VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

INFORMATION SECURITY ANALYSIS AND AUDIT

J-COMPONENT FINAL REVIEW

FALL SEMESTER 2020-21

Group Members:

KHWAAB THAREJA (18BCE0930)
STUTI TIWARI (18BCE0950)
RAHUL ANAND (18BCE0953)

Submitted To:

Prof. LAVANYA K.

NETWORK INTRUSION DETECTION SYSTEM

ABSTRACT

Network has brought convenience to the world by allowing flexible transformation of data, but it also exposes a high number of vulnerabilities. A Network Intrusion Detection System (NIDS) helps system and network administrators to detect network security breaches in their organizations. Identifying anonymous and new attacks is one of the main challenges in IDSs researches. Machine learning methodologies have been widely used in identifying various types of attacks, and a machine learning approach can help the network administrator take the corresponding measures for preventing intrusions. However, most of the traditional machine learning methodologies belong to shallow learning and often emphasize feature engineering and selection. They cannot effectively solve the massive intrusion data classification problem that arises in the face of a real network application environment.

Machine Learning is a vast field and comprises of number of algorithms which can perform supervised as well as unsupervised tasks. For Network Intrusion Detection, we have used a few of the Supervised Algorithms to detect anomaly in the network activity. These algorithms will predict the occurrence of a potential attack with a certain probability and will thus help the network administrator prepare well.

INTRODUCTION

An IDS is a device or an application that monitors, [1] detects and identifies network operations for unauthorized and malicious activities that end up in policy violation. They generally focus on detecting and identifying possible circumstances and then they log all the possible information and report attempts. [2] In principle, an IDS has the capability of real-time detection of all possible intrusions and put into work the counter measures to stop the attack.

Intrusion detection system can be classified into three categories:

1. **Host Based IDS**: evaluate data obtained on a single or multiple host systems which includes contents of OS, system and applications.[4] HIDS tends to be more accurate and less false positive than network-based IDS because it analyses the log files, and as a result, it can determine whether an attack successful occurred or not [6].
2. **Network Based IDS**: evaluates information obtained from the communication between the networks, the obtained packets are [5] analysed. Sensors are used to capture the network traffic packets.
   The problem with NIDS is that it has restricted visibility inside the host machine, and there is no effective way to analyse encrypted network traffic to detect attack [7]. Therefore, until now, many researches progressed to develop effective ways for NIDS to detect attacks.

3. **Vulnerability Assessment IDS**: the vulnerability on the internal networks and the firewall are detected. Two primary models are available: [8]

   a. Misuse Detection Model

   b. Anomaly Detection Model

Most IDS commercial tools refer to the misuse detection model, and signatures of intrusions must always be updated by vendors. [3] IDS based on anomaly detection model have the ability to detect symptoms of attack without specifying model of attacks but they are very sensitive to false alarms.

Because of the great precision rate, bunches of IDS frameworks like the Neural system (NN) based preparing for building their knowledgebase contrasting and the conventional learning calculations. Be that as it may, in the most recent decade, there is an expanding pattern of the utilization of profound learning

approaches because of its capacity to utilize the large information for the preparation and its all-out precision rate.

Comparison of IDS with Firewalls:

IDS and firewall both are identified with the system security yet an IDS varies from a firewall as a firewall searches externally for interruptions so as to prevent them from occurring. Firewalls limit access between systems to forestall interruption and if an assault is from inside the system it doesn't flag. An IDS portrays a presumed interruption once it has occurred and afterward flags a caution.

Literature Review

How Does Network Intrusion Work?

Attackers can take several different approaches when attempting to penetrate a system. With network intruder detection software, understanding what kinds of attacks can be used is critically important for setting up effective prevention.

In many cases of network intrusion, the attack involves flooding or overloading the network, gathering data about the network to attack it from a weak point later, or inserting information into the network to spread and gain access from inside. It's important to keep hacker detection tools active, so you can prevent these vulnerabilities from getting into your system in the first place.

Thanks to the paper of Yogita Bhavsar,

Intrusion Detection System Using Data Mining Technique, we were able to classify the attack types as: -

- Denial of service attack (Dos)
- Probing Attack (Probe)

- User to Root Attack (U2R)
- Remote to Local Attack (R2L)

DENIAL OF SERVICE (DoS) ATTACKS

A **Denial-of-Service (DoS) attack** is an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash. In both instances, the DoS attack deprives legitimate users (i.e. employees, members, or account holders) of the service or resource they expected.

There are two general methods of DoS attacks: flooding services or crashing services. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop. Popular flood attacks include:

- **Buffer overflow attacks**
- **ICMP flood**
- **SYN flood**

PROBING ATTACKS

Probe-response attacks are a new threat for collaborative intrusion detection systems.

A probe is an attack which is deliberately crafted so that its target detects and reports it with a recognizable "fingerprint" in the report. The attacker then uses the collaborative infrastructure to learn the detector's location and defensive capabilities from this report.

USER TO ROOT ATTACKS

User to root attack (u2r) is usually launched for illegally obtaining the root's privileges when legally accessing a local machine.

REMOTE TO LOCAL ATTACKS

Remote to local attack (r2l) has been widely known to be launched by an attacker to gain unauthorized access to a victim machine in the entire network.

One approach for detecting both attacks is to formulate both problems as a binary classification problem by deciding whether to accept or reject access requests from remote sites to local user machine or by accepting or rejecting access as root attempts.

Evasion Techniques

Being aware of the techniques available to cyber criminals who are trying to breach a secure network can help IT departments understand how IDS systems can be tricked into not missing actionable threats:

- **Fragmentation:** Sending fragmented parcels permit the assailant to remain under the radar, bypassing the identification framework's capacity to identify the assault signature.
- **Avoiding defaults:** A port utilized by a protocol does not always provide an indication to the protocol that's being transported. If an attacker had reconfigured it to use a different port, the IDS may not be able to detect the presence of a trojan.
- **Coordinated, low-bandwidth attacks:** coordinating a scan among numerous attackers, or even allocating various ports or hosts to different attackers. This makes it difficult for the IDS to correlate the captured packets and deduce that a network scan is in progress.
- **Address spoofing/proxying:** attackers can obscure the source of the attack by using poorly secured or incorrectly configured proxy servers to bounce an attack. If the source is spoofed and bounced by a server, it makes it very difficult to detect.
- **Pattern change evasion:** IDS rely on pattern matching to detect attacks. By making slight adjust to the attack architecture, detection can be avoided.

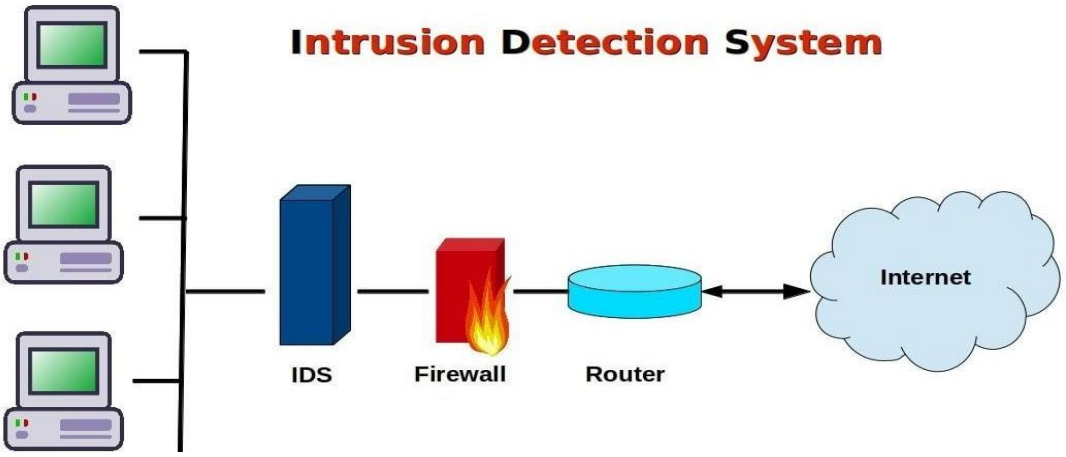


Fig. 1 Overview of IDS

Detection Method of IDS:

1. **Signature-based Method:**

   Signature-based IDS detects the attacks on the basis of the specific patterns such as number of bytes or number of 1's or number of 0's in the network traffic. It also detects on the basis of the already known malicious instruction sequence that is used by the malware. The detected patterns in the IDS are known as signatures.[9]
   Signature-based IDS can easily detect the attacks whose pattern (signature) already exists in system but it is quite difficult to detect the new malware attacks as their pattern (signature) is not known.

2. **Anomaly-based Method:**

   Anomaly-based IDS was introduced to detect the unknown malware attacks as new malware are developed rapidly. In anomaly-based IDS there is use of machine learning to create a trustful activity [10] model and anything coming is compared with that model and it is declared suspicious if it is not found in model. Machine learning based method has a better generalized property in comparison to signature-based IDS as these models can be trained according to the applications and hardware configurations.

PROS AND CONS OF INTRUSION DETECTION

Pros of IDS are as follows:

- Detects external hackers and network-based attacks.
- Offers centralized management for correlation of distributed attacks.
- Provides the system administrator the ability to quantify attacks.
- Provides an additional layer of protection.
- Provides defence in depth.

Cons of IDS are as follows:

- Generates false positives and negatives.
- Require full time monitoring.
- It is expensive
- Require highly skilled staff.

Structure and architecture of intrusion detection systems

An intrusion detection system always has its core element - a sensor (an analysis engine) that is responsible for detecting intrusions. This sensor contains decision-making mechanisms regarding intrusions. Sensors receive raw data from three major information sources own IDS knowledge base, syslog and audit trails. The syslog may include, for example, configuration of file system, user authorizations etc. This information creates the basis for a further decision-making process.

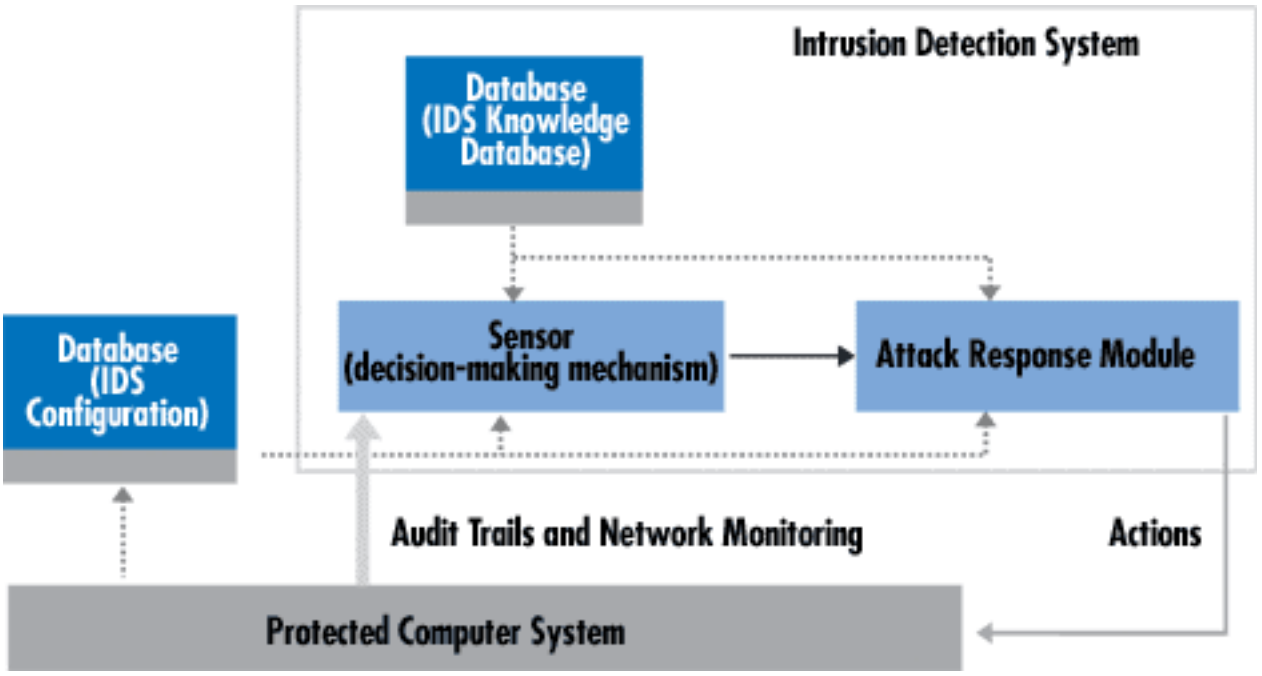


Fig. 2 System Architecture

The sensor is integrated with the component responsible for data collection - an event generator. The collection manner is determined by the event generator policy that defines the filtering mode of event notification information.

The event generator (operating system, network, application) produces a policy-consistent set of events that may be a log (or audit) of system events, or network packets.

This, set along with the policy information can be stored either in the protected system or outside. In certain cases, no data storage is employed for example, when event data streams are transferred directly to the analyser.
This concerns the network packets in particular.

The role of the sensor is to filter information and discard any irrelevant data obtained from the event set associated with the protected system, thereby detecting suspicious activities. The analyser uses the detection policy database for this purpose.
The latter comprises the following elements: attack signatures, normal behaviour profiles, necessary parameters (for example, thresholds).

Intrusion detection systems can be arranged as either centralized (for example, physically integrated within a firewall) or distributed.

A distributed IDS consists of multiple Intrusion Detection Systems (IDS) over a large network, all of which communicate with each other. More sophisticated systems follow an agent structure principle where small autonomous modules are organized on a per-host basis across the protected network [14].

The role of the agent is to monitor and filter all activities within the protected area and - depending on the approach adopted - make an initial analysis and even undertake a response action.

Another separate role of the agent is associated with its mobility and roaming across multiple physical locations [15].

Apart from agents, the system may have transceivers to monitor all operations effected by agents of a specific host.

Transceivers always send the results of their operations to a unique single monitor. Monitors receive information from a specific network area (not only from a single host), which means that they can correlate distributed information. Additionally, some filters may be introduced for data selection and aggregation [17], [16].

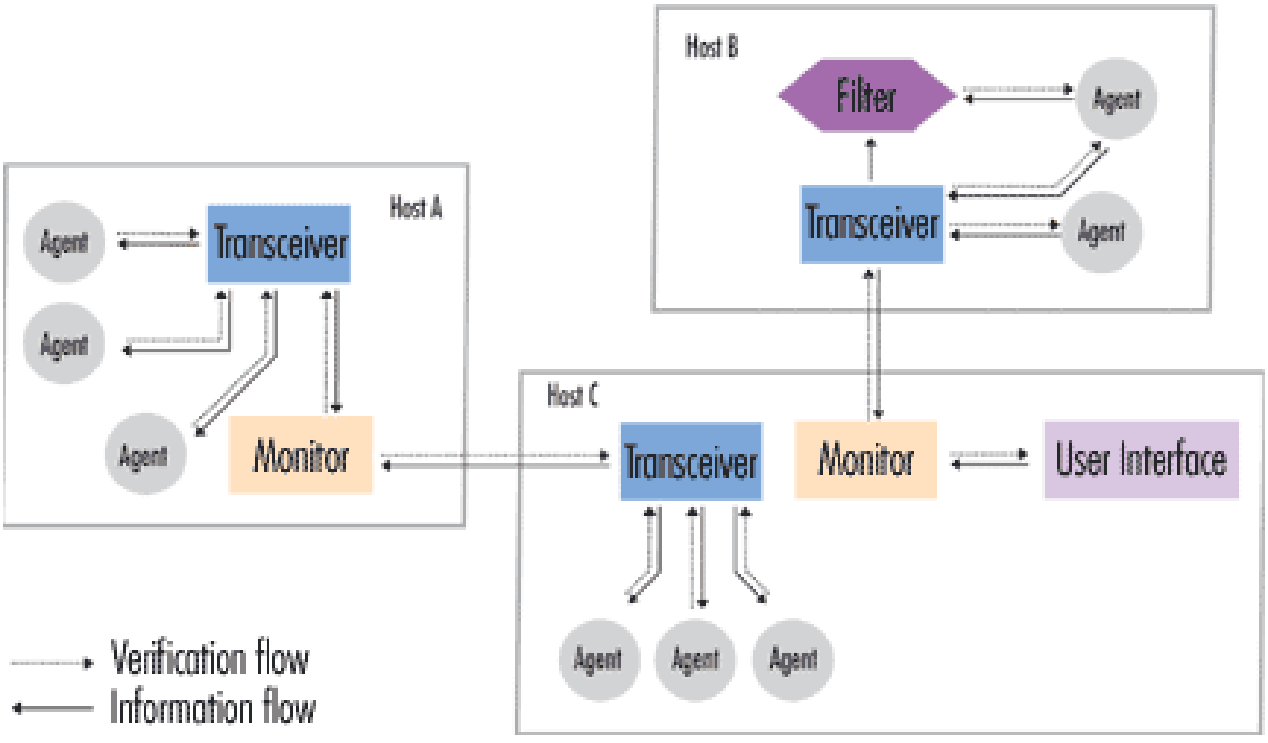


Fig. 3 Flow of Network

INNOVATIIVE IDEA

Machine Learning is implemented for the Intrusion Detection system. The innovative idea for the project was to improve the training accuracy by reducing the overfitting and classifying it better. Along with that this project does not only tell that there is an intrusion but also tell the intrusion through different attacks and classify them into different classes.

COMPARISON RESULTS

The attack classes are divided into 4 categories and a single normal category depicting no sign of attack. These 4 categories i.e. –

1. DOS
2. Probe
3. R2L
4. U2R

These classes have different frequency of attacks and as seen from the figure, Dos attack is the most common among other types of attacks.

	Model	Accuracy
0	Naive Bayes	0.760171
1	Decision Tree	0.999629
2	K Neighbors	0.993677
3	Logistic Regresssion	0.929804

Fig. 4 Accuracies of classifiers

Type of Scores/Models	Precision	Recall score	F1-score
Naive Bayes	0.76	0.76	0.76
Decision Tree	1.0	1.0	1.0
KNN	1.0	1.0	1.0
Logistic Regression	0.92	0.92	0.92

Fig. 5 Evaluation metrices of classifiers

The above table shows that the Decision Tree Classifier and KNN have the highest scores as compared to the other models.

SAMPLE CODE

Import the libraries

In [4]:
```
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import sklearn
import warnings
warnings.filterwarnings('ignore')
import imblearn
%matplotlib inline
```

Load the Dataset

KDD stands for Knowledge Discovery and Data Miniming.We have used the NSL KDD Dataset which is cleaned and upgraded version of the KDD'99 dataset which was mainly used for Intrusion detection earlier. The KDD'99 Dataset - http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

The NSL-KDD Dataset - https://www.unb.ca/cic/datasets/nsl.html

The names and description of the columns were obtained from - https://docs.google.com/spreadsheets/d/1uDYzhEuKlXiM-f2f4f_gthgS0uqh2NN-lzR9dGxP2ZI/edit#gid=0

In [5]:
```
# 42 columns present in the datset
columns = ["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","attack", "last_flag"]

# Load NSL_KDD train dataset
dfkdd_train = pd.read_table("KDDTrain.txt", sep=",", names=columns)
dfkdd_train = dfkdd_train.iloc[:,:-1]

# Load NSL_KDD test dataset
dfkdd_test = pd.read_table("KDDTest.txt", sep=",", names=columns)
dfkdd_test = dfkdd_test.iloc[:,:-1]
```

Training dataset

In [6]:
```
dfkdd_train.head(5)
```

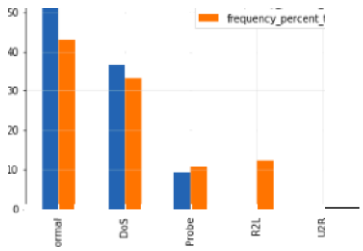
In [15]:
```
#Contains only 0's so drop

dfkdd_zest.drop(['num outbound cmds'], axis=1, inplace=True)
```

In [16]:
```
attack_class_freq_train = dfkdd_train[['attack_class']].apply(lambda x: x.value_counts())
attack_class_freq_test = dfkdd_test[['attack_class']].apply(lambda x: x.value_counts())
```

In [17]:
```
plot = attack class dist[['frequency percent train', 'frequency percent test']].plot(kind="bar")

pLoi.sei_iiile("Aiidck Class Disiribuiio " foniaize=22)
```

Ozi{1"]: maiploilib.legend.Legend ai Ox7T<aa45f66d0›

Attack Class Distribution



<F ig fl *re* s i *ze* 1440 x1440 or t fi 0 Ax es >

Feature Scaling

In [18]:
```
from sklearn.preprocessing import StandardScaler

sc_zesa = sparer.<ir_trans<om <df*dd_iesnseieci_divpes_inc1ude=['f',oai6n',‘i ni64']]
sc_train = scaler.fit_transform(dfkdd_train.select_dtypes(include=['float64','int64']))
```

Encoding

```
# extract categorical attributes from both training and test sets
cattrain = dfkdd_train.select_dtypes(include=['object']).copy()
cattest = dfkdd_test.select_dtypes(include=['object']).copy()
```

```
cat_Ytest = testcat[['attack_class']].copy()
```

Sampling

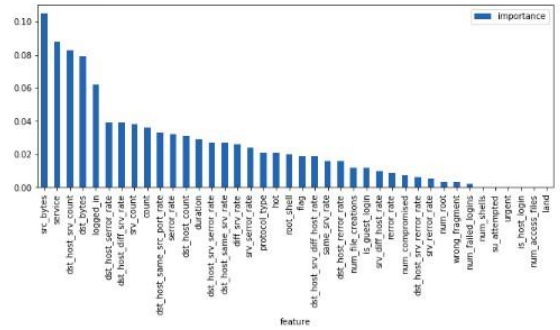
```
# define columns and extract encoded train set for sampling
sc_traindf = dfkdd_train.select_dtypes(include=['float64','int64'])
```

```
X_res, y_res = ros.fit_sample(X, y)
print('Original dataset shape {}'.format(Counter(y)))
```

```
Original dataset shape Counter({1: 67343, 0: 45927, 2: 11656, 3: 995, 4: 52})
Resampled dataset shape Counter({1: 67343, 0: 67343, 3: 67343, 2: 67343, 4: 67343})
```

Feature Selection

In [21]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier();

# fit random forest classifier on the training set
rfc.fit(X_res, y_res);
# extract important features
score = np.round(rfc.feature_importances_,3)
importances = pd.DataFrame({'feature':refclasscol,'importance':score})
importances = importances.sort_values('importance',ascending=False).set_index('feature')
# plot importances
plt.rcParams['figure.figsize'] = (11, 4)
importances.plot.bar();
```



In [22]:
```
from sklearn.feature_selection import RFE
import itertools
rfc = RandomForestClassifier()

# create the RFE model and select 10 attributes
rfe = RFE(rfc, n_features_to_select=10)
rfe = rfe.fit(X_res, y_res)

# summarize the selection of the attributes
feature_map = [(i, v) for i, v in itertools.zip_longest(rfe.get_support(), refclasscol)]
selected_features = [v for i, v in feature_map if i==True]
```

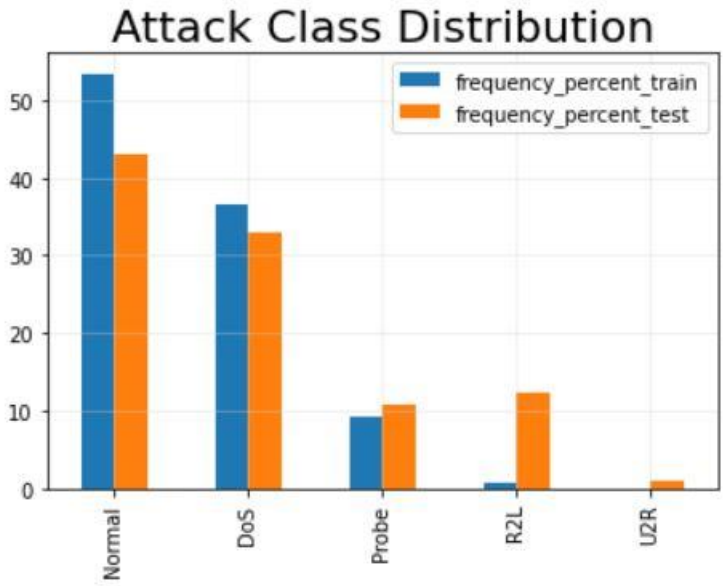
In [23]:
```
print(selected_features)
```

```
['src_bytes', 'dst_bytes', 'logged_in', 'count', 'srv_count', 'dst_host_srv_count', 'dst_host_diff_srv_rate', 'dst
_host_same_src_port_rate', 'dst_host_serror_rate', 'service']
```

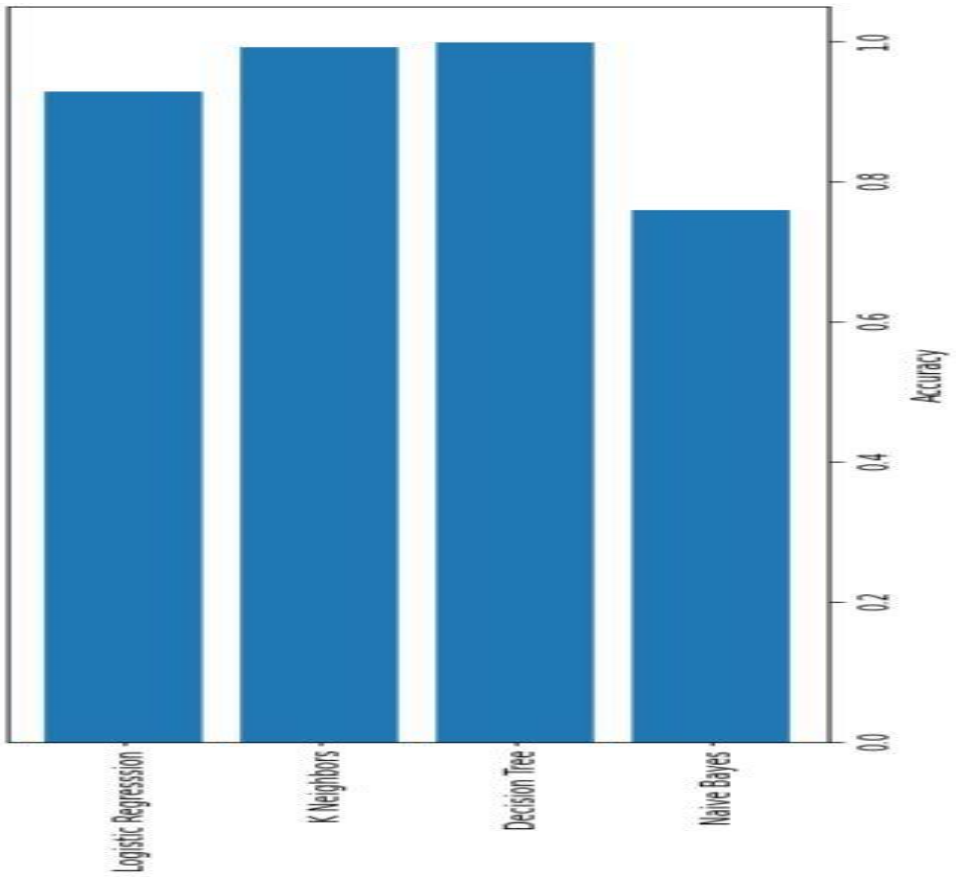
In [33]:
```
from sklearn.svm import SVC
from sklearn.naive_bayes import BernoulliNB
from sklearn import tree
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import VotingClassifier

# Train KNeighborsClassifier Model
KNN_Classifier = KNeighborsClassifier(n_jobs=-1)
KNN_Classifier.fit(X_train, Y_train);

# Train LogisticRegression Model
LGR_Classifier = LogisticRegression(n_jobs=-1, random_state=0)
LGR_Classifier.fit(X_train, Y_train);

# Train Gaussian Naive Baye Model
BNB_Classifier = BernoulliNB()
BNB_Classifier.fit(X_train, Y_train)

# Train Decision Tree Model
DTC_Classifier = tree.DecisionTreeClassifier(criterion='entropy', random_state=0)
DTC_Classifier.fit(X_train, Y_train);

# Train RandomForestClassifier Model
#RF_Classifier = RandomForestClassifier(criterion='entropy', n_jobs=-1, random_state=0)
#RF_Classifier.fit(X_train, Y_train);

# Train SVM Model
#SVC_Classifier = SVC(random_state=0)
#SVC_Classifier.fit(X_train, Y_train)

## Train Ensemble Model (This method combines all the individual models above except RandomForest)
#combined_model = [('Naive Baye Classifier', BNB_Classifier),
#                  ('Decision Tree Classifier', DTC_Classifier),
#                  ('KNeighborsClassifier', KNN_Classifier),
#                  ('LogisticRegression', LGR_Classifier)
#                 ]
#VotingClassifier =  VotingClassifier(estimators = combined_model,voting = 'soft', n_jobs=-1)
#VotingClassifier.fit(X_train, Y_train);
```

In [51]:
```
from sklearn import metrics

models = []
#models.append(('SVM Classifier', SVC_Classifier))
models.append(('Naive Baye Classifier', BNB_Classifier))
models.append(('Decision Tree Classifier', DTC_Classifier))
#models.append(('RandomForest Classifier', RF_Classifier))
models.append(('KNeighborsClassifier', KNN_Classifier))
models.append(('LogisticRegression', LGR_Classifier))
#models.append(('VotingClassifier', VotingClassifier))

acc = []
for i, v in models:
    scores = cross_val_score(v, X_train, Y_train, cv=10)
    accuracy = metrics.accuracy_score(Y_train, v.predict(X_train))
    acc.append(accuracy)
    confusion_matrix = metrics.confusion_matrix(Y_train, v.predict(X_train))
    classification = metrics.classification_report(Y_train, v.predict(X_train))
    print()
    print('============================== {} {} Model Evaluation ==============================='.format(grpclass, i))
    print()
    print ("Cross Validation Mean Score:" "\n", scores.mean())
```

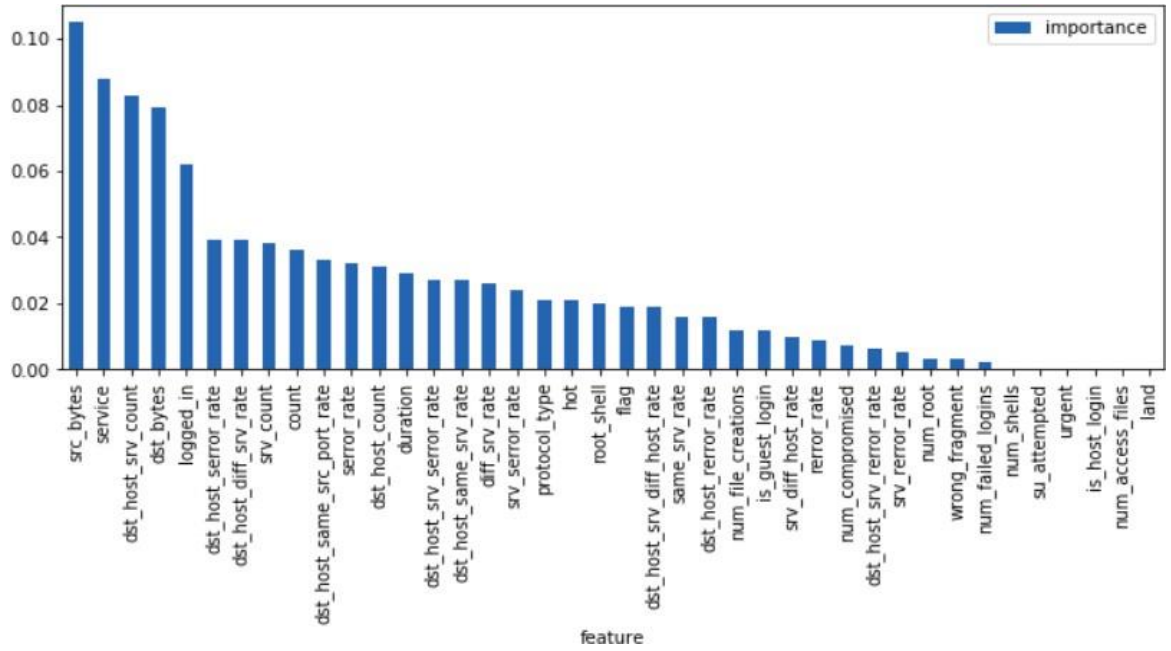
SNAPSHOTS



Frequencies of the attacks



Accuracy Comparison



Feature Importance

CONCLUSION

Intrusion Detection Systems (IDS) are the second layer of defence. It detects the presence of attacks within traffic that flows in through the holes punched into the firewall. An Intrusion Detection System (IDS) constantly monitors actions in a certain environment and decides whether they are part of a possible hostile attack or a valid use of the environment. The intrusion discovery and intrusion avoidance fields are amazingly powerful, with new discoveries, capacities, and models being made constantly. A lot of research on information representation techniques for intrusion location information is additionally right now being led. The conclusion, of this study has shown that the data mining methods generate interesting rules that are crucial for intrusion detection and prevention in the networking industry. We showed how intrusion detection can benefit from high performance computing techniques. This project attempts to address the problem of intrusion attack detection with the use of data mining supervised model.

In summary, the results from this study can contribute towards in improving the networking security.

Future Works

In future, it is possible to provide extensions or modifications to the proposed clustering and classification algorithms using intelligent agents to achieve further increased performance.

Apart from the experimented combination of data mining techniques, further combinations such as artificial intelligence, soft computing and other deep learning algorithms can be used to improve the detection accuracy and to reduce the rate of false negative alarm and false positive alarm.

Finally, the intrusion detection system can be extended as an intrusion prevention system to enhance the performance of the system.

REFERENCES

[1]. R. Heady, G. Luger, A. Maccabe, and B. Mukherjee. A Method To Detect Intrusive Activity in a Networked Environment. In Proceedings of the 14th National Computer Security Conference, pages 362-371, October 1991.

[2]. Biswanath Mukherjee, L Todd Heberlein and Karl N Levitt. Network Intrusion Detection, IEEE Network, May/June 1994, pages 26-41.

[3]. Terresa F Lunt. A survey of intrusion detection techniques. In Computers and Security, 12(1993), pages 405-418.

[4] Vokorokos, L. and A. Baláž. Host-based intrusion detection system. in Intelligent Engineering Systems (INES), 2010 14th International Conference on. 2010. IEEE.

[5] Chauhan, P. and N. Chandra, A Review on Hybrid Intrusion Detection System using Artificial Immune System Approaches. International Journal of Computer Applications, 2013. 68(20).

[6]. Sarmah, A., Intrusion detection systems; definition, need and challenges. 2001, October

[7] Modi, C., et al., A survey of Intrusion detection techniques in cloud. Journal of Network and Computer Applications, 2013. 36(1): p. 42-57.

[8] Latha, S. and S.J. Prakash. A survey on network attacks and Intrusion detection systems. in Advanced Computing and Communication Systems (ICACCS), 2017 4th International Conference on. 2017. IEEE.

[9] Debar H, Dacier M, Wespi A. A revised taxonomy of intrusion-detection systems.
Annales des Telecommunications. 2000;55(7–8): 361-337

[10] Gong Y, Mabu S, Chen C, Wang Y, Hirasawa K. Intrusion detection system combining misuse detection and anomaly detection using genetic network programming. In: ICCAS-SICE. 2009

[11] Posted on Dec 25, 2014 by Robert Moskowitz industry-topics/blog/network intrusion-methods-of attack#:~:text=Protocol%2DSpecific%20Attacks,%22)%20or%20malformed%20protocol%20messages.

[12] E. Packel, [internet] 2012 [cited 2013 July 10]. Available from: http://www.databreachlegalwatch.com/2012/05/cyber-warfare-and-collateral-damage-flame-malware-heats-up-data-security-threat/

[13] ESET. 2012 [cited 2013 July 10]. Available from: http://go.eset.com/us/threat-center/

[14] E. Messmer, Unique malware samples broke the 75 million mark in 2011 [internet]. 2012 [cited 2013 July 10].
Available from: http://www.networkworld.com/news/2012/022112-mcafee-malware-report-256316.html

[15] Computer Economics [internet]. 2007 [cited 2013 July 10]. Available from: http://www.computereconomics.com/article.cfm?id=1225

[16] Vilela, Douglas W. F. L.; Lotufo, Anna Diva P.; Santos, Carlos R. (2018). Fuzzy ARTMAP Neural Network IDS Evaluation applied for real IEEE 802.11w data base. 2018 International Joint Conference on Neural Networks (IJCNN). IEEE. doi:10.1109/ijcnn.2018.8489217. ISBN 9781509060146.

[17] Dias, L. P.; Cerqueira, J. J. F.; Assis, K. D. R.; Almeida, R. C. (2017). Using artificial neural network in intrusion detection systems to computer networks. 2017 9th Computer Science and Electronic Engineering (CEEC). IEEE. doi:10.1109/ceec.2017.8101615. ISBN 9781538630075.

[18] Inc, IDG Network World (2003-09-15). Network World. IDG Network World Inc.

[19] Brandon Lokesak (December 4, 2008). "A Comparison Between Signature Based and Anomaly Based Intrusion Detection Systems" (PPT). www.iup.edu.