International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 09, September 2022

Attack Analysis and Prediction using Machine Learning

Utkarsha Bamane [1], Sumitra Pundlik [2]
[1]MTech Student, Information and Technology (Cyber Security), MIT ADT University, Pune, Maharashtra, India [2]Professor, Dept. of Information and Technology, MIT ADT University, Pune, Maharashtra, India

***Abstract*: The intentional breach of a defense policy is what intrusion detection is. In order to look for any malicious activities or threats, intrusion detection systems monitor network traffic passing across various types of computer systems and provide warnings when it detects any dangers. Systems for detecting threats should be able to identify every harmful software and event in the network. All forms of attacks, including incursion, file-less malware, botnets, and malware, are changing the threat environment. In order to identify harmful events by examining the program's behavioral pattern, a learning detection system is necessary. In this context, we have form the structure to specify the type of assault that machine learning has recognized. Malicious activity detection can be divided into two categories: signature-based detection and misuse detection. For both types of detection, an IDS must gather the necessary data, evaluate it, and then compare it to attack signatures kept in big databases.**

In our study, we suggested a method for creating effective IDS utilizing either the stacking algorithm or the decision tree algorithm. According to the results, the suggested method performs more accurately and efficiently than other methods like logistic regression and random forest. The accuracy rate (%) values for the outcomes produced by the suggested method are 99.36%. Attack analyzer system uses four different algorithms to check multiple types of protocols parameters and authenticate users. After that, it stacks methods with and without feature selection to assess the accuracy and choose the best algorithm to identify which kinds of attacks such as benign, DoS, port scans, brute force attacks, web attacks, bot attacks, and infiltration.

KeyWords: Threat detection, Machine Learning, IDS, malicious, datasets, Intrusion, classifiers, regression.

I. INTRODUCTION

Threats or malicious activity is found using an intrusion detection system (IDS). To protect a computer network, the IDS takes network-level defensive action. The danger or incursion always manifests itself as an anomaly in a network. The protection of a network is violated when an intruder takes advantage of system defects such as lax security rules, software issues like buffer overflows, and DoS attacks that exploit network flaws. The intruders could be cybercriminals, who are regular internet users who want to steal or harm extremely sensitive data from the victim's system, or they could be system users with fewer privileges who want to have more access to allowed data. The types of intrusion detection methods include signature-based and anomaly-based techniques. A specialized system or piece of software monitors packet flow in the network and compares it to earlier discovered, configured known signatures of known threats. This is known as signature-based detection. Comparing defined legitimate user parameters with occurrences that reveal divergence from the legitimate user parameters is how the anomaly detection technique finds assaults, in contrast. Whenever malicious behavior occurs in a network, the IDS creates logs and notifies the network administrator. Systems for detecting threats should be able to identify every harmful software and activity in the network. All forms of threats, including incursion, file-less malware, botnets, and malware, are changing the threat environment. In order to identify harmful events by examining the program's behavioral pattern, a learning detection system is necessary. Using machine learning and deep learning approaches, we have created models to recognize the malicious software and system events. Before generating the end outcome, ensemble is a technique for mixing the output of various algorithms.

1.1 OBJECTIVES

1) Threat detection systems that can accurately identify all malicious programmers and network events
2) The threat environment is changing for all forms of attacks, including intrusion, malware, file-less malware, and botnets. To identify malicious occurrences, it is necessary to use a learning detection system that examines the program's behavioral pattern.
3) Secure automatic threat detection and prevention scans the network and server functions and alerts the analyst if any suspicious behavior is found in the network traffic. This method is more efficient at reducing the burden of the analyst. It continuously monitors the system and reacts in accordance with the threat environment.
4Our technology uses a variety of machine learning methods to identify network intrusion. IDS keeps an eye out for malicious behavior and guards against unauthorized access from users, possibly even from insiders, to a computer network.
5) The danger or intrusion manifests as an anomaly in a network. Network faults are exploited by hackers that violate the security of the network by abusing network vulnerabilities like lax security regulations and software problems like buffer overflows.

II. HISTORY & BACKGROUND

At the center of the project is a machine learning algorithm. The most pertinent items are suggested to users via a recommendation engine, which filters the data using various techniques. It records the user's preferences and inclinations and then proposes alternatives that are consistent with those preferences.

2.1 Algorithm Used

2.1.1 *Extra tree Classifier:*

Extremely randomised trees (Extra Trees) are a component of ensemble learning methods. The decision trees are constructed by it. The decision rule is drawn at random during tree construction. With the exception of random split value selection, this algorithm's rule is quite similar to that of Random Forest.

2.2.2 *Decision Tree Classifier:*

Data input is classified as normal or anomalous using the decision tree classifier. A decision tree is a graph in the shape of a tree with core nodes that represent tests on attributes, branches that indicate the results of the tests, and leaf nodes that represent class labels. The route chosen from the root node to the leaf determines the classification models. The root node is divided first, then each input information. Decision trees are able to evaluate data and spot patterns in the network that point to malicious activity. By examining a significant amount of intrusion detection data, it can improve many real-time security systems. It can spot patterns and trends that aid in surveillance, attack signature generation, and other investigative tasks. Decision trees offer a rich set of guidelines that are simple to grasp and can be easily linked with real-time solutions. This is the fundamental benefit of utilizing decision trees instead of other classification systems.

2.2.3 *Random Forest Algorithm*

The suggested intrusion detection framework employs Random Forests as a classifier. According to empirical findings, developing an IDS that is successful and efficient for network intrusion detection is made possible by the Random Forests classifier with SMOTE and information gain-based feature selection.

2.2.4 *XGBoost Algorithm*

A gradient boosting framework is used by the ensemble machine learning method XGBoost, which is decision-tree based. Artificial neural networks frequently outperforms all other algorithms or frameworks in prediction problems requiring unstructured knowledge (pictures, text, etc.). But when it involves small to medium amounts of structured/tabular data, decision tree-based algorithms are right away regarded as best-in-class.

2.2.5 *Ensemble Algorithm:*

An ensemble machine learning approach called "Stacking," or simply "Stacking," uses generalisation. It entails using techniques like bagging and boosting to combine the predictions from various machine learning models on the same dataset. Stacking frequently takes into account diverse weak learners, trains them in parallel, and then combines them by teaching a meta-learner to produce estimates based on the predictions of the diverse weak learners.

LITERATURE REVIEW

[1] Correlation-based feature selection (CFS-BA) Ensemble technique, which consists of the C4.5, Random Forest (RF), and Forest by Penalizing algorithms (Forest PA) [1] in this study's outlier detection method, the neighbourhood outlier factor is used to measure the dataset of anomalies (NOF). The trained model in this instance uses a distributed storage infrastructure and large datasets to improve the effectiveness of the intrusion detection system. The outcomes of the experiments demonstrated that the suggested approach finds abnormalities far more accurately than any other approaches. [1]

[2] This study introduces an efficient and automatic network monitoring system that keeps track of all network switches and notifies the administrator through email or SMS whenever a network switch fails. Additionally, this system indicates where the topology of the network is problematic and how it affects the remainder of the network. In a Linux context, this network monitoring solution makes use of the clever interplay between Request Tracker (RT) and Nagios software. In Nagios, the network architecture is constructed, and every network node is continuously monitored according to the services assigned to them. The RT software receives a notification from Nagios as soon as a network node fails. With details about the faulty node and how it affects the rest of the network, this message will create a ticket in the RT database. The RT software is set up to immediately transmit the ticket to the network administrator through email and SMS after it is produced. According to the stated priority, if the administrator is currently busy and does not resolve the complaint within an hour, the same issue is immediately sent to the second network responsible person. As a result, each person on the priority list is notified individually until the problem is addressed. [2]

[3] Secure automated threat detection and prevention scans the network and server functions and alerts the analyst if any suspicious activity is found in the network traffic. This method is more successful at reducing the workload of the analyst. It continuously monitors the system and reacts in accordance with the threat environment. From phase to phase, this reaction action changes. In this case, suspicious activity is discovered with the aid of artificial intelligence, which serves as a virtual analyst while working with network intrusion detection systems to protect against the threat environment and take appropriate action with the analyst's approval. Its final phase entails performing packet analysis to look for attack vectors and classifying both supervised and unstructured data. Wherein the algorithm will be automatically updated after the unsupervised data has been decoded or converted to supervised data with the assistance of analyst feedback (Virtual Analyst Algorithm). In order for the algorithm to improve over time by becoming stronger and more efficient, it uses an active learning mechanism. As a result, it can fight against similar or identical attacks [3].

[4] Numerous public and commercial enterprises are in danger due to malicious insider activity. In this research, a novel method for identifying malevolent behaviour is presented. Textual session-based data samples are the granularity level we suggest using to describe user log data. Character embedding and a deep learning model made up of

CNN and LSTM are used to model the user's behaviour. Using characters embedding, the input samples are represented. Then, local tri-gram features are extracted from the input samples using a convolution layer, and the order of these features is taken into account using an LSTM layer (tri-grams). We run tests using a variety of model designs that lack any custom features. A portion of the CERT Insider Threat dataset, version 4.2, is used to evaluate the proposed model.

H/W AND S/W REQUIREMENTS:

- Operating system: Windows 7/ WINDOWS 10
- Language: PYTHON- DJANGO
- Software with version: VS CODE 1.48.2
- Database Proposed: SQLITE /MySQL

III. PROPOSED SYSTEM ARCHITECTURE

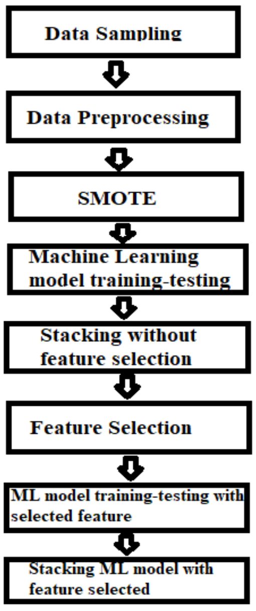

Data Sampling
Data Preprocessing
SMOTE
Machine Learning model training-testing
Stacking without feature selection
Feature Selection
ML model training-testing with selected feature
Stacking ML model with feature selected

Fig.1. Project Flow

The real project flow, starting with data sampling that is still stacking, is depicted in Fig. 1 above. It also includes the technique used for feature selection.

A training dataset can be transformed using a variety of approaches provided by data sampling in order to balance or better balance the class distribution. The newly converted dataset can be trained directly using normal machine learning techniques after it has been balanced. This enables the difficulty of imbalanced categorization to be treated and overcame using a data preparation strategy, even with substantially imbalanced class distributions.

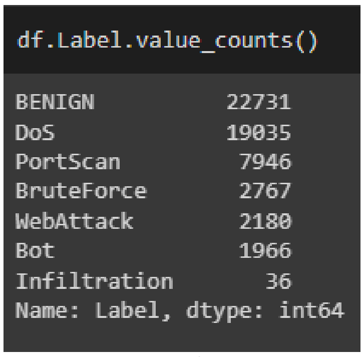
Data pre-processing is the process of transforming raw data into something that can be utilised to train or test a machine learning model. The initial and most important stage in developing a machine learning model is this one. We rarely see clean, organised data when developing a machine learning project. Additionally, any time you work with data, you need to cleanse it up and prepare it. So, in order to do this, we pre-process data.

The most popular oversampling technique used to address the imbalance issue we previously addressed is called SMOTE (synthetic minority oversampling technique). By boosting the random replication of minority class cases, it seeks to balance class distribution. SMOTE combines already existing minority instances to create new minority instances. For the minority class, it creates virtual training records using linear interpolation. By randomly choosing one or more examples from the minority class, these synthetic training records are created.

Stacking, also known as Stacked Generalization, Exploring a range of several models for the same problem is the goal of stacking. The concept is that you can utilise a learning problem with various sorts of models that can only learn a portion of the problem—not the entire problem field. In order to create an intermediate prediction, you can design numerous learning machines, each of which you utilise to make a single forecast for each taught model. Then you incorporate a fresh model that has the same aim that will gain knowledge from the earlier predictions. The actual objective and the anticipated target will be compared.

This last form is described as being layered on top of one another. As a result, it enhances overall performance and frequently results in a model that is superior to each particular intermediate model. The advantage of this over a single Notice, as is frequently the case with any machine learning technique, is that it does not provide you with any guarantees.

A subset of pertinent features are chosen through the feature selection (or attribute selection) procedure to be used in the model construction [15]. In order to avoid dimensionality in machine learning (ML), boost generalisation by lowering variance, and save training time, feature selection approaches are used. When using the feature selection technique on data, it is common for the data to still have traces of characteristics that are redundant or unnecessary but can be deleted without significantly affecting the data's quality.



```
df.Label.value_counts()

BENIGN          22731
DoS             19035
PortScan         7946
BruteForce       2767
WebAttack        2180
Bot              1966
Infiltration       36
Name: Label, dtype: int64
```

Fig.2. Total number of records per attack

With the training dataset's full set of features, four different single classifiers are trained, and predictions are made. Table I displays the accuracy results for the various training methods. Decision tree accuracy is 99.36%, Random Forest accuracy is 98.04%, Extra Tree Classifier accuracy is 98.89%, and XGBoost accuracy is 97.07% according to the decision algorithm. All four algorithms employ the ML staking approach. The total output from each classifier is used as input for the staking procedure, which returns a value of 99.36%.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 09, September 2022

Table I. Results comparison without feature selection

Method	*Accuracy Rate (%)*
Extra Tree Classifier	98.89
Decision Tree	99.36
XGBoost	97.07
Random Forest	98.04
STACKING	99.36

The four classifiers' importance are averaged to select the features. Four classifiers use chosen features to calculate accuracy. The accuracy of each algorithm is listed below. With the chosen feature, Random Forest and Extra Tree classifiers performed well.

Table II. Results comparison with feature selection

Method	*Accuracy Rate (%)*
Extra Tree Classifier	98.28
Decision Tree	99.12
XGBoost	96.59
Random Forest	99.20
STACKING	98.36

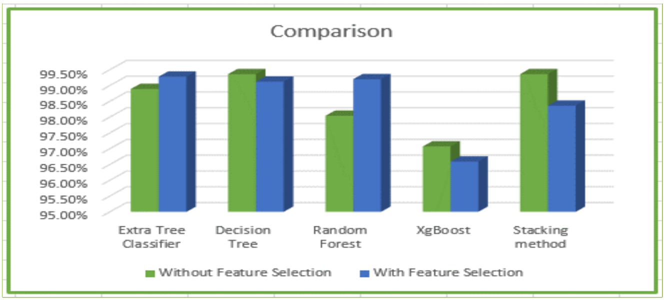


Fig.3. Comparison of algorithm with & without feature selection

The graphical representation of the value acquired for each algorithm based on its accuracy is shown above Fig. 3. It can be observed that the suggested approach was successful.
In our project, the Attack Analyzer system authenticates the user and adds data value that uses a decision tree algorithm to identify several attack kinds, such as "Benign," "DoS," "PortScan," "BruteForce," "WebAttack," "Bot," and "Infiltration," before identifying infiltration.

Final Output- which types of attack is detected

Attack Types	Benign	DoS	Port Scan	Brute Force	Web Attack	Bot	Infiltration

IV. CONCLUSION

As there is an increase in devices or large system over internet, the security concerns have also been increased. A learning detection system is required to detect malicious events by analyzing the behavioral pattern of the program. The proposed algorithms decision tree and stacking method has performed well as compared to random forest, extra tree classifier xgboost without any feature selection method implemented. The result obtained by our proposed method has the Accuracy rate (%) is 99.36%.

V. FUTURE SCOPE

In the future, we plan to hybridize our approach with other machine learning techniques and integrate them with different algorithm by calculating its performance and error rate to develop a real-time adaptive intrusion detection system that can efficiently detect attacks.

REFERENCE

[1] Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach International conference on Intelligent Computing, Communication & Convergence (ICCC-2014)

[2] D. Ten, S. Manickam, S. Ramadass, and H. A. Bazar, “Study on Advanced Visualization Tools In Network Monitoring Platform,” in Third UKSim European Symposium on Computer Modeling and Simulation, EMS ‘09’, Minden Penang, Malaysia, December 2009.

[3] L. Chang, W.L. Chan, J. Chang, P. Ting, M. Netrakanti, “A network status monitoring system using personal computer,” presented at IEEE Global Telecommunications Conference, August 2002.

[4] Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach International conference on Intelligent Computing, Communication & Convergence (ICCC-2014)

[5] J. Brownlee, “A Tour of Machine Learning Algorithms”, https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/ 2013

[6] E. K. Viegas, A. O. Santin, and L. S. Oliveira, “Toward a reliable anomaly-based intrusion detection in real-world environments,” Comput. Networks, vol. 127, pp. 200–216, 2017.

[7] A. Verma and V. Ranga, “Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine learning,” Procedia Comput. Sci., vol. 125, pp. 709–716, 2018.

[8] T. Hamed, R. Dara, and S. C. Kremer, “Network intrusion detection system based on recursive feature addition and bigram technique,” Computer. Secure. vol. 73, pp. 137–155, 2018.

[9] C. R. Wang, R. F. Xu, S. J. Lee, and C. H. Lee, “Network intrusion detection using equality constrained-optimization-based extreme learning machines,” Knowledge-Based Syst., vol. 147, pp. 68–80, 2018.

[10] G. Fernandes, L. F. Carvalho, J. J. P. C. Rodrigues, and M. L. Proença, “Network anomaly detection using IP flows with Principal Component Analysis and Ant Colony Optimization,” J. Netw. Comput. Appl., vol. 64, pp. 1–11, 2016.

[11] U. Ravale, N. Marathe, and P. Padiya, “Feature selection based hybrid anomaly intrusion detection system using K Means and RBF kernel function,” Procedia Comput. Sci., vol. 45, no. C, pp. 428–435, 2015.

[12] V. Hajisalem and S. Babaie, “A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection,” Computer Networks, vol. 136, pp. 37–50, 2018.

[13] C. Khammassi and S. Krichen, “A GA-LR wrapper approach for feature selection in network intrusion detection,” Comput. Secur., vol. 70, pp. 255–277, 2017.

[14] M. R. Gauthama Raman, N. Somu, K. Kirthivasan, R. Liscano, and V. S. Shankar Sriram, “An efficient intrusion detection system based on hypergraph - Genetic algorithm for parameter optimization and feature selection in support vector machine,” Knowledge-Based Syst., vol. 134, pp. 1–12, 2017.

[15] S. Shitharth and D. Prince Winston, “An enhanced optimization based algorithm for intrusion detection in SCADA network,” Comput. Secur., vol. 70, pp. 16–26, 2017.

[16] S. M. Hosseini Bamakan, H. Wang, T. Yingjie, and Y. Shi, “An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization,” Neurocomputing, vol. 199, pp. 90–102, 2016.

[17] D. C. Le and A. N. Zincir-Heywood, “Evaluating insider threat detection workflow using supervised and unsupervised learning,” in IEEE Security and Privacy Workshops, 2018.

[18] P. A. Legg, O. Buckley, M. Goldsmith, and S. Creese, “Automated insider threat detection system using user and role-based profile assessment,” IEEE Systems Journal, vol. 11, no. 2, 2017.

[19] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, “Deep learning for unsupervised insider threat detection in structured cybersecurity data streams,” in AAAI Workshop on AI for CyberSec., 2017.

[20] B. Bose, B. Avasarala, S. Tirthapura, Y. Y. Chung, and D. Steiner, “Detecting insider threats using radish: A system for real-time anomaly detection in heterogeneous data streams,” IEEE Systems Journal, 2017.