Journal of Advances and Scholarly Researches in Allied Education
Vol. 21, Issue No. 7, October-2024, ISSN 2230-7540

DOI: https://doi.org/10.29070/mzrc2k24

A Defensive Framework Against Metigatic Generic Attacks Targeting Network Imbalance in Machine Learning-Based Cyber Defense Systems

Vineeta Shrivastava [1] * , Dr. Sakshi Rai [2]

1. Research Scholar, Department of Computer Science and Engineering, LNCT University, Bhopal, M.P., India
shrivastavavinita21@gmail.com ,
2. Professor, Department of Computer Science and Engineering, LNCT University, Bhopal, M.P., India

Abstract: The increasing complexity of cyber threats is making it harder for traditional intrusion detection systems to identify adaptive and subtle attacks. Machine learning (ML)-powered network intrusion detection systems (NIDS) are more capable of detecting threats, but they may be compromised by generic attacks. This can result in network imbalances and poor model performance since hostile traffic is under-represented. In order to improve ML-based NIDS, this research proposes a metigatic defensive architecture. The framework incorporates methods such as adversarial training, dataset balancing, feature preprocessing, feature engineering, and model fine-tuning. We construct realistic adversarial traffic samples with feature interdependencies and protocol compliance to evaluate and enhance the resilience of the model. Tested on the UNSW-NB15 and NSL-KDD datasets, the results demonstrate considerable improvements in accuracy, recall, and precision, particularly for under-represented or adversarial perturbed attacks. According to the results, the proposed architecture mitigates the impact of traffic imbalance brought on by generic attacks, providing a workable and scalable approach for robust intrusion detection in dynamic network environments.

Keywords: Metigatic Generic Attack, Network Imbalancing, Machine Learning, Network Intrusion Detection System (NIDS), Adversarial Training, Dataset Balancing

---------- X ----------

INTRODUCTION

There has been a meteoric rise in the importance of network security due to the proliferation of cloud-based infrastructures and digital connections. Traditional security measures, such as firewalls and rule-based intrusion detection systems (IDS), don't always work against increasingly complex cyberattacks. More and more, these problems are being addressed by Network Intrusion Detection Systems (NIDS) that rely on Machine Learning (ML) [1]. This is mainly because these systems can learn assault patterns, adjust to new dangers, and spot anomalies instantly.

Network Imbalancing and Generic Attacks

"Network imbalance" happens when "good traffic" significantly outnumbers "bad traffic" in datasets, and it poses a significant problem for ML-based NIDS. In certain cases, it might be difficult to identify generic attacks because of this imbalance [2]. Generic attacks are assaults that make advantage of system vulnerabilities without using unique signatures. The flexibility of generic attacks causes machine learning models to become less accurate in their classifications, which in turn weakens the models.

Metigatic Approach

In addition to taking advantage of network vulnerabilities, metigatic generic assaults are disruptive or hostile methods that undermine the performance of network intrusion detection systems (NIDS) by creating imbalanced circulations of traffic. These kinds of attacks have the potential to confuse machine learning algorithms, leading them to incorrectly identify potentially hazardous traffic as harmless. It has been proposed that a solution might be a Metigatic framework, which is designed to eliminate imbalances via the use of complex preprocessing techniques, resampling methods, and powerful machine learning algorithms [3].

Role of Machine Learning in NIDS

Support Vector Machines (SVM), Decision Trees, Random Forests, and Deep Learning architectures are some of the most popular machine learning (ML) models used in network intrusion detection systems (NIDS) for attack detection. On the other hand, they are unable to perform their duties effectively unless the training data is balanced and they are able to generalize to a variety of assaults [4]. The Metigatic framework is designed to increase the accuracy of intrusion detection and resistance against generic attacks by integrating resampling techniques, ensemble learning, and cost-sensitive classifiers. This is the purpose of the framework.

Although several studies have concentrated on network intrusion detection systems that use machine learning (ML-based NIDS), the effects of a generally imbalanced network caused by attacks have received very less attention. Unfortunately, the methodologies that are currently being used do not take any measures to combat adversarial strategies that make use of dataset dispersion. Instead, they focus on feature optimization and anomaly detection. In order to bridge this gap, it is required to implement a Metigatic ML-based NIDS system. This system should include data-balancing strategies together with powerful ML classifiers.

OBJECTIVES

1. To examine how network traffic imbalance in ML-based NIDS is affected by generic assaults.

2. To create and assess a mitigation architecture that reduces imbalance and improves the accuracy of intrusion detection

RESEARCH METHODOLOGY

Metigatic generic assaults are a kind of network attack that aims to degrade the effectiveness of ML-based NIDS by intentionally interfering with traffic patterns. Because these attacks change the ratio of malicious to benign traffic instead of targeting system vulnerabilities directly, training data does not reflect attack patterns properly. This incongruity makes the model less vigilant against serious but infrequent dangers. Because of this, malicious traffic becomes easier to identify and false negatives become more common. This obstacle can only be surmounted with a system that can quickly adapt to varying traffic loads and identify malicious inputs.

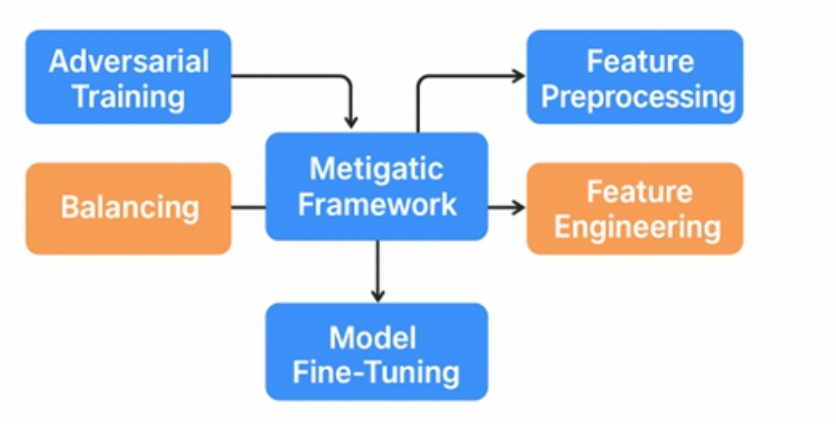

Figure 1. Methodology Framework

NIDS based on ML

Network intrusion detection systems (NIDS) powered by machine learning are able to spot harmful actions by noticing anomalies in network data. In contrast to signature-based methods, ML-based NIDS can learn from past data to identify both known and unknown threats, including zero-day assaults. Logistic Regression, Decision Trees, Random Forests, and Support Vector Machines are some of the most important machine learning models employed. These models include supervised and unsupervised methodologies. In addition, algorithms that use deep learning, such CNNs, RNNs, and LSTM networks, may improve the accuracy with which complicated attack patterns and abnormalities can be detected [5-9].

Threat Landscape for ML-Based NIDS Attacks

In adversarial attacks on ML-based network intrusion detection systems (NIDS), subtle changes to the input data might fool the machine learning model into mistakenly classifying malicious traffic as benign. The attackers produce xˆ by intentionally perturbing the feature vector x that is provided to them. This is done in a formal way.

$$\hat{\mathbf{x}} = \mathbf{x} + \boldsymbol{\delta}, \quad \text{and} \quad f(\mathbf{x}) \neq f(\hat{\mathbf{x}})$$

Certain requirements, such as low size, feature specificity, and protocol compliance, must be met by the perturbations in order to prevent detection or detection of the perturbations.

- Minimal Perturbation: Ensure $\|\boldsymbol{\delta}\|_p \leq \epsilon$ to avoid detection.
- Feature-Specific Constraints: Unlike categorical characteristics, which cannot be changed, continuous features may be altered.
- Protocol Compliance: Common protocols used by modified traffic include TCP, UDP, and ICMP.

By resolving an optimisation issue, adversaries aim to minimise $\|\boldsymbol{\delta}\|_p$ while ensuring $\hat{\mathbf{x}} \in C$, the set of valid inputs adhering to protocol standards.

Packets, Flows, and ML-Based Pattern Recognition

In the process of analyzing network traffic, flows of packets are used. Individual packets, which are the basic building blocks of communication, are examined to look for irregularities in their headers or conduct that violates protocol standards. Flows, on the other hand, give a more thorough view since they track groups of packets that share properties such as IP addresses and transport protocols. For example, SYN flooding, incomplete handshakes, and session hijacking are all examples of more complicated attacks that may be identified using this method. Models that use machine learning look at flows in order to identify irregularities. For instance, if there are flows that are extremely high or low, this might be an indication of a distributed denial of service attack or data exfiltration. Models like long short-term memory (LSTMs) and recurrent neural networks (RNNs) perform very well in this setting. Analysis of sequential flows reveals assaults that are multi-staged or coordinated. By aggregating flows, network intrusion detection systems (NIDS) have the potential to reveal patterns that span several connections, such as port scanning or botnet activities. Behavioural profiling is an approach that enhances detection even further by analyzing regular patterns of communication and drawing attention to any alterations that may occur.

Table 1: Analysis Level Comparison

Analysis Level	Description	Useful For Detecting
Individual Packet	Examination of individual packets with an emphasis on the payload and headers.	Various paths that ultimately intersect to form a single objective.
Flows	Packet sequence having common characteristics (e.g., protocol, source/destination IP).	DDoS at a sluggish pace, incomplete handshakes, SYN floods.
Flow Groups	Several flows that all lead to the same final goal.	Interconnected assaults, invasions with several stages, port scanning.

Strengthening NIDS to Resist Attacks

The use of adversarial training, strong feature engineering, and meticulous model tuning are essential for an NIDS's defence against adversarial assaults. With adversarial training, we can identify model manipulation better. While feature engineering focusses on creating high-quality, attack-resistant features, model fine-tuning guarantees classifier flexibility and real-world optimisation [10]. By strengthening the system against

many attack vectors, integrated defences enhance performance.

1. Training with Adversaries

One typical method of protecting oneself from hostile assaults is adversarial training. This technique teaches the model to detect and classify intentionally altered inputs by training it on both benign and malicious network traffic. Samples that are hostile to the NIDS make it more resistant to manipulation. Nevertheless, it is still difficult to create realistic antagonistic settings that mimic assaults. To prevent the model from being overfit to synthetic data and losing its capacity to generalise, adversarial samples should accurately reflect real attack patterns. This might make it less effective at detecting illicit messages, whether they are completely uninterrupted or just slightly delayed. Using Monte Carlo perturbations, we avoided these difficulties and kept the protocol intact during adversarial training by focussing on variable characteristics. This strategy increased its evasion resistance by testing the model in many real adversarial instances. To balance the dataset, we oversampled minority classes using the Synthetic Minority Over-sampling Technique (SMOTE) to reflect benign and malicious traffic. We used SMOTE and adversarial training to increase model performance and generalisability. Unbalanced data did not affect our identification of under-represented attack types. Adversarial training and balancing datasets are crucial due to Metigatic Generic Attacks' network effect. NIDSs use balancing techniques like SMOTE oversampling to increase the representation of under-represented malicious traffic to counterbalance imbalanced attack patterns and make them more sensitive to rare or subtle attacks. Due to their durability and generalisability, these approaches enhance benign and malicious communication detection [11].

2. Engineering, Feature Extraction, and Selection

For ML-based NIDS to be resilient against adversarial assaults, engineering, feature extraction, and selection are of utmost importance. The system may concentrate on its most important features, making it simpler to comprehend and less vulnerable to attacks, by streamlining these operations. Robust characteristics withstand hostile manipulation and make vulnerabilities more difficult to exploit.

Feature Extraction and Selection: Network traffic data must be feature extracted before ML models may handle unstructured data. Important information includes packet sizes, timing, protocol kinds, and header elements. Also relevant are TCP, UDP, and ICMP vulnerabilities. SYN-ACK sequences and TCP retransmission rates may suggest tampering. Model characteristics like protocol measurements, anomaly indicators, and traffic flow statistics are selected via feature selection. This boosts detection efficiency, generalises across attack kinds and network circumstances, and reduces unnecessary data sensitivity.

Feature Engineering for Security: In addition to extraction and selection, feature engineering improves the model's ability to recognise hostile attacks by creating domain-specific features. Remember that attackers may modify changeable characteristics like packet size and timing without breaking protocols, but they cannot change immutable features like protocol flags without alerts or disruptions. Protocol limitations reduce adversarial changing feature perturbations. Excessive packet size or flow rate variations may break connections and identify attacks. Attackers are more likely to be caught using bidirectional protocols like TCP because they must change both sides of the discussion without breaking the connection.

Our strategy emphasises protocol-aware feature engineering to honour feature interdependence. Unrealistic

traffic patterns might result by perturbing TCP flags without protocol standards. Protocol-specific information improves detection accuracy and reduces false positives. By aggregating traffic data over time frames or segments, feature engineering helps the model to detect minor abnormalities that single-packet studies miss. Monitoring statistical aspects like variance and mean of network flows over time might reveal persistent assaults, anomalies with a slow advancement rate, and multi-stage intrusions. NIDS protocol-aware feature refinement aims to maintain effectiveness against a variety of hostile techniques.

3. Building Models and Learning in Groups

In order to construct our models, we make use of a mix of deep learning models (LSTM and MLP) and more traditional models (Random Forest and Logistic Regression). Instead of relying on the flaws of just one model, the ensemble technique makes advantage of the strengths of several models in order to make the system more robust. To ensure the Network Intrusion Detection System's (NIDS) performance in real-world settings, it is important to test each model with both benign and malicious traffic.

Evaluation on Adversarially Imbalanced Datasets

For the purpose of evaluating the framework's resistance to Metigatic Generic Attacks, we used adversarially imbalanced datasets such as NSL-KDD and UNSW-NB15. By modelling instances in which particular attack types are under-represented as a consequence of traffic manipulation, these datasets accurately reflect the real imbalances that exist inside networks. Evaluations of accuracy, precision, and memory were carried out separately for both neutral and hostile imbalanced traffic. In order to evaluate the usefulness of the framework in terms of reducing imbalances and enhancing detection performance, it was necessary to conduct an analysis of the sequential inclusion of defensive measures. These measures included adversarial training, balancing, feature engineering, and model fine-tuning.

Table 2. Performance Comparison on Adversarially Imbalanced Datasets (Before and After Defensive Techniques)

Dataset	Attack Type Imbalance (%)	Baseline Detection Accuracy	After Adversarial Training + SMOTE
NSL-KDD	DoS: 60%, Probe: 25%, U2R: 5%, R2L: 10%	0.588 (Normal Recall) / 0.050 (Adversarial Recall)	0.832 / 0.644
UNSW-NB15	Fuzzers: 50%, Backdoor: 15%, DoS: 20%, Exploits: 15%	0.957 / 0.064	0.972 / 0.512

RESULT

An evaluation of the suggested defensive system was carried out with the assistance of two well-known NIDS benchmark datasets, namely NSL-KDD [12] and UNSW-NB15 [13]. In addition to addressing issues with data imbalance and duplicate entries, the NSL-KDD dataset, which is an upgraded version of the KDD Cup 1999 dataset, incorporates attack types such as DoS, Probe, U2R, and RML. To capture modern network traffic, on the other hand, UNSW-NB15 makes use of a 49-feature evaluation benchmark that was developed by IXIA PerfectStorm. This benchmark takes into account risks like as fuzzers and backdoors into account. The framework's parameters were evaluated using a variety of techniques, including model fine-tuning, adversarial training, feature preprocessing, dataset balancing, and feature engineering, to name a few. A combination of normal and adversarial datasets were used throughout the performance testing process. More specifically, the latter comprised communications that had been purposefully altered in order to evade detection.

Evaluation Metrics

Each model was evaluated using the following metrics:

Accuracy: The rate of properly identified cases

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: The percentage of correct forecasts to the total number of correct guesses.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall: True positive rate as a percentage of total positives.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Classifier Ensembles and the Process of Enhancement

The Deep Learning Ensemble and the Traditional Classifier Ensemble are the two classifier ensembles that we assess. Starting with a baseline and then adding more defensive techniques, the assessment process is cumulative. Here is how the process of improvement unfolds:

- **Baseline**: First assessment of the model without protective measures is required.
- **Adversarial Training & Balancing:** Applying adversarial training and balance (e.g., SMOTE).
- **Feature Engineering & Preprocessing:** The incorporation of preprocessing and feature engineering methods.
- **Model Fine-Tuning:** Performing last-ditch model optimisation for peak functioning.

The NSL-KDD and UNSW-NB15 datasets are used to assess each classifier ensemble, and results are provided for each. The findings are for both the conventional testing and hostile testing scenarios. The former uses unmanipulated traffic, while the latter uses disrupted traffic.

Results on UNSW-NB15 and NSL-KDD Datasets

The results of the Deep Learning Ensemble (DC) and the Traditional Classifier Ensemble (TC) for the NSL-KDD and UNSW-NB15 datasets, respectively, are shown in Tables II and III. At each phase of the cumulative improvement process, accuracy, precision, and recall are employed to deliver results for normal and hostile testing datasets [14]. Adversarial training settings raised erroneous positives but decreased false negatives, improving robustness to adversarial inputs. Balancing approaches like SMOTE lowered false positive rates, notably in high-dimensional datasets like UNSW-NB15. Compared to unaffected datasets, adversarial training fared better. SMOTE immediately improved recall and reduced false positives. Scaling, feature interaction, and advanced feature engineering enhanced model performance and generalisation, especially for intricate classifiers.

By adjusting for subtle changes, fine-tuning deep learning and tree-based models performed better. These techniques substantially enhanced accuracy and memory in all setups. The effectiveness of the architecture against Metigatic Generic Attacks, which disturb traffic patterns, is shown by the fact that adversarial recall improves across all configurations. Both UNSW-NB15's Backdoor and NSL-KDD's U2R had low detection rates when tested under baseline settings. There was an improvement in detection, particularly for uncommon or subtle attack types, after using adversarial training, dataset balance (SMOTE), feature engineering, and model fine-tuning. This demonstrates that the framework reduces the imbalance in traffic caused by generic assaults. Defensive methods enhance both benign and malicious traffic detection, as shown in Figure 2 [15]. The graph demonstrates that defensive metigatics reduce the disparity between adversarial and regular accuracy. The design effectively counters the traffic imbalance caused by generic assaults, as shown here.

Table 3. NSL-KDD – TC Ensemble

Stage	Normal Accuracy	Adversarial Accuracy
Baseline	0.713	0.396
Adversarial Training + SMOTE	0.865	0.742
Feature Engineering + Preprocessing	0.905	0.822
Model Fine-Tuning	0.931	0.920

Table 4. UNSW-NB15 – TC Ensemble

Stage	Normal Accuracy	Adversarial Accuracy
Baseline	0.750	0.254
Adversarial Training + SMOTE	0.860	0.684
Feature Engineering + Preprocessing	0.913	0.792
Model Fine-Tuning	0.944	0.942

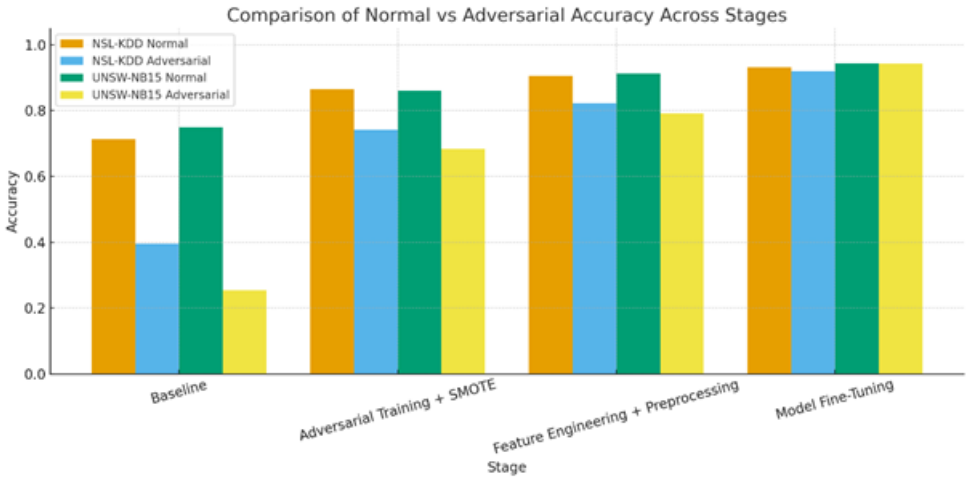

Figure 2. Comparison Of Normal Vs. Adversarial Accuracy Across Stages

Table 5: NSL-KDD Dataset Performance

Classifier Ensemble	Improvement Stage	Normal Accuracy	Normal Precision	Normal Recall	Adversarial Accuracy	Adversarial Precision	Adversarial Recall
TC Ensemble	Initial	0.713	0.887	0.588	0.396	0.399	0.050

	+ Balancing & Adversarial Training	0.865	0.915	0.832	0.742	0.792	0.644
	+ Preprocessing & Feature Engineering	0.905	0.945	0.890	0.822	0.876	0.759
	+ Fine-Tuning the Model	0.931	0.967	0.915	0.920	0.952	0.910
DL Ensemble	Initial	0.746	0.863	0.684	0.431	0.592	0.157
	+ Balancing & Adversarial Training	0.894	0.920	0.853	0.815	0.859	0.722
	+ Preprocessing & Feature Engineering	0.931	0.946	0.901	0.872	0.903	0.797
	+ Fine-Tuning the Model	0.960	0.953	0.982	0.940	0.933	0.969

Table 6: UNSW-NB15 Dataset performance

Classifier Ensemble	Improvement Stage	Normal Accuracy	Normal Precision	Normal Recall	Adversarial Accuracy	Adversarial Precision	Adversarial Recall
TC Ensemble	Initial	0.750	0.703	0.957	0.254	0.137	0.064

	+ Adversarial Training & Balancing	0.860	0.813	0.972	0.684	0.579	0.512
	+ Preprocessing & Feature Engineering	0.913	0.871	0.990	0.792	0.679	0.611
	+ Fine-Tuning the Model	0.944	0.912	0.996	0.942	0.916	0.987
DL Ensemble	Initial	0.821	0.966	0.722	0.397	0.844	0.150
	+ Balancing & Adversarial Training	0.885	0.939	0.805	0.743	0.877	0.689
	+ Preprocessing & Feature Engineering	0.932	0.967	0.893	0.874	0.909	0.786
	+ Fine-Tuning the Model	0.977	0.993	0.969	0.963	0.981	0.956

The results of this study demonstrate that our approach is able to effectively mitigate the repercussions of traffic imbalance that is generated by Metigatic Generic Attacks [16]. This is accomplished by enhanced detection of attack types that are under-represented and subtle across both datasets.

- **Vulnerability of ML-Based NIDS:** Because Metigatic Generic assaults disrupt networks and produce traffic imbalances, baseline ML models have a hard time detecting nuanced and under-represented assaults.
- **Adversarial Training & Dataset Balancing:** Combining adversarial training with SMOTE greatly enhances attack detection accuracy while decreasing false negatives.
- **Feature Engineering & Preprocessing:** Model resilience is improved by the use of protocol-aware feature engineering and thorough preprocessing, which centre on realistic and attack-resistant traffic patterns.

- **Model Fine-Tuning & Ensemble Learning:** Accuracy, precision, and recall for both benign and malicious traffic may be enhanced by fine-tuning combinations of classic and deep learning models.
- **Robustness Against Traffic Imbalance:** Reliable detection of subtle and infrequent assaults is made possible by the proposed Metigatic defensive architecture, which also remains scalable for dynamic network settings. It successfully mitigates attack-induced traffic imbalance.

CONCLUSION

To make machine learning-based network intrusion detection systems (ML-based NIDSs) more resistant to broad attacks that induce imbalances in network traffic, we designed a metigatic defensive architecture as part of this study. The framework use a mix of adversarial training, dataset balance, feature preprocessing, feature engineering, and model fine-tuning to reduce the impact of under-represented attack patterns and adversarial manipulations. By taking a holistic view, the system is able to mitigate the effects of these kinds of assaults. We employed actual adversarial traffic samples that were built with feature interdependencies and protocol compliance in order to test and develop the models. These samples formed the basis for our testing and improvement efforts. The experimental results on the UNSW-NB15 and NSL-KDD datasets demonstrated significant improvements in recall, precision, and accuracy. These improvements were specifically seen for traffic that was imbalanced and adversarially disrupted. In order to address problems that are brought about by traffic imbalance and new generic threats, our findings demonstrate that the system is capable of accurately identifying non-standard and nuanced forms of assault. When used to ML-based NIDS installations in the actual world, the approach that was proposed provides a defence that is both practical and scalable, in addition to ensuring increased security and dependability in network situations that are constantly altering.

References

1. Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. 2010 IEEE Symposium on Security and Privacy, 305–316.
2. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321–357.
3. Zhang, J., Zulkernine, M., & Haque, A. (2019). Random-forest-based network intrusion detection systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(5), 649–659.
4. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41–50
5. S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. IEEE Transactions on Systems, Man, and Cybernetics, 21(3), pages 660–674, 1991.
6. S. Zhang, C. Li, and J. Jiang. A Random Forest-Based Anomaly Detection System for Network Intrusion Detection. Security and Communication Networks, vol. 2018, Article ID 9671863, 2018.

7. R. Vinayakumar, K. P. Soman, and P. Poornachandran. Applying Convolutional Neural Network for Network Intrusion Detection. Journal of Network and Computer Applications, 2017.

8. C. Yin, Y. Zhu, J. Fei, and X. He. A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 2017.

9. S. Roy, W. H. Cheung, and A. Roy. LSTM networks for intrusion detection. In IEEE International Conference on Computer and Information Technology, 2017

10. G Emile S, Mbungu Kala, "Critical Role of Cyber Security in Global Economy", Open Journal of Safety Science and Technology, Vol. 13, pp. 231-248, 2023. doi: 10.4236/ojsst.2023.134012.

11. Von Solms, Rossouw, and Johan Van Niekerk. "From information security to cyber security." computers & security, Vol. 38, pp. 97-102, 2013.

12. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pages 1–6. IEEE, 2009.

13. Nour Moustafa and Jill 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), pages 1–6. IEEE, 2015.

14. J W Goodell and S. Corbet, Commodity market exposure to energy firm distress: Evidence from the colonial pipeline ransomware attack,''Finance Res. Lett., vol. 51, Jan. 2023, Art. no. 103329

15. R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar winds hack: In-depth analysis and countermeasures," in Proc. 12th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT), Jul. 2021, pp. 1–7.

16. Cobalt https://www.cobalt.io/blog/biggest-cybersecurity-attacks-inhistory