# Ensemble Based Intrusion Detection System for Multi attack Environment

Satyapriya Raut[1], Aniketh Poojary[2], Aditya Naiknaware[3], Sushant Vairat[4], Shraddha R. Khonde[5]
*[1,2,3,4] Student, Computer Engineering, Modern Education Society's College of Engineering, Pune, India*
*[5] Assistant Professor, Student, Computer Engineering, Modern Education Society's College of Engineering, Pune, India*

***Abstract*- — Due to mass usage of Internet in today's era, cyber-attacks have been very common. These pose a serious threat to the organization as well as for an individual. The sensitive and confidential data that needs to be protected is at high risk and is stolen by attackers using various types of attacks. In multi attack environment, there would be more than one attack occurring simultaneously or within a short span of time. In our project, we have considered all those attacks as multi attacks which occur within one second of time span. We have proposed a system that captures live packets from the network and classifies whether the packet is normal or belongs to one of the subclasses of attack using various ensemble approaches such as Bagging, Boosting and Stacking. NSL-KDD dataset has been used for both training and testing the model. We found out that XGBoost outperforms with highest accuracy, 72.27%, followed by Random Forest classifier, 72.22%.**

***Index terms*- Ensemble, Intrusion Detection System, Machine Learning, XGBoost, Random Forest, Extra Tree, Bagging, Boosting, Stacking, NSL-KDD**

## I.INTRODUCTION

In today's era, no matter whether its a big organization or a small-scale industry or an individual, everyone is using Internet massively to carry out their daily business or personal work. Internet has simplified our work to a great extent, but along with it comes various threats and cyber-attacks. These attacks could be stealing of your data or maybe Denial of Services (DoS), so that the user is deprived from any kind of services or any of its kind or other attack as described in Table I. It is thus important to identify such malicious network packets and alert the user regarding the same.

In a given network, when there is more than one attack taking place, such an environment is called as multi attack environment. In this multi attack environment, the attacker tries to attack the victim's device using more than one attack. We have considered all those attacks to be multi attacks which occur within one second of time span and hence those attacks will be classified as multi attack as opposed to single attack. To achieve this, we sort the dataset according to the time stamp of packets and then divide the dataset into chunks where the time difference is one second.

In our proposed system, we have used NSL-KDD dataset, an improvised version of KDDCup99 dataset. The NSL-KDD dataset has several improvements [1] to the KDDCup99 dataset.

- Redundant observations in the training dataset are removed. This reduces the bias with most frequent records.
- Redundant observations in the testing dataset are also removed.
- Total records selected from each difficulty group are inversely proportional with respect to their percentage in the original KDDCup99 dataset. This results in a varied accuracy as well as performance of various machine learning models and hence makes the algorithm more efficient.
- The reduced number of records in training and testing datasets make it affordable to run various algorithms on the entire dataset, and not by selecting a random small sample from the dataset.

Other researchers [3],[4],[5] have shared the same views about the improvements of the NSL-KDD data set.

The NSL-KDD training set consists of 23 subclasses and that the testing set consists of 39 subclasses. Fig. 1 shows the distribution of the class of various attacks in NSL-KDD training and testing dataset.

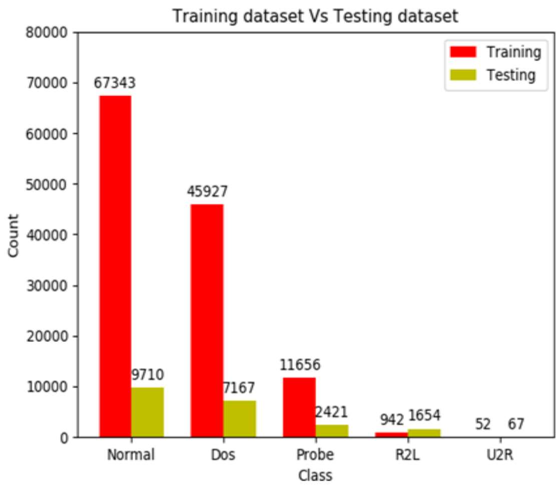

Fig. 1 Distribution of attacks in NSL-KDD Dataset

Table I shows various subclasses of attacks viz. DoS, Probe, R2L and U2R and their respective subclasses.

TABLE I ATTACKS IN NSL-KDD DATASET

| Sr. No | Class | Subclass |
|---|---|---|
| 1 | Denial of Service (DoS) | Back, Land, Neptune, Pod, Smurf, Teardrop, Mailbomb, Processtable, Udpstorm, Apache2, Worm |
| 2 | Probe | Satan, Ipsweep, Nmap,Portsweep, Mscan, Saint |
| 3 | R2L | Guesspassword,Ftp_write,Imap, Phf,Multihop,Warezmaster,Xlock,Xsnoop,Snmpguess,Snmpgetattack,Httptunnel,Sendmail, Named |
| 4 | U2R | Buffer_overflow,Loadmodule, Rootkit,Perl,Sqlattack,Xterm,Ps |

## II. ENSEMBLE APPROACHES USED

A. Bagging: Bagging is an ensemble approach in which number of homogenous weak classifiers are trained on random sample without replacement. The final predicted class of bagging is the majority predicted class of all the weak classifiers. We have used Bagging Classifier with base classifier as Decision Tree. Other trained classifiers include Random Forest and Extra Tree classifier.

B. Boosting: Boosting is an iterative ensemble approach where each weak classifier learns from the previous weak classifier. Here, the misclassified observations are given more weightage during random selection of samples. We have used Boosting Classifier with base classifier as Decision Tree. We also trained ensemble XGBoost classifier and got the highest accuracy compared to all the classifiers we used.

C. Stacking: Stacking is another ensemble approach that often considers heterogeneous weak learners, learns them in parallel and combines them by training a meta-model to output a prediction based on the different weak models' predictions [7]. In stacking, we have used Logistic regression, K-Nearest Neighbour and Support Vector Machine.

## III. DATA PRE-PROCESSING

NSL-KDD dataset contains total of 43 values per observation, with 41 of the features referring to the traffic input itself and the last two are labels telling the type of attack and Score which describes the severity of the traffic input itself.

All these 41 features are not required for classifying a record and hence pre-processing is to be done to reduce the dimensionality of the dataset. KDD Extractor [2] extracts 28 prominent features from a network packet along with 5 other features (Source IP, Destination IP, Source port no, Destination port no, Timestamp). Again, out of this 28 features some classifiers were trained on 25 features whereas some were trained on 28 features to achieve as much accuracy as possible.

The dataset with 28 features now, had a row with 27 missing features. Since it was only one row with missing values in the entire dataset, we just discarded it from the dataset.

The reduced dataset consisted of numerical as well as categorical features. These categorical features could not be given directly to the classifier and hence need to be encoded first. We use One Hot Encoding technique to encode the categorical features into numeric features. One Hot Encoder creates a binary column for each category and returns a sparse matrix. The number of columns in the sparse matrix is equal to the number of distinct values for that corresponding column.

## IV. WORKFLOW OF THE SYSTEM

1. Step 1 – Capturing .pcap file using Sniffer: From a given network, the packets are captured using packet sniffer and a .pcap file generated. This

.pcap file is used to diagnose network packets and help understanding them. Python scripts are used for achieving this task.

2. Step 2 – Generating .csv file from .pcap file using KDD Extractor: The .pcap file cannot be given to the ensemble classifier directly. Hence using the KDD Extractor [2] the data is extracted from the .pcap file according to the NSL-KDD dataset format.
3. Step 3 – Pre-processing of the .csv file: The .csv file generated contains some features not required by the classifier. These include (Source IP, Destination IP, Source port no, Destination port no, Timestamp). Hence those features are removed from the .csv file. Also, the categorical features are encoded using One Hot Encoding as explained in section III. Finally, a dataset is prepared, fit for the classifier.

The workflow of the project is depicted in figure 2.

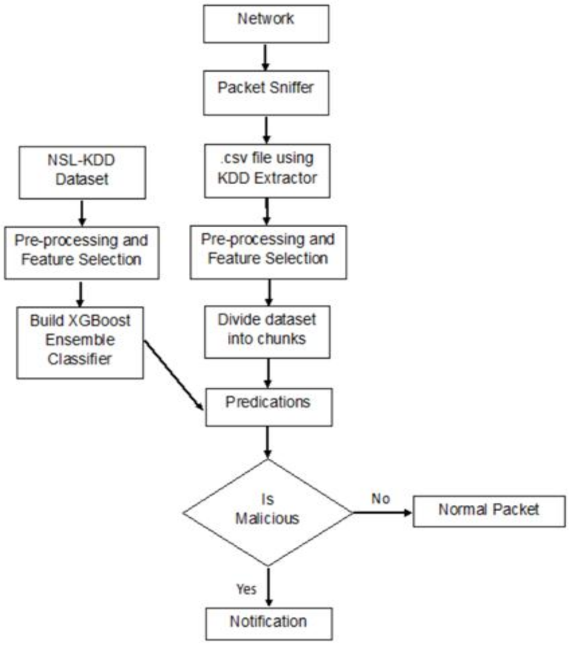
Network
Packet Sniffer
NSL-KDD Dataset
.csv file using KDD Extractor
Pre-processing and Feature Selection
Pre-processing and Feature Selection
Build XGBoost Ensemble Classifier
Divide dataset into chunks
Predications
Is Malicious
No
Normal Packet
Yes
Notification

Fig. 2 Workflow of the Ensemble based IDS

4. Step 4 – Division of dataset into chunks: For the attacks to be classified as multi attack, only those attacks are considered as multi attack which occur in one second of time span. The dataset is sorted in ascending order based on the timestamp of the network packets arrival time. They are then divided into various chunks.
5. Step 5 – Prediction using Ensemble Classifier: After experimenting with various ensemble methods we found that XGBoost gave the highest accuracy. The classifiers along with their corresponding accuracy is shown in Table 2. The dataset is then given to the XGBoost for prediction of network packets as malicious or normal.
6. Step 6 – Display of the results: After the prediction, if any malicious network packet is detected, then the user is alerted about the same using a pop-up window.

The classifiers used and their corresponding accuracy is shown in below Table II.

TABLE II ENSEMBLE CLASSIFIER USED AND THEIR ACCURACY

| Sr. No. | Ensemble Classifier Used | Number of Features used | Accuracy on KDDTest+.csv |
|---|---|---|---|
| 1. | XGBoost | 27 | 72.27% |
| 2. | Random Forest | 25 | 72.22% |
| 3. | Bagging with base as Decision Tree classifier | 27 | 71.81% |
| 4. | Extra Tree | 25 | 71.42% |
| 5. | Stacking | 25 | 71.34% |
| 6. | AdaBoost with base as Decision Tree classifier | 27 | 71.22% |

## V. CONCLUSION

In this project, the ensemble XGBoost classifier of machine learning is used to detect the multi attacks. We have considered attacks occurring within one second of time to be as multi attack. We experimented with various ensemble approaches such as Bagging, Boosting and Stacking. We predicted the KDDTest+.csv dataset with 28 features with various ensemble classifiers like XGBoost, Random Forest, Extra Trees, Bagging with base classifier as Decision Tree, Boosting with base classifier as Decision Tree and Stacking with Logistic Regression, Decision Tree, K-Nearest Neighbour, Support Vector Machine. Out of all these approaches we found that XGBoost gave the highest accuracy of 72.27% on KDDTest+.csv.

## VI. FUTURE WORK

This system can be further extended to detect attacks with timestamp of less than one second. It can also be further extended to detect attacks occurring simultaneously. More advanced ensemble methods of machine learning can be implemented to achieve high accuracy.

## REFERENCES

[1] "NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity UNB," 2017. [Online]. Available: http://www.unb.ca/cic/datasets/nsl.html. [Accessed: 10-December-2019].

[2] "KDDExtractor".https://github.com/AI-IDS/kdd99_feature_extractor

[3] L. Dhanabal and S. P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms," Int. J. Adv. Res. Comput. Commun. Eng., vol. 4, no. 6, pp. 446–452,2015.

[4] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-basedintrusion detection system through feature selection analysis andbuilding hybrid efficient model," J. Comput. Sci., vol. 25, pp. 152– 160, 2016.

[5] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in 2015 Military Communications and Information SystemsConference, MilCIS 2015 - Proceedings, 2015.

[6] Explaination of ensemble classifiers | Bagging | Bossting. https://analyticsindiamag.com/primer-ensemble-learning-bagging-boosting. [Accessed: 10-March-2020]

[7] Explaination of ensemble classifiers | Bagging | Bossting | Stacking. https://towardsdatascience.com/ensemble-methods-bagging- boosting-and-stacking-c9214a10a205