A Detailed Investigation and Analysis of using Machine Learning Techniques for Intrusion Detection

K. Gopi[1], B. Ravali[2]

[1]Asst. Prof, Dept of CSE, Tirumala Engineering College, Narasaraopet, Guntur, A.P., India

[2]PG Scholar, Dept of CSE, Tirumala Engineering College, Narasaraopet, Guntur, A.P., India

ABSTRACT - One of the critical cyber-security concerns of today is intrusion detection. A lot of machine learning-based strategies have been created, and some of them are pretty innovative. But they are unable to detect all kinds of break-ins. Machine learning approaches have been carefully explored and analysed in this work to discover why they have difficulty identifying the cause of interference in machine learning techniques. Each attack has a unique attack categorization and attacks feature mapping that is provided. The many categories of attacks can be detected using the techniques of machine learning, which have been examined and compared in terms of their effectiveness. A discussion of the restrictions imposed by each group is included. The article also features several different machine learning tools. The end of the paper provides possible machine learning-based approaches to attack detection. It also describes how to improve the detection of low-frequency assaults through network attack datasets.

Keywords: Intrusion detection, Machine learning, SVM

I. INTRODUCTION

Intrusion detection monitors and analyses incidents within the system to detect when someone is inappropriately using a computer or network. A computer system intrusion detection system (IDS) employs modelling and recognition techniques to identify malicious behavior. Any action that falls outside the regular, expected system operation for a given system could be called intrusive. The issues of intrusion detection, fraud detection, and fault management/localization are all closely linked. These areas are not explicitly addressed here, but they have a lot of overlap, especially regarding event correlation. The following are various intrusions, which makes it hard to come up with a single definition.

The general goal of DoS attacks is to prevent particular requests from being addressed on a host, which can be accomplished by interrupting a service. It's possible that this is just one part of a more powerful attack, like the Mitnick attack, which is explained below, or that it's designed to crash a host or make it impossible to use correctly, like the Slammer worm.

The U2R exploits system vulnerabilities to attain root access to an operating system. Kendall (1999) also talks about buffer overflow attacks: A buffer overflow occurs when a program transfers too much data into a static buffer without first ensuring that the data will fit.

R2L: This class of attack shares some parallels with U2R, with both types of intrusion attacks possible. In this instance, the intruder doesn't have an account on the host, and they're trying to get a network connection to get local access. This is possible through buffer overflow assaults, exploited security policy misconfigurations, or social engineering.

Intruder scans for vulnerabilities in software and setups, attempting to acquire information about target computers. Additionally, this involves password cracking.

This report doesn't examine the unique architectures of IDSs because these vary and are subject to constant change. According to Verwoerd and Hunt (2002), a few elements found in most IDSs are as follows:

Sensor probes: acquire information from the inspected system. To alert the appropriate authorities of a possible security breach, the monitor processes various data and then sends suspicious information to a "resolver."

Resolver: decides on a proper response to potentially harmful content.

The controller provides administering functions. This section expands on the three main points presented previously. A taxonomy of IDSs classifies the devices into four categories, as outlined in (Kruegel et al. 2004, pp. 20–21):

Decide on a host- or network-based audit source location. The detection approach is either the detection of misuse or anomalies. Whether they take a quiet or assertive approach when they are discovered:

Real-time or offline usage frequency. And the detection strategy, which details the strategies utilized to identify intrusions, is also discussed. The detection method for misuse is connected to this topic, and a certain amount of discussion of it is found in (Kruegel et al. 2004, pp. 20–21) in the field of misuse detection systems. Although this detection approach is not limited to misuse detection, it is considered a different characteristic for this discussion. The following paragraphs will look at the five properties in detail.

II. RELATED WORK

According to polls, many organizations are using machine learning to fight network infiltration. It mentions a few of their achievements. The unique aspects of our work distinguish our projects. According to Agrawal et al. [9], data mining can be used to locate anomalies for intrusion detection. They have grouped anomaly detection approaches based on the three factors of clustering, classification, and hybrid techniques. K-means. Methods for EM clustering, K-Meoids, and the detection of outliers have been detailed under the clustering category. The categorization approaches include Naive Bayes Algorithm, Genetic Algorithm, Neural Networks, and Support Vector Machine. Combined machine learning techniques might be referred to as hybrid methods. They summarized the ensemble techniques used in studies with a brief comparison.

In their survey on machine learning applications in intrusion detection, Haq et al. [10] described several applications. They described a few classifier and ensemble techniques without any critical analysis or observations. The methodologies have been divided into three groups: supervised learning, unsupervised learning, and reinforcement learning. A classifier is trained on a labeled dataset during supervised learning. Supervised learning requires labeled data; unsupervised learning is practical when the labeled data is lacking. In the field of reinforcement learning, a domain expert might assign labels to cases that have yet to be categorized.

Another method for defining territory boundaries is via a detached neighborhood, as suggested by KaiYan Feng [11]. Point B is the closest dormant kth-organized nearest neighbor of An, which is also equitable and just if An is the kth nearest neighbor of B. Every time a question is raised, its genuine nearest neighbor and unapproachable nearest neighbors are learned first. Then, for each class, a general score is calculated based on these discoveries. The class score decides if request guides are likely to have a place with that class.

Vincenzo Gulisano et al. [19] presented a way to screen quick traffic by adding IP addresses to route the information to single out a set of irregulars. Zhang Fu, Marina Papatrianta, et al. [8] suggest using variable and static clock coast to verify synchronous correspondence. The hoped-for technique will employ a clock scan that utilizes speed and accuracy to choose support strikes. According to Li Jimin et al. [12], the speed and precision of machine tool toolpaths can be improved with an SVM framework employing a tri-state scheme planning. Monowar H. Bhuyan [13] approach has introduced a tree-based method to identify bunches that do not utilize named data. A CLUS framework is utilized to tag the information utilizing the gathering method. The recommended framework results in improved mixed-kind framework data and numeric data. Carlos A. Catania and Carlos Garino demonstrated a technique for reducing preparation time, with spare time being used to create the foundation of the algorithm's framework.

The Apriori approach in [14] uses a calculation to generate a rule for the known information objects, and the results are expected to be of particular help and assistance. This computation can be referred to as an expanded Apriori calculation. The Apriori calculation uses a subset of a continuous assault item set [14]. As for this character, construct the small-scale attack set if the general thing sets are inadequate to the character, and then erase them. After that, we may achieve improved productivity. As described by [16], a half-and-half computation has been put out for intrusion detection with highlight determination calculation. The strategy is a module of affiliation and strives to reduce the number of attributes connected with each data set. One may observe the use of the K-implies algorithm in Intrusion Detection Calculations.

III. PROPOSED ARCHITECTURE

the design we are proposing The accompanying key modules for IIDSS (Intelligent Intrusion Detection Security System) is present. In other words, they are Preparation-Previously pre-handled information is immediately available for the attack's execution, and attack-related information is gathered from the KDD Cup data set. The selection of data for identifying specific incidents depends on an investigation of about 23 distinct assaults. Pre-preparing Pre-processing of data was accomplished using the WEKA software, using the RemoveUseless() method to eliminate unnecessary qualities. Only 41 of the total 41 traits could be dispensed with for the overall performance increase in the framework. Arrangement Simple K Nearest Neighbors Classification can classify types of assaults, which makes those assaults more accurate. Location of interruption Figure 1 displays the modules sent.

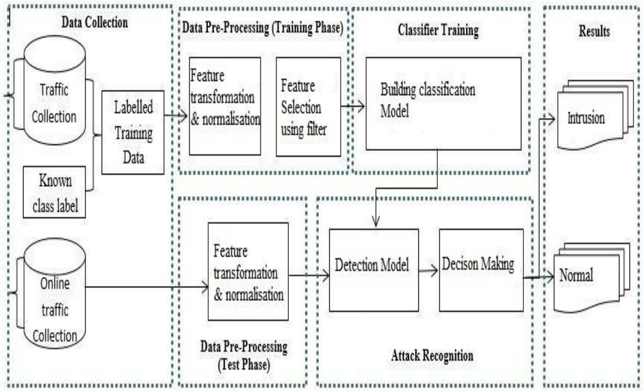
Training Data Phase
Data Collection
Traffic Collection
Known class label
Labelled Training Data
Data Pre-Processing (Training Phase)
Feature transformation & normalisation
Feature Selection using filter
Classifier Training
Building classification Model
Results
Intrusion
Test Data Phase
Online traffic Collection
Feature transformation & normalisation
Data Pre-Processing (Test Phase)
Detection Model
Decison Making
Attack Recognition
Normal

Figure 1: Proposed IDS Architecture

IV. RESULTS AND OBSERVATION

Recent Snort rules have been evaluated using real-time traffic and simulations and proven successful after being used for a week on live traffic. We downloaded the files from the company's internal network. The daily attack count can be seen in Table 1. Snort has detected 210 attacks on day one, including 884 TCP packets and 34 UDP packets. It takes 0.0037 milliseconds to find TCP packets and 0.0001 milliseconds to find UDP packets.

Table 1: Attacks Detected Using Proposed IDS

DAY	IP Protocol Types	Count	No. of Packets	Rate (ms)	Attacks Detected
DAY1	UDP	884	918	0.0037	210
	TCP	34		0.0001	
DAY2	UDP	1079	1739	0.0044	239
	TCP	660		0.0027	
DAY3	UDP	1233	1286	0.0128	189
	TCP	53		0.0006	
DAY4	UDP	1619	1896	0.0034	197
	TCP	277		0.0006	
DAY5	UDP	1821	1950	0.0035	213
	TCP	129		0.0002	
DAY6	UDP	1713	1742	0.0071	235
	TCP	29		0.0001	
DAY7	UDP	2184	3161	0.0076	359
	TCP	977		0.0034	

V. CONCLUSION

The proposed Intrusion Detection display has been shown to deliver superior results to the current methods. This method has been shown to increase the staff's productivity by minimizing the fake alarm rate and reducing managerial effort. This model has created higher affectability (compared to C4.5, C4.5+ACO, and EDADT) of 13.24 percent, 10.55 percent, and 2.95 percent, respectively. In contrast to the current framework, the experimental result reveals improved precision. Future IDS work will be conducted to determine the number of assaults better, and the total number of assaults can be expanded from 23 to 40. Because intrusion detection systems are highly adept at detecting network traffic data packets, they are ideal for monitoring them. This study shows that deviations in packet behavior trigger alarms. The patterns are checked to the snort rules signature base to see if they match. The system under consideration was rigorously put to the test and compared to existing Snort rules. The proposed rules demonstrated themselves to be more accurate and efficient. Future research will focus on implementing powerful data mining and machine learning methods to discover new threats using a large volume of data.

VI. REFERENCES

[1] K. S. Desale and R. Ade, "Genetic algorithm-based feature selection approach for an effective intrusion detection system," in 2015 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, pp. 1-6, 2015.

[2] M. Monshizadeh and Z. Yan, "Security-Related Data Mining," in IEEE International Conference on Computer and Information Technology, Xi'an, pp. 775-782, 2014.

[3] A. D. Pietro et al., "Dynamic deep packet inspection for anomaly detection," US Patent 2017099310 (A1), 6 Apr. 2017.

[4] J. Vasseur et al., "Anomaly detection in a network coupling state information with machine learning outputs," US Patent 20170104774 (A1), 13 Apr. 2017.

[5] A. D. Pietro et al., "Signature creation for unknown attacks," US Patent 20160028750 (A1), 28 Jan. 2016.

[6] N. Yadav et al., "Network behavior data collection and analytics for anomaly detection," US Patent 20160359695 (A1), 8 Dec. 2016.

[7] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, "From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods," in IEEE Communications Surveys & Tutorials, vol. 20, no. 4, pp. 3369-3388, 2018.

[8] B. G. Atli, "Anomaly-based intrusion detection by modeling probability distributions of flow characteristics," 2017.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436-444, 2015.

[10] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: theory and applications," in Neurocomputing, vol. 70, no. 1-3, pp. 489-501, 2006.

[11] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set", Information Security Journal: A Global Perspective, vol. 25, no. 1-3, pp. 18-31, 2016.

[12] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to Information Retrieval," in Natural Language Engineering, 16(1), pp. 100-103.

[13] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," in Machine Learning, vol. 46, no. 1-3, pp. 389–422, 2002.

[14] P. Laskov, C. Gehl, S. Krüger and K. Müller, "Incremental support vector learning: Analysis, implementation and applications," in The Journal of Machine Learning Research, vol. 7, pp. 1909-1936, 2006.

[15] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," in Computers & Security, Volume 31, Issue 3, May 2012, pp. 357-374.

[16] 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 Systems Conference (MilCIS), Canberra, ACT, 2015, pp. 1-6.

[17] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, 2018, pp. 108-116.

[18] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in ACM CoNEXT 2010, Philadelphia, PA, 2010, pp. 8:1-8:12.

[19] M. M. Rahman and D. N. Davis, "Addressing the Class Imbalance Problem in Medical Datasets," in International Journal of Machine Learning and Computing vol. 3, no. 2, pp. 224-228, 2013.

[20] scikit-learn Machine Learning in Python, https://scikit-learn.org/stable/