dataset into 4 datasets. The training dataset is divided into 4 datasets in such a way that each dataset consists of records belonging to the same attack class along with some of the records of the original dataset. This stage is performed so that the feature selection method is unbiasedwhile selecting features for frequently occurring attacks in the dataset. This step helps us in preventing unwanted bias and to make sure the results retrieved are accurate.

Stage 2: In this stage the datasets for each attack class are sent separately as input into the method used to calculate the information gain. The output of this method gives us the most significant features for each attack type.

Stage 3: In the third stage we generate a list of ranked features for each attack class. Now we eliminate all the irrelevant features from the list in accordance with the fixed threshold values.

3.1.2 Combining Optimal Features:

In the last stage of feature selection, we combine the list of features generated foreachattackintoasinglelist.Forsomeoftheattackclassesthehighestranks
i.e. the top 4 features chosen for classification. But for some types of attack classes we can only take 1 feature since that particular feature is at the top of the rank table and the remaining features are at the very bottom of the table. So, the final set of combined optimal features can be used to entirely distinguish the attack types.

3.1.3 Building a Classifier:

The obtained subset of relevant features of each attack type in the above step is now used to build a classifier in this phase. As the model that is going to be used is a decision tree classifier, the dataset containing only the obtained subset of features is used to build 4 classifiers i.e. 1 for each attack type. This creates trees with only the relevant features of each attack type in each decisiontree.

3.1.4 Model Used: Decision Tree Classifier:

A Decision tree is an algorithm which takes decisions at each node of the tree and is widely used for regression and classification. It is a supervised learning algorithm in Machine learning where which attribute should be at which node is learnt by using a set of labelled examples. The main advantage in using decision trees is that they can be trained very easily and they can even classify non linear data. It is more productive than

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

most of the classification algorithms in ML like K-Nearest Neighbours in most of the cases. The common measures used to select attributes at each node in Decision trees are Info gain and Gain ratio.[13]

Entropy: A measure of impurity which is defined as the degree or the amount of uncertainty in the randomness ofelements

E(S) = Σ−= 12 Homogeneity of a sample is calculated by the Entropy.

- Entropy = 0, if the sample is absolutelyhomogeneous
- Entropy = 1, if the sample is uniformlydivided.

Information Gain: The relative variation in entropy can be measured using Information Gain which corresponds to an independent attribute.While constructing a Decision tree, an attribute at a node is selected by calculating the information gain of all attributes and determining the attribute which has the maximum information gain. Info gain calculates the information encompassed by eachfeature.

Gain (D, Attr.) = Entropy(D) – Entropy (D, Attr.)

Where Gain (D, Attr.) is the information gain of the attribute Attr. Entropy(D) is the entropy of the complete set, and after implementing the attributes entropy is calculated using Entropy(D, Attr). Overfitting is one of the main problems ofthe decision tree as it constructs its nodes by thoroughly going through the entire training set which may result in a low accuracy while using unknown data like the testset.

Decision Tree works in the followingway:

a.The best attribute is identified for separating the records using the Attribute Selection Steps (ASM).

b. Now the dataset is split into smaller subsets by using the previously selected attribute as the decision node.

c. A tree is built by the algorithm by iteratively executing this cycle before any one condition fits in :

1. There are no attributes left overanymore.
2. No more instances doexist
3. The Tuples generated have the same value of the attribute.

Entropy of attribute:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

Information gain:

$$Gain(A) = Info(D) - Info_A(D).$$

Entropy:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

Split info:

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

Gain ratio:

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

3.2 Data Collection (Dataset):

An accurate intrusion detection system requires a good set of labelled examples to sharply detect intrusions in the real world. In order to achieve higher accuracies while modelling an intrusion detection system, a dataset that represents the real world more accurately can only be used. The NSL-KDD data set is a polished version of its forerunner KDD 99 data set.

NSL-KDD data set is employed to check the

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

validity and performance of the different classification algorithms in classifying the attacks in the traffic patterns of the network.

Description:

It has all the basic records of the complete KDD data set required for various models. There are a bunch of files to download on the internet for ML practitioners and researchers.

Each record has a total of 41 attributes for identifying the various features of the network packets and a label is named to every packet. This label is used to classify that record either as an attack category or as none/normal. The fundamental points of the features are their name, their description, and sample values. The following table contains information about all the 41 attributes present within the NSL-KDD data set. [14] The 42nd attribute unfolds data about labels of each record which represents the packets in the network and that they're categorized as 1 ordinary class and 4 attack-type classes. The 4 attack classes used to classify the records in the dataset are DoS, Probe, R2L, and U2R. The mentioned table is as follows:

Attribute No.	Attribute Name	Description	Sample Data
1	Duration	Time duration of the connection	0
2	Protocol_type	Protocol utilized for the Connection	Tcp
3	Service	Destination network service used	ftp_data
4	Flag	Status of the connection – Normal or Error	SF
5	Src_bytes	Number of information bytes transferred from sorce to destination in single connection	491

Table. 3.1

The advantages in adopting the NSL-KDD dataset are

- There won't be any similar records that have a stronger rate of decline in the testsample.
- As compared to the KDD-99, the NSL-KDD shows a lesser number of data points.Therefore using them for training machine learning models is less costlyalgorithmically.
- The number of records chosen for each tough group of rates is inversely proportional to the percentage of records in the original KDDdataset.

3.2.1 NSL-KDD train and test data distribution

The 4 attack categories available within the NSL-KDD data set are:

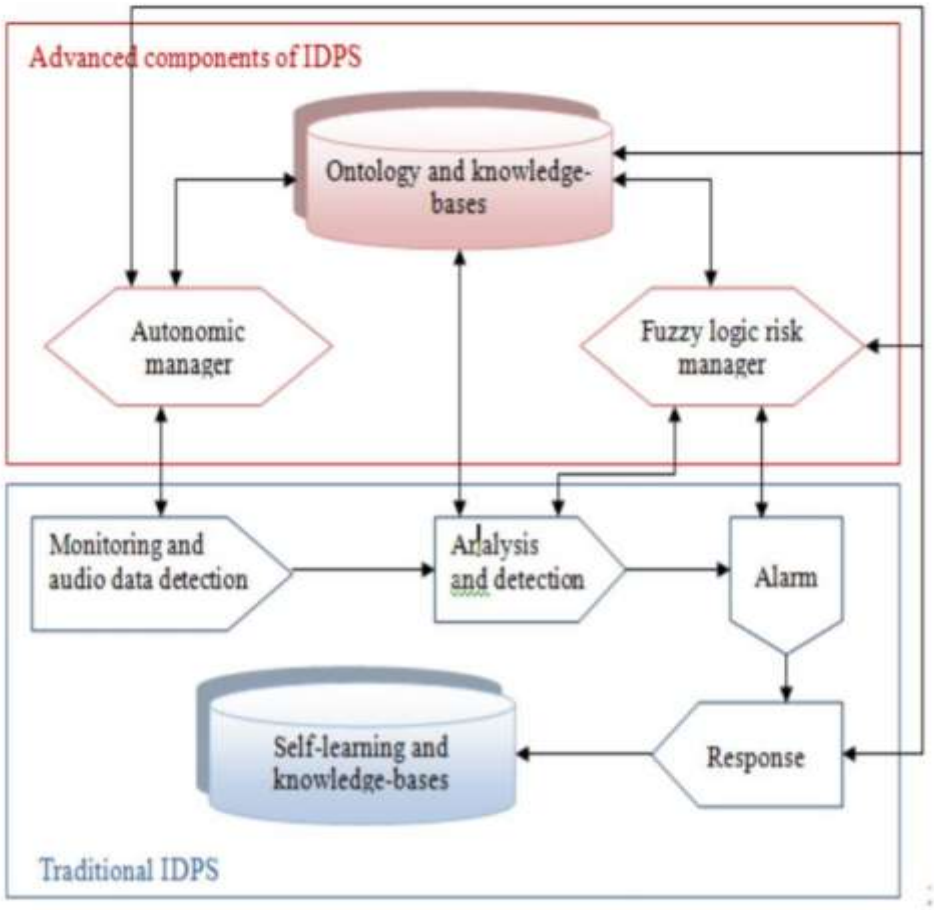
- DOS: This kind of attack leads to draining of the victim's resources and making it incapable in responding to legitimate requests. This is one of the 4 attackcategories.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

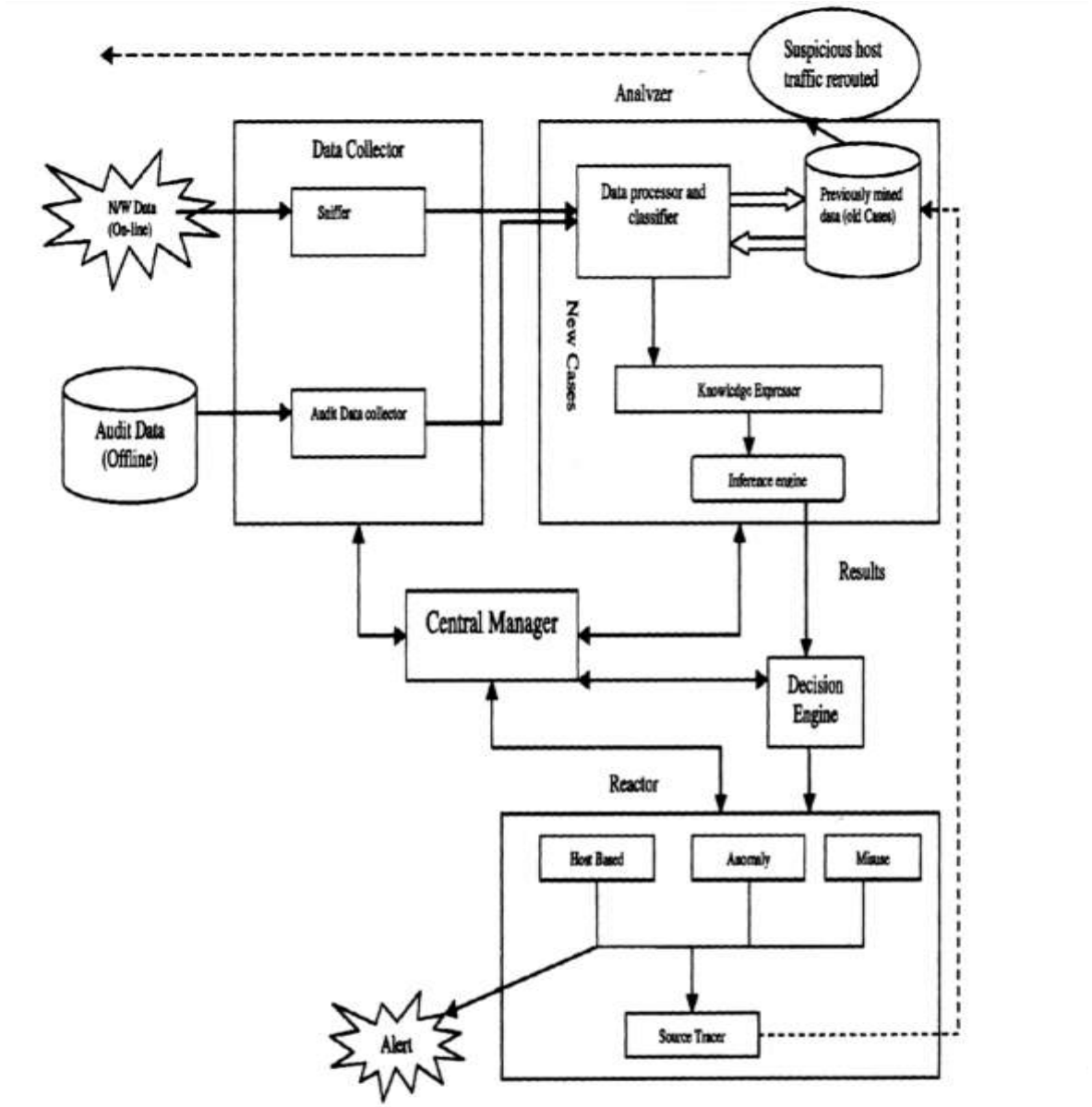
Example: syn flooding. The suitable features from the dataset for this attack class are: –serror_rate| and –flag_SF|.

- U2R(unauthorized access to local root privileges): In this kind of attack, an attacker tries to obtain root/administrator privileges by taking advantage of some vulnerability within the victim's system. The attacker usually uses a traditional account to login into a victim's system. The suitable attributes from the dataset for this attack class are: –root_shell|,–service_http|,and–dst_host_same_src_port_rate|.
- Probing:This kind of attack involves obtaining sensitive information present in the victim's computer/device. The suitable attributes for this attack classare:–Protocol_type_icmp| and–dst_host_same_src_port_rate|.
- R2L: This kind of attack involves unapproved access of the victim's device by gaining root access where he/she can view the data within that device with root privileges and all this is done from a far off(remote) machine by the attacker.

3.2.2Architecture Diagram

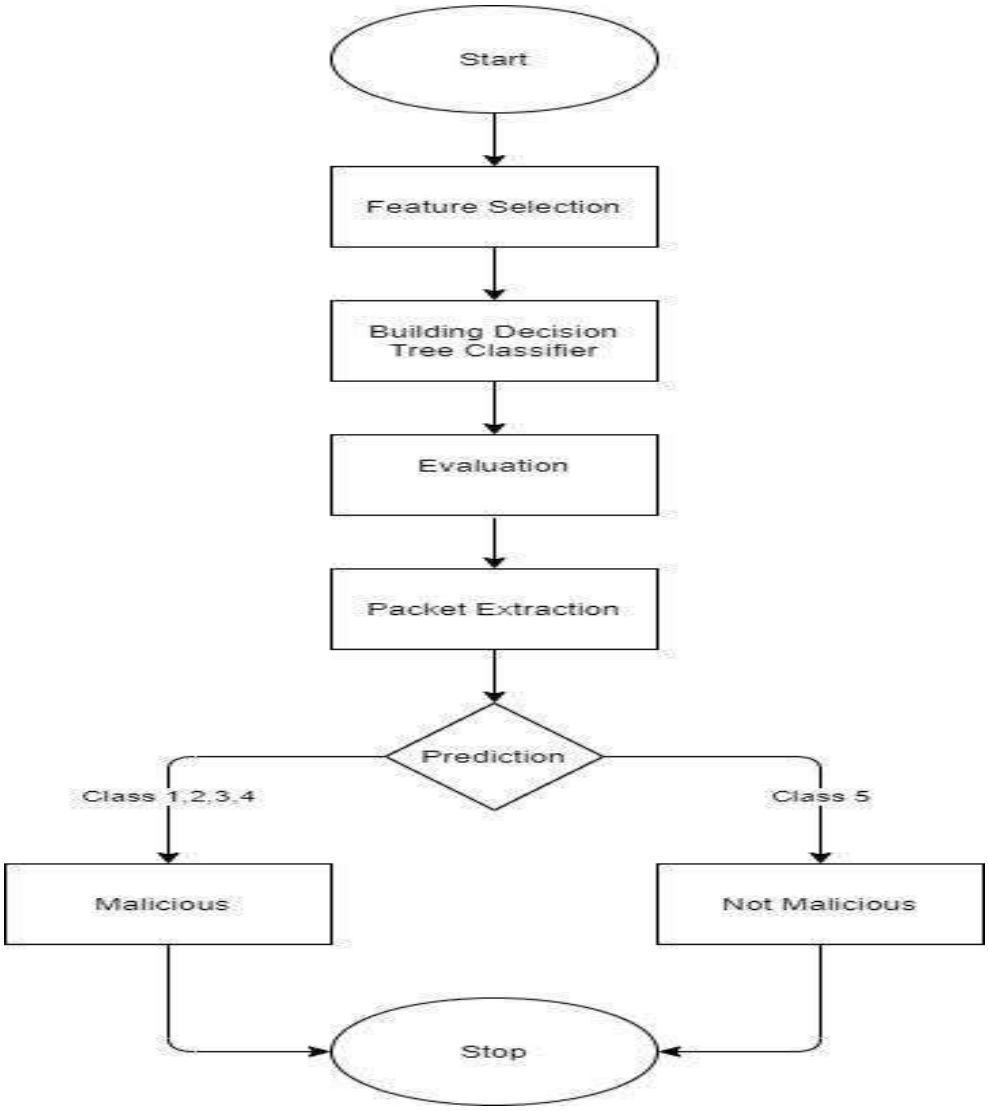


International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252



International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

Fig.3.1 : Project WorkFlow



INTERPRETATION

The data processor and classifier summarizes and tabulates the data into carefully selected categories i.e. the attack types are carefully correlated. This is the stage where a kind of data mining is performed on the collected data. In the next stage, the current data is compared with the historical mined data to create values that reflect how new data differs from the past observed data.

A. Attributes

Prior to any data analysis, attributes representing relevant features of the input data (packets) must be established. The set of attributes provided to the Data Analyzer is a subset of all possible attributes pertaining to the information contained in packet headers, packet payloads, as well as aggregate information such as statistics on the number and type of packets or established TCP connections. Attributes are represented by 4 names that will be used as linguistic variables by the Data Miner and the Fuzzy Inference Engine.

B.Data analyzer

Once attributes of relevance have been defined and a data source identified, a Data Analyzer is employed to compute configuration parameters that regulate operation of the IDS. This module analyzes packets and computes aggregate information by grouping packets. Packets can be placed in fixed size groups (s-group) or in groups of packets captured in a fixed amount of time (t-

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

group). Each s-group contains the same number of packets covering a variable time range and each t-group contains a variable number of packets captured over a fixed period of time.

The architecture as shown in Fig is now under construction. Our preliminary work demonstrates that multi model tree based profiling with data mining can and will provide an efficient solution for Intrusion problems.

3.3 Experimental Implementation / Analytical Computations :

import pandas as pd import numpy as np import sys import sklearn
from IPython.display import Image from sklearn import tree
import pydotplus

3.3.1. Data Preprocessing:

```
col_names = ["duration","protocol_type","service","flag","src_bytes",
    "dst_bytes","land","wrong_fragment","urgent","hot","num_failed_logins",
    "logged_in","num_compromised","root_shell","su_attempted","num_root",
    "num_file_creations","num_shells","num_access_files","num_outbound_cmds",
    "is_host_login","is_guest_login","count","srv_count","serror_rate",
    "srv_serror_rate","rerror_rate","srv_rerror_rate","same_srv_rate",
    "diff_srv_rate","srv_diff_host_rate","dst_host_count","dst_host_srv_count",
    "dst_host_same_srv_rate","dst_host_diff_srv_rate","dst_host_same_src_port_rate",
    "dst_host_srv_diff_host_rate","dst_host_serror_rate","dst_host_srv_serror_rate",
    "dst_host_rerror_rate","dst_host_srv_rerror_rate","label"]
```

df = pd.read_csv("NIDS/KDDTrain+_2.csv", header=None, names = col_names)
df_test = pd.read_csv("NIDS/Network-Intrusion-Detection/KDDTest+_2.csv", header=None, names = col_names)
#this gives the dimensions of the dataset using .shape print('Dimensions of the Training set are:',df.shape) print('Dimensions of the Test set are:',df_test.shape)

```
Dimensions of the Training set: (125973, 42)
Dimensions of the Test set: (22544, 42)
```

3.3.2 Identify categorical features:

print('Training set:') for cname in df.columns:
if (df[cname].dtypes == 'object') : u_categ = len(df[cname].unique())
print("Attribute: '{cname}' has {u_cat} categories".format(cname=cname, u_cat=u_categ))
print('Distribution of categories in service column:')
print(df['service'].value_counts().sort_values(ascending=False).head ())

```
Training set:
Feature 'protocol_type' has 3 categories
Feature 'service' has 70 categories
Feature 'flag' has 11 categories
Feature 'label' has 23 categories

Distribution of categories in service:
http         40338
private      21853
domain_u      9043
smtp          7313
ftp_data      6860
Name: service, dtype: int64
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

3.3.3 Make column names for dummies:

```
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
# code to get a list of categorical columns into a variable, categ_cols
categ_cols=['protocol_type', 'service', 'flag'] df_categ_vals = df[categ_cols] testdf_categ_vals =df_test[categ_cols]
# protocol type col
u_protocol = sorted(df.protocol_type.unique()) s_1 = 'Protocol_type_'
u_protocol_2 = [s_1 + str for str in u_protocol] # service col
u_service = sorted(df.service.unique()) s_2 = 'service_'
u_service_2 = [s_2 + str for str in u_service] # flag col
u_flag = sorted(df.flag.unique()) s_3 = 'flag_'
u_flag_2=[s_3 + atr for str in u_flag] dum_cols=u_protocol_2 + u_service_2 + u_flag_2 print(dum_cols)
# for test set
u_service_test = sorted(df_test.service.unique()) u_service_2_test=[s_2 + str for str in u_service_test] testdum_cols=u_protocol_2 + u_service_2_test + u_flag_2
```

3.3.4 Transform categorical features into numbers using LabelEncoder():

```
df_categ_vals_enc=df_categ_vals.apply(LabelEncoder().fit_transform)
print(df_categ_vals_enc.head())
# test set

testdf_categ_vals_enc=testdf_categ_vals.apply(LabelEncoder().fit_tra nsform)
```

	protocol_type	service	flag
0	1	20	9
1	2	44	9
2	1	49	5
3	1	24	9
4	1	24	9

Table 3.2

3.3.5 One-Hot-Encoding:

```
enc = OneHotEncoder()
df_categ_vals_encenc = enc.fit_transform(df_categ_vals_enc)
df_categ = pd.DataFrame(df_categ_vals_encenc.toarray(),columns=dum_cols)
# for test set
testdf_categ_vals_encenc = enc.fit_transform(testdf_categ_vals_enc)
testdf_categ = pd.DataFrame(testdf_categ_vals_encenc.toarray(),columns=testdum_cols
)
df_categ.head()
```

	Protocol_type_icmp	Protocol_type_tcp	Protocol_type_udp	service_IRC	service_X11
0	0.0	1.0	0.0	0.0	0.0
1	0.0	0.0	1.0	0.0	0.0
2	0.0	1.0	0.0	0.0	0.0
3	0.0	1.0	0.0	0.0	0.0
4	0.0	1.0	0.0	0.0	0.0

5 rows × 84 columns

Table 3.3

3.3.6Join encoded categorical dataframe with the non-categorical dataframe:

```
new_dataset=df.join(df_categ)
new_dataset.drop('flag', axis=1, inplace=True)
new_dataset.drop('protocol_type', axis=1, inplace=True) new_dataset.drop('service', axis=1, inplace=True)
# test data
new_dataset_test=df_test.join(testdf_categ)
new_dataset_test.drop('flag', axis=1, inplace=True)
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

```
new_dataset_test.drop('protocol_type', axis=1, inplace=True) new_dataset_test.drop('service', axis=1, inplace=True) print(new_dataset.shape) print(new_dataset_test.shape)
```

```
(125973, 123)
(22544, 123)
```

3.3.7 Split Dataset into 4 datasets for every attack category:

```
[ ] # take label column
    label_df=new_dataset['label']
    label_df_test=new_dataset_test['label']
    # change the label column
    new_label_df=label_df.replace({ 'normal' : 0, 'neptune' : 1 ,'back': 1, 'land': 1,
                                    'pod': 1, 'smurf': 1, 'teardrop': 1,'mailbomb': 1,
                                    'apache2': 1, 'processtable': 1, 'udpstorm': 1,
                                    'worm': 1, 'ipsweep' : 2,'nmap' : 2,'portsweep' : 2,
                                    'satan' : 2,'mscan' : 2,'saint' : 2,
                                    'ftp_write': 3,'guess_passwd': 3,'imap': 3,
                                    'multihop': 3,'phf': 3,'spy': 3,'warezclient': 3,
                                    'warezmaster': 3,'sendmail': 3,'named': 3,
                                    'snmpgetattack': 3,'snmpguess': 3,'xlock': 3,
                                    'xsnoop': 3,'httptunnel': 3, 'buffer_overflow': 4,
                                    'loadmodule': 4,'perl': 4,'rootkit': 4,'ps': 4,
                                    'sqlattack': 4,'xterm': 4})
```

```
new_label_df_test=label_df_test.replace({ 'normal' : 0, 'neptune' : 1 ,'back': 1,
                                          'land': 1, 'pod': 1, 'smurf': 1,
                                          'teardrop': 1,'mailbomb': 1, 'apache2': 1,
                                          'processtable': 1, 'udpstorm': 1,
                                          'worm': 1, 'ipsweep' : 2,'nmap' : 2,
                                          'portsweep' : 2,'satan' : 2,'mscan' : 2,
                                          'saint' : 2, 'ftp_write': 3,
                                          'guess_passwd': 3,'imap': 3,'multihop': 3,
                                          'phf': 3,'spy': 3,'warezclient': 3,
                                          'warezmaster': 3,'sendmail': 3,'named': 3,
                                          'snmpgetattack': 3,'snmpguess': 3,
                                          'xlock': 3,'xsnoop': 3,'httptunnel': 3,
                                          'buffer_overflow': 4,'loadmodule': 4,
                                          'perl': 4,'rootkit': 4,'ps': 4,
                                          'sqlattack': 4,'xterm': 4})
```

```
# put the new label column back
new_dataset['label'] = new_label_2df
new_dataset_test['label'] = new_label_2df_test
print(new_dataset['label'].head())
elm_DoS = [2,3,4]
elm_Probe = [1,3,4]
elm_R2L =[1,2,4]
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

```
elm_U2R =[1,2,3]

DoS_dataset=new_dataset[~new_dataset['label'].isin(elm_DoS)];
Probe_dataset=new_dataset[~new_dataset['label'].isin(elm_Probe)];
R2L_dataset=new_dataset[~new_dataset['label'].isin(elm_R2L)];
U2R_dataset=new_dataset[~new_dataset['label'].isin(elm_U2R)];
#for test set DoS_dataset_test = new_dataset_test[~new_dataset_test['label'].isin(elm_DoS)];

Probe_dataset_test = new_dataset_test[~new_dataset_test['label'].isin(elm_Probe)];
R2L_dataset_test = new_dataset_test[~new_dataset_test['label'].isin(elm_R2L)];
U2R_dataset_test = new_dataset_test[~new_dataset_test['label'].isin(elm_U2R)];

print('Training set:')
print('Dimensions of DoS:' ,DoS_dataset.shape)
print('Dimensions of Probe:' ,Probe_dataset.shape)
print('Dimensions of R2L:' ,R2L_dataset.shape)
print('Dimensions of U2R:' ,U2R_dataset.shape)
print('Test set:')
print('Dimensions of DoS:' ,DoS_dataset_test.shape) print('Dimensions of Probe:' ,Probe_dataset_test.shape)
print('Dimensions of R2L:' ,R2L_dataset_test.shape) print('Dimensions of U2R:' ,U2R_dataset_test.shape)
Output:
```

```
Train:
Dimensions of DoS: (113270, 123)
Dimensions of Probe: (78999, 123)
Dimensions of R2L: (68338, 123)
Dimensions of U2R: (67395, 123)
Test:
Dimensions of DoS: (17171, 123)
Dimensions of Probe: (12132, 123)
Dimensions of R2L: (12596, 123)
Dimensions of U2R: (9778, 123)
```

3.3.8 Feature scaling:

```
# divide the dataset into features & class X_DoS = DoS_dataset.drop('label',1)
Y_DoS = DoS_dataset.label

X_Probe = Probe_dataset.drop('label',1) Y_Probe = Probe_dataset.label

X_R2L = R2L_dataset.drop('label',1) Y_R2L = R2L_dataset.label

X_U2R = U2R_dataset.drop('label',1)

Y_U2R = U2R_dataset.label # test set
X_DoS_test = DoS_dataset_test.drop('label',1)
Y_DoS_test = DoS_dataset_test.label

X_Probe_test = Probe_dataset_test.drop('label',1)
Y_Probe_test = Probe_dataset_test.label

X_R2L_test = R2L_dataset_test.drop('label',1)
Y_R2L_test = R2L_dataset_test.label

X_U2R_test = U2R_dataset_test.drop('label',1)

Y_U2R_test = U2R_dataset_test.label
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

```
col_Names=list(X_DoS)
col_Names_test=list(X_DoS_test)
from sklearn import preprocessing
sclr_1 = preprocessing.StandardScaler().fit(X_DoS)
X_DoS=sclr_1.transform(X_DoS)
sclr_2 = preprocessing.StandardScaler().fit(X_Probe)
X_Probe=sclr_2.transform(X_Probe)
sclr_3 = preprocessing.StandardScaler().fit(X_R2L)
X_R2L=sclr_3.transform(X_R2L)
sclr_4 = preprocessing.StandardScaler().fit(X_U2R)
X_U2R=sclr_4.transform(X_U2R)
# test data
sclr_5 = preprocessing.StandardScaler().fit(X_DoS_test)
X_DoS_test = sclr_5.transform(X_DoS_test)
sclr_6 = preprocessing.StandardScaler().fit(X_Probe_test)
X_Probe_test = sclr_6.transform(X_Probe_test)
sclr_7 = preprocessing.StandardScaler().fit(X_R2L_test)
X_R2L_test =sclr_7.transform(X_R2L_test)
sclr_8 = preprocessing.StandardScaler().fit(X_U2R_test)
X_U2R_test =sclr_8.transform(X_U2R_test)
```

3.3.9 Feature Selection:

```
from sklearn.feature_selection import RFE
from sklearn.tree import DecisionTreeClassifier
classif = DecisionTreeClassifier(random_state=0)
feature_sel = RFE(estimator=classif, n_features_to_select=13, step=1)

Y_DoS = Y_DoS.astype('int')
feature_sel.fit(X_DoS, Y_DoS)
X_DoS_rfe = feature_sel.transform(X_DoS) true = feature_sel.support_
sel_colindexDoS=[i for i, j in enumerate(true) if j]

Y_Probe = Y_Probe.astype('int')
feature_sel.fit(X_Probe, Y_Probe) X_Probe_rfe = feature_sel.transform(X_Probe)
true=feature_sel.support_
sel_colindexProbe=[i for i, j in enumerate(true) if j]

Y_R2L = Y_R2L.astype('int')
feature_sel.fit(X_R2L, Y_R2L)
X_R2L_rfe = feature_sel.transform(X_R2L) true = feature_sel.support_
sel_colindexR2L=[i for i, j in enumerate(true) if j]

Y_U2R = Y_U2R.astype('int')
feature_sel.fit(X_U2R, Y_U2R)
X_U2R_rfe = feature_sel.transform(X_U2R) true = feature_sel.support_
sel_colindexU2R=[i for i, j in enumerate(true) if j]
```

3.3.10Build the model:

```
classif_DoS_rfe = DecisionTreeClassifier(random_state=0)
classif_Probe_rfe = DecisionTreeClassifier(random_state=0)
classif_R2L_rfe = DecisionTreeClassifier(random_state=0)
classif_U2R_rfe = DecisionTreeClassifier(random_state=0)

classif_DoS_rfe.fit(X_DoS_rfe, Y_DoS)
classif_Probe_rfe.fit(X_Probe_rfe, Y_Probe)
classif_R2L_rfe.fit(X_R2L_rfe, Y_R2L)
classif_U2R_rfe.fit(X_U2R_rfe, Y_U2R)

X_DoS_rfe_test = X_DoS_test[:,sel_colindexDoS]
X_Probe_rfe_test = X_Probe_test[:,sel_colindexProbe]
X_R2L_rfe_test = X_R2L_test[:,sel_colindexR2L]
X_U2R_rfe_test = X_U2R_test[:,sel_colindexU2R]
```

3.4 Tools Used :

3.4.1 scikit-learn

scikit-learn is a Machine learning library developed with the help of SciPy. It is widely used by Machine learning practitioners.

David Cournapeau started the project in 2007 for GSoC competition and since then many developers have been working on the code which is open sourced.

Important features of scikit-learn:

1. It is a plain and productive tool used for data processing software and data mining. Some of the algorithms it includes are: regression, classification, clustering algorithms like k-means, random forests, gradient boosting, support vector machines,etc.
2. Open to all and reusable in differentsituations.
3. It was built with the foundation of matplotlib, NumPy andSciPy.
4. Commercially available open source-BSDlicense.

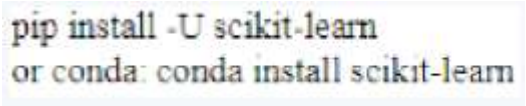
Installation Dependencies:

1.Python (>=v3.5)
2. NumPy (>= v1.11.0)
3. SciPy (>= v0.17.0)
4. joblib (>= v0.11)

The last version to support Python 2.7 and Python

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

3.4 was Scikit-learn 0.20. ThelaterversionsofScikit-learni.e.Scikit-learn0.21andlaterrequirePython 3.5 ornewer.
Scikit-learn also provides plotting facilities (i.e., functions start with "plot_" and classes end with "Display") which need Matplotlib (>= v1.5.1) for running. Some examples may also need scikit-image >= v0.12.3, and some may need pandas >= v0.18.0.



User installation
If your system already has the required versions numpy and scipy which is completely working, the simplest method to install scikit-learn will be by using

3.4.2 PyCharm

PyCharm is an integrated development environment (IDE) employed in programming and developing daily use softwares, developed for the Python language explicitly. PyCharm is developed by the very popular Czech company- JetBrains. It provides an interface for code analysis, a graphical debugger for debugging the code, integration with version control systems (VCSes), and supports web development using Django. And it also supports Data Science using Anaconda.

PyCharm is a cross-platform software, which makes it compatible with versions of Windows, Linux and MacOS. The software's Community Version comes out under the Apache License. It also provides extra features to Professional Edition-published under a proprietary license that can be used for technical purposes.

Installing Python

- Python can be downloaded from its official website http://www.python.org/downloads/, where you choose a version as per your needs. We choose Python v3.6.3 for ourconvenience.
- Python is installed by running the exe file after the completion of the download.
- Now, you can see a Window where Python is beinginstalled.
- It will display a message saying : The Setup was Successful. Now press the closebutton.

Installing Pycharm

- PyCharm can be downloaded from its official website https://www.jetbrains.com/pycharm/download/ .Visit the website and click DOWNLOAD"under the CommunitySection.
- PyCharm can be installed by running the exe file, once the download is complete.You can click the "Next" button on the SetupWizard.
- The Path of installation can be changed on the following screen. Click the "Next"button.
- You can build a desktop shortcut on the next screen if you wish, and press "Next".
- Choose the folder in the Start tab. Make sure JetBrains is selected and press"Install".
- Wait until the setup iscomplete.
- When the installation is complete, you will be able to see a screen that shows the message that says PyCharm is installed. Click on "Run PyCharm Community Edition" before clicking on the "Finish"button.

3.4.3 Numpy:

NumPy is Python's most commonly used scientific computing program. Among other items, it includes the following:

- a powerful N-dimensional arrayobject
- sophisticated (broadcasting)functions
- tools for integrating C/C++ and Fortrancode
- Fourier transform calculator, functions for linear algebra, and random numberfacilities

We may use NumPy as a powerful multi-dimensional container of general purpose data in addition to its obvious scientific uses. We can also describe arbitrary data types. This allows NumPy to integrate with a wide variety of databases seamlessly and effortlessly.
NumPy is licensed under the BSD license.This enables reuse of the package with few restrictions.[11]

3.4.4 Pandas:

pandasis a Python programming language software library written for use in. The package's principal aim is to control and analyse data. It provides only data structures and other operations to manipulate numerical tables and statistics.

This program is fully free and was published under the BSD license with three clauses. The library name derives from the term "panel data," a term for data sets which uses observations for identical individuals from multiple time periods.
Features:

1. DataFrame object with Integrated Indexing for manipulatingdata.
2. It provides resources between in-memory data

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

structures and various fileformats for both reading and writingdata.
3. Data synchronization, and streamlined datamanagement.
4. Pivoting and reshaping of huge datasets.
5. Label slicing, clever indexing and subsetting of large datasets.
6. Insertion and deletion of column DataStructure.
7. This also comes with a community by engine that enables operations on data sets with split-apply-combine.
8. Data set merging andjoining

CHAPTER- IV

4.1 Results :

```
from sklearn.model_selection import cross_val_score
```

4.1.1 DoS:

```
acc= cross_val_score(classif_DoS_rfe, X_DoS_rfe_test, Y_DoS_test, cv=10,scoring='accuracy')
print("Accuracy: %0.5f (+/- %0.5f)" % (acc.mean(), acc.std() * 2))

prec = cross_val_score(classif_DoS_rfe, X_DoS_rfe_test, Y_DoS_test, cv=10, scoring='precision')
print("Precision: %0.5f (+/- %0.5f)" % (prec.mean(), prec.std() * 2))

recall = cross_val_score(classif_DoS_rfe, X_DoS_rfe_test, Y_DoS_test, cv=10, scoring='recall')
print("Recall: %0.5f (+/- %0.5f)" % (recall.mean(), recall.std() * 2))

f1 = cross_val_score(classif_DoS_rfe, X_DoS_rfe_test, Y_DoS_test, cv=10, scoring='f1')
print("F-measure: %0.5f (+/- %0.5f)" % (f1.mean(), f1.std() * 2))
```

RESULT:

```
Accuracy: 0.99738 (+/- 0.00267)
Precision: 0.99692 (+/- 0.00492)
Recall: 0.99705 (+/- 0.00356)
F-measure: 0.99698 (+/- 0.00307)
```

4.1.2 Probe:

```
acc = cross_val_score(classif_Probe_rfe, X_Probe_rfe_test, Y_Probe_test, cv=10,scoring='accuracy')
print("Accuracy: %0.5f (+/- %0.5f)" % (acc.mean(), acc.std() * 2))
prec = cross_val_score(classif_Probe_rfe, X_Probe_rfe_test, Y_Probe_test, cv=10,scoring='precision_macro')
print("Precision: %0.5f (+/- %0.5f)" % (prec.mean(), prec.std() * 2))
recall = cross_val_score(classif_Probe_rfe, X_Probe_rfe_test, Y_Probe_test, cv=10,scoring='recall_macro')
print("Recall: %0.5f (+/- %0.5f)" % (recall.mean(), recall.std() * 2))

f1 = cross_val_score(classif_Probe_rfe, X_Probe_rfe_test, Y_Probe_test, cv=10, scoring='f1_macro')
print("F-measure: %0.5f (+/- %0.5f)" % (f1.mean(), f1.std() * 2))
```

RESULT:

```
Accuracy: 0.99085 (+/- 0.00559)
Precision: 0.98674 (+/- 0.01179)
Recall: 0.98467 (+/- 0.01026)
F-measure: 0.98566 (+/- 0.00871)
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

4.1.3 R2L:

```
acc = cross_val_score(classif_R2L_rfe, X_R2L_rfe_test, Y_R2L_test, cv=10,scoring='accuracy')
print("Accuracy: %0.5f (+/- %0.5f)" % (acc.mean(), acc.std() * 2))
prec = cross_val_score(classif_R2L_rfe, X_R2L_rfe_test, Y_R2L_test, cv=10,scoring='precision_macro') print("Precision: %0.5f (+/- %0.5f)" % (prec.mean(), prec.std() * 2))
recall = cross_val_score(classif_R2L_rfe, X_R2L_rfe_test, Y_R2L_test, cv=10,scoring='recall_macro')
print("Recall: %0.5f (+/- %0.5f)" % (recall.mean(), recall.std() * 2))
f1 = cross_val_score(classif_R2L_rfe, X_R2L_rfe_test, Y_R2L_test, cv=10,scoring='f1_macro')

print("F-measure: %0.5f (+/- %0.5f)" % (f1.mean(), f1.std() * 2))
```

RESULT:

```
Accuracy: 0.97459 (+/- 0.00910)
Precision: 0.96689 (+/- 0.01311)
Recall: 0.96086 (+/- 0.01571)
F-measure: 0.96379 (+/- 0.01305)
```

4.1.4 U2R:

```
acc = cross_val_score(classif_U2R_rfe, X_U2R_rfe_test, Y_U2R_test, cv=10,scoring='accuracy')
print("Accuracy: %0.5f (+/- %0.5f)" % (acc.mean(), acc.std() * 2))
prec = cross_val_score(classif_U2R_rfe, X_U2R_rfe_test, Y_U2R_test, cv=10,scoring='precision_macro') print("Precision: %0.5f (+/- %0.5f)" % (prec.mean(), prec.std() * 2))
recall = cross_val_score(classif_U2R_rfe, X_U2R_rfe_test, Y_U2R_test, cv=10,scoring='recall_macro')
print("Recall: %0.5f (+/- %0.5f)" % (recall.mean(), recall.std() * 2)) f1 = cross_val_score(classif_U2R_rfe, X_U2R_rfe_test, Y_U2R_test, cv=10,scoring='f1_macro')
print("F-measure: %0.5f (+/- %0.5f)" % (f1.mean(), f1.std() * 2))
```

RESULT:

```
Accuracy: 0.99652 (+/- 0.00278)
Precision: 0.87538 (+/- 0.15433)
Recall: 0.89540 (+/- 0.14777)
F-measure: 0.87731 (+/- 0.09647)
```

4.2 Interpretation of Results :

Confusion Matrices

4.2.1 DoS:

```
DoS_Y_predicted = classif_DoS_rfe.predict(X_DoS_rfe_test)
pd.crosstab(Y_DoS_test, DoS_Y_predicted, rownames=['Actual attacks'], colnames=['Predicted attacks'])
```

Predicted attacks	0	1
Actual attacks		
0	9602	109
1	2625	4835

Table 4.1

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

4.2.2 PROBE:

```
Probe_Y_predicted=classif_Probe_rfe.predict(X_Probe_rfe_test)
pd.crosstab(Y_Probe_test, Probe_Y_predicted, rownames=['Actual attacks'],colnames=['Predicted attacks'])
```

Predicted attacks	0	2
Actual attacks		
0	8709	1002
2	944	1477

Table 4.2

4.2.3 R2L:

```
R2L_Y_predicted=classif_R2L_rfe.predict(X_R2L_rfe_test)
pd.crosstab(Y_R2L_test, R2L_Y_predicted, rownames=['Actual attacks'], colnames=['Predicted attacks'])
```

Predicted attacks	0	3
Actual attacks		
0	9649	62
3	2560	325

Table 4.3

4.2.4 U2R:

```
U2R_Y_predicted=classif_U2R_rfe.predict(X_U2R_rfe_test)
pd.crosstab(Y_U2R_test, U2R_Y_predicted, rownames=['Actual attacks'], colnames=['Predicted attacks'])
```

Predicted attacks	0	4
Actual attacks		
0	9706	5
4	52	15

Table 4.4

4.2.5 Cross Validation: Accuracy, Precision, Recall, F-measure:

Accuracy:

The accuracy (AC) is defined as the distribution of the total number of correct predictions. The equation is estimated as:

Accuracy= TP+TN/TP+TN+FP+FN

Precision:

Precision is defined as the ratio of the total no. of true positives and the sum of the number of true positives and the number of false positives. It is calculated by theequation:

Precision 1= TP/TP+FP

Recall:

Recall can be defined as the proportion of the total number of positive examples rightly listed, divided by the total number of positive ones. High Recall suggests correct identification of class. It is also defined in scientific terms as Detection Frequency, True Positive Rate, or Sensitivity. The equation computes as:

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 2 Feb 2021, pp: 521-540 www.ijaem.net ISSN: 2395-5252

Recall = TP/TP+FN

F-score:

The F score is also known as F1 score or F measure. It is a measure of the accuracy of a test. The F score is interpreted as the weighted harmonic mean of the test's precision and recall. This score is calculated using the following equation:

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precisior}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precisior} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

4.3 Inferences from Results and Analysis

As seen above, we have managed to detect four major types of intrusion attacks namely: Dos, Probe, R2L, U2R with more than 98 % accuracy with the help of our decision model tree classifier. Along with accuracy, we have calculated three other parameters such as precision, recall and f-measure as well which help us evaluate the performance of our model and generate accurate analysis as far as detection of each of the four intrusion attacks are concerned.

CHAPTER- V

5.1 Summary :

Intrusion detection is one of the key interests in network administration and security. There is a need to protect the networks from known vulnerabilities and at the same time take steps to identify new and unknown but potential device abuses by creating more robust and effective systems for intrusion detection. A Decision tree is an algorithm which takes decisions at each node of the tree and is widely used for regression and classification. It is a supervised learning algorithm in Machine learning where which attribute should be at which node is learnt by using a set of labelled examples.

Another type of Intrusion detection system is used to detect anomalies. But, the current software which detects anomalies detects a high number of false positives which leads to an increase in the rate of false alarms. Also, the results obtained were highly inaccurate and in many cases, some types of attacks were not able to be detected. Therefore we conducted an experiment to assess the accuracies and detection rates of various algorithms using the NSL-KDD dataset. We could detect four major types of attacks namely Dos, Probe, U2R and R2L with an accuracy of approximately 98%. According to our results, we were able to conclude that our approach i.e. Single Level Multi-model using Decision Trees, has performed exceptionally well and has shown very low rates of false alarms.We even used the concept of data visualisation to plot effective histogram graph and line graph todenote the frequency of attacks.

5.2 Conclusions :

Nowadays, almost every system is prone to attacks. There is a need to increase the security in our daily use systems. This can be achieved by using Intrusion Detection Systems. It helps us in detecting malicious activities efficiently. Another type of Intrusion detection system is used to detect anomalies. But, the current software which detects anomalies detects high number false positives which leads to an increase in the rate of false alarms. Also the results obtained were highly inaccurate and in many cases some types of attacks were not able to be detected. Therefore we conducted an experiment to assess the accuracies and detection rates of various algorithms using the NSL-KDD dataset. According to our results we were able to conclude that our approach i.e. Single Level Multi-model using Decision Trees, has performed exceptionally well and has shown very low rates of false alarms. We have taken into consideration the factors like Accuracy, Precision, Recall and F-Measure to select our approach.

5.3 Scope For Further Study :

In the current project, we have only implemented intrusion detection for four types of attacks.So, in the future, we would like to integrate Intrusion Prevention system to it. We plan to use different ML Models such as Convolution Neural Network Model and Support Vector Network Model to incorporate an efficient backbone of Intrusion Prevention architecture. Policies will alsobe highlighted to clearly define the roles and access benefits of the different actors inan organization. Alarm Systems can also be integrated to alert the officers-in-charge of a possible intrusionfrom an attacker. Software like a Firewall can help in the prevention of an attack and prevent the intruder from accessing the systems.

REFERENCES

[1]. Peyman Kabiri, Ali A. Ghorbani (2005, Sept). Research on Intrusion Detection and Response: A Survey. International Journal of

Network Security, Vol.1, No.2, PP.84–102
[2]. Nancy Agarwal, Syed Zeeshan Hussain. (Feb,2010). A Closer Look at Intrusion Detection System for Web Applications, Hindawi, Security and Communication Networks Volume 2018, Article ID 9601357
[3]. Adeeb Alhomouda *, Rashid Munira ,Jules Pagna Dissoa ,Irfan Awana,b,A. Al-Dhelaanb. (2011). Performance Evaluation Study of Intrusion Detection Systems. The 2nd International Conference on Ambient Systems, Networks and Technologies (ANT)
[4]. Mostaque Md. Morshedur Hassan (2013, March). Current Studies on Intrusion Detection System, Genetic and fuzzy logic. International Journal of Distributed and Parallel Systems (IJDPS), Vol.4, No.2
[5]. Muhammad K. Asif, Talha A. Khan, Talha A. Taj, Umar Naeem. Sufyan Yakoob. (May, 2013). Network Intrusion Detection and its Strategic Importance. IEEE Business Engineering and Industrial Applications Colloquium (BEIAC)
[6]. Mahdi Zamani. (Dec,2013) Machine Learning Techniques for Intrusion Detection.
[7]. Ansam Khraisat , Iqbal Gondal, Peter Vamplew and Joarder Kamruzzaman (2019, July). Survey of intrusion detection systems: techniques, datasets and challenges. Khraisat et al. Cybersecurity
[8]. Anish Naik, (December 12, 2019). Issues and Recent Advances in Machine Learning Techniques for Intrusion Detection Systems.
[9]. Roberto Magán-Carrión, Daniel Urda, Ignacio Díaz-Cano and Bernabé Dorronsoro. (2020, Mar). Towards a Reliable Comparison and Evaluation of Network Intrusion Detection Systems Based on Machine Learning Approaches
[10]. Ankit Thakkar, Ritika Lohiya. (June, 2020). A Review of the Advancement in Intrusion Detection Datasets. International Conference on Computational Intelligence and Data Science (ICCIDS 2019)
[11]. Precision and recall: Precision and Recall Definition Numpy: https://numpy.org/
[12]. Code in Collab https://colab.research.google.com/drive/1OG5Ku-B9y_T1haixTyh N82BToygip UFp#scrollTo=FuZj__qvRyJn
[13]. probleNSL-KDD Dataset: NSL-KDD https://www.kaggle.com/hassan06/nslkdd

International Journal of Advances in Engineering and Management
ISSN: 2395-5252

IJAEM

Volume: 03 Issue: 02 DOI: 10.35629/5252

www.ijaem.net
Email id: ijaem.paper@gmail.com