National College of Ireland

An Artificial Intelligence aided simulation testing framework for network intrusion detection in different operating systems

MSc Research Project
Cybersecurity

Srilakshmi Pamarthi

Student ID: x23142294

National College of

Ireland

Supervisor: JAWAD SALAHUDDIN

National College of Ireland
Project Submission Sheet School of Computing

Student Name:	Srilakshmi Pamarthi
Student ID:	X23142294
Programme:	Cybersecurity
Year:	2024
Module:	MSc Research Project
Supervisor:	Jawad Salahuddin
Submission Due Date:	12/08/2024
Project Title:	An Artificial Intelligence aided simulation testing framework for network intrusion detection in different operating systems
Word Count:	6932
Page Count:	26

Signature:	Srilakshmi Pamarthi
Date:	12th August 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

An Artificial Intelligence aided simulation testing framework for network intrusion detection in different operating systems

Srilakshmi Pamarthi
x23142294

National College of Ireland

Abstract

This paper examines how to design and develop a machine learning-based Intrusion Detection System (IDS) for enhancing the level of security by timely s detection of the intrusions. Because of the rapid growth of the number of connected objects, the relied-on approaches to security are insufficient, thus exacerbating the vulnerabilities. The architecture that we want to develop adapts and implements AI technology in the process of simulation testing. under the guidance of which many features are complied with the regulation of the GDPR. For the assessment of various machine learning algorithms using the NSL-KDD dataset, it was discovered that both Random Forest and Decision Tree algorithms demonstrated better results in the detection of intrusions and vulnerabilities. The versatility in the framework is that it is capable of learning from new threats as they form thus making it dynamic in nature. This leads to minimizing the extent of use of manpower and consequently lowering of operating costs.

Keywords: Simulation Testing, Machine Learning, Cyber Security, IT solution, Compliance

1: Introduction

An Intrusion Detection System (IDS) (**Saied, M. , Guirguis, S. and Madbouly, M. , 2024**) is a security tool that is used to constantly scan and identify harassing activities or violations of security policies on a computer network or in certain systems. In this sense, the system assists in the detection of unauthorized attempts, potential threats, and suspicious activities, while calling administrators' attention to them to take necessary actions. The reliable system that can be utilized to maintain network security and protect the critical information from cyber threats is Intrusion Detection System (IDS).

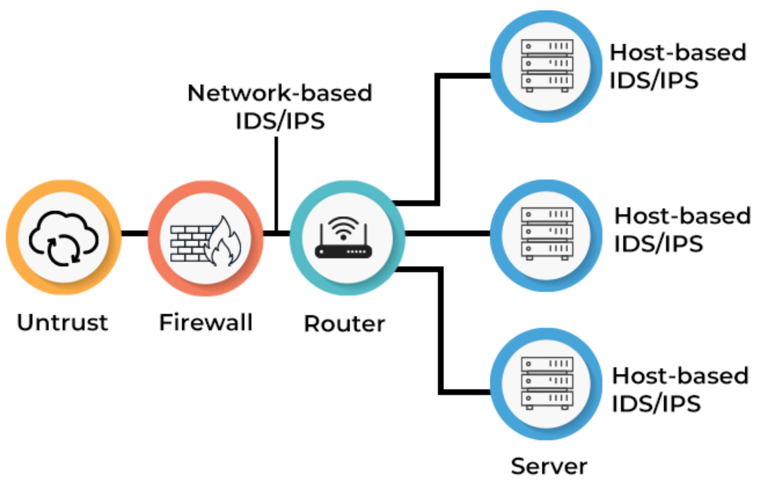


Figure 1: An IDS/IPS on an enterprise network (**Source: Spiceworks**)

1.1 **Motivation**

An Intrusion Detection System (IDS) monitors network traffic for anomalous behaviour and generates alarms when such behaviour is detected. An Intrusion Detection System (IDS) primarily performs the tasks of detecting and reporting anomalies. However, certain IDSs can respond to hostile activities or abnormal network traffic. This article will comprehensively cover all aspects of the Intrusion Detection System. An Intrusion Detection System (IDS) is a security mechanism designed to detect and respond to unauthorized access attempts or malicious activities within a computer network or system. These systems can monitor network traffic for malicious activities and promptly notifies of any detected incidents. Network or system monitoring software is designed to detect and identify harmful activity or policy breaches (**Nebbione, G. and Calzarossa, M.C., 2023**). Every instance of unlawful activity or violation is often documented either in a centralized manner utilizing a Security Information and Event Management (SIEM) system or reported to an administrative entity. An Intrusion Detection System (IDS) is responsible for monitoring a computer network or system to detect any harmful activity. Its main purpose is to safeguard the network against unwanted access by users, including potential insiders. The objective of the intrusion detector learning challenge is to construct a prediction model, specifically a classifier, that can effectively differentiate between 'bad connections' (intrusions/attacks) and 'good (normal) connections'. Based on this a network based using the AI (Artificial Intelligence) can be made that can be able to handle the vulnerability attacks.

1.2 Problem Statement

In today's digital world, the number of linked devices has grown rapidly, surpassing the global population. This rise is similar to the significant changes brought about by the Industrial Revolution. The rapid increase in the number of instances has brought up considerable security difficulties, especially in the area of safeguarding data. Out all these risks, phishing, ransomware, and system misconfiguration are the most widespread, with phishing being the primary issue worldwide. Conventional security procedures (**Chaudhari, A., Gohil, B. and Rao, U.P., 2024**), such simulation testing, have demonstrated their effectiveness in detecting and reducing vulnerabilities. Nevertheless, the manual implementation of these tests, particularly for wireless networks, places significant budgetary and resource constraints on enterprsises.

Due to the intricate nature and large amount of data produced by wireless networks, it is not feasible for simulation testers to consistently monitor and analyse traffic manually. This constraint not only burdens the resources of the firm but also exposes systems to advanced assaults, such as SQL injections and cross-site scripting (**Ghanem, M.C., Chen, T.M. and Nepomuceno, E.G., 2023**). Additionally, it puts private information and intranet server addresses at risk of being compromised.

1.3 Business Outcome

Integrating a machine learning-powered framework for identifying vulnerabilities and conducting simulation testing can greatly increase the cybersecurity defences of enterprises in many sectors (**www. akamai.com/solutions/security**). Such a solution gives full security and flexibility by expanding the ability of learning threats on any OS including the Windows OS, Linux OS, MAC OS as well as Ubuntu. Important business results include (**https://www.geeksforgeeks.org/Intrusion Detection System**):

1.4 **Research Question**

Based on the above discussions following are the research questions that we are focussed upon,

RQ1: Can Machine learning-based AI system be designed that will be agnostic in detecting and preventing the network intrusion detection in different operating systems?

RQ2: Can an AI based solution mitigate the vulnerability attacks by complying with the GDPR?

1.5 **Objective**:

The objective of this work is, therefore, to develop a suitable simulation testing framework for automation of intrusion and vulnerability detection while integrating continuous learning capability to network systems. Thus, by informulating AI models into the central part of the framework, the project aims to enhance the efficiency of simulation testing procedures and their adaptability to new security threats. Majority of the works towards the simulation testing has utilized some of the best practices of the use of KALI Linux and SQL injection for the prohibiting of the attacks together with the incorporation of SQL injections. In this research, emphasis will be laid on demonstrating how integration of AI in automatic simulation testing framework can aid in the security of the system. The NSL-KDD dataset shall be used for evaluation testing for training and finalising the system. Therefore, we put forward an AI-Penetration-Framework, which is a manually crafted auto ML that re-estimates hyperparameters from the training and produces the most accurate trained model. We found some of the machine learning models being used, here our target will be an enhanced response system which not only focusses on the improvement of the intrusion detection but also minimises the space without continuously deploying the solution but only when required.

Thesis Structure

In the next chapter we will discuss in details about the different researches that has been done and try to find those gaps and improvement areas that can be worked upon. In the chapter on methodology, we will discuss in details about the different techniques and methods that can be applied in making the proposed solution. In the chapter on implementation we will discuss in details about the implemented solution and the way it has been implemented. In the chapter on results and analysis we will discuss in details about the analysis and use cases.

2 Literature Review:

Security of operating systems and network systems has become quite important in the contemporary world where every computer is connected with another in a network for the security of significant infrastructures and important data. The main security types that are no longer useful when it comes to handling new forms of cyber threats and their attack vectors are as follows. One potential solution to this growing threat is combining DL and AI (**Chaudhari, A. , Gohil, B. and Rao, U. P. , 2024**); which can boost the network security and enhance the ability to identify cyber threats.

2.1 AI Techniques in Intrusion Detection Systems

In a research by (**H. Sadia et al. ,2024**) proposed a revealed NIDS to safeguard the Wi-Fi based WSNs from cyber attacks like impersonation attacks, flooding attack, and injection attacks. They brought down the number of features originally used from 154 to 13 core features, and they used a Convolutional Neural Network (CNN) for multiclass intrusion detection. The CNN model of analysis gave a high recognition rate of 97% and a loss measurement of 0. True positive: 14, false positive: low. This work considerably enhances the efficiency of IDSs that overprompt WSNs' security against agenda cyber threats.

In a research by (**Jie, W. , Chen, Q. , Wang, J. , Koe, A. S. V. , Li, J. , Huang, P. , Wu, Y. and Wang, Y. , 2023**) in the identified shortcomings in the current approaches through empirical findings and the flexibility of the feature selection. This framework combines strong whitebox knowledge and uses supervised multitrajectory tasks in the view of static analysis. Their approach consists of such stages as feature selection, dimensions normalization, features joining, models' training, and making of decisions. For each task, the researchers use self-attentive bi-LSTMs, (text) CNNs, and RFs to attain the joint multimodal feature representations. These include code and graph embeddings both at intramodal or single level and intermodal or cross level. Analyzing 101,082 functions from the SmartEmbed dataset proves that their approach is better as their detector gets a detection rate of up to 99%. 71%, surpassing existing schemes. Through frameworks, checklists, and best practices within this publication, the developer receives practical and flexible solutions to improve the identification of threats in smart contracts.

In a research (**Siva Shankar, S. , Hung, B. T. , Prashant Chakrabarti, Tito Chakrabarti, & Gopal Parasa, 2024**) proved a novel optimization-based deep learning technique assisted by artificial intelligence with surveillance of intrusion threats in network system. The authors successfully approach the issue of quantifying the trends of the malicious detection techniques in the IDS market with the introduction of the optimised Artificial Intelligence approach to IDs. The goal of this approach was to effectively detect intrusions in the networks that are experiencing increased vulnerability to cyber threats mainly due to problems of resource limitations and network heterogeneity. The authors presented a scenario of the suggested strategy utilizing the datasets of NSL-KDD and UNSW-NB15 with the help of experiments on the Python platform. His method of operation when tested on the label data exhibited a detection rate of 99% higher than the other GEO-SMPIF solutions. 78% and 99. 70% , an accuracy rate that was recorded at ninety-nine point nine percent. 99% and 99. 97%.

(**Abuali, K. M. , Nissirat, L. and Al-Samawi, A. , 2023**) proposed support vector machine frequently called deep learning to categorize data to find intrusion incident on social media networks. They used the CSE-CIC-IDS 2018 dataset which required to be preprocessed for the training phase to be carried out. To test the proposed model, it was applied on a small sample data of 100,000 instances. Extraordinary outcome of the study was noted: the model yield 1.00 of the accuracy, sensitivity, positive predictivity, exactness, and F1-measure, whereas the false-positive recall was equal to zero.

Another new methodological framework for packet-based NIDS, framework by (**Ghadermazi, J. , Shah, A. and Bastian, N. D. , 2024**) also considers the dependency between temporal elements of packets and is capable to perform efficient analysis of the payload and the header data. Their system employs intrusion detection model based on the CNN, where the system translates the sequences of packets into the two-dimensional images and then searches for any sign of aggressive activity. The authors applied this technique on publicly available huge datasets and proved that it gets the detection rates from 97%. from 7 percent to 99 percent across the different kinds of attacks. Along with enhancing the performance for intrusion detection using networks and mitigating crucial limitations of packet based NIDS, their approach also provided fair resistance against adversarial cases..

2.2 Network Intrusion Detection Systems (NIDS) Enhancements

In a research by (**Chen, J. L. , Chen, Z. Z. , Chang, Y. S. , Li, C. I. , Kao, T. I. , Lin, Y. T. , Xiao, Y. Y. and Qiu, J. F. , 2023, February**) put forward an intrusion detection method using AI for the ITRI AI BOX information security purpose. The packets are analyzed by the AI BOX based on the AI algorithms that identify threats or unusual traffic in the network. After that, the approach changes or isolates the abnormal or malicious network data transmission behavior to ensure information security. AI models for anomaly detection help the system to either enable or restrict the flow of data therefore is used to secure a device. The future developments of the approach endeavour to detect and mitigate the packets in both IT and OT environments thus ensuring safe data processing and conversions in various networks. T he experimental assessment of the intrusion detection style demonstrated its effectiveness, insofar as it ascertain to a publicly available data set with a 99% rate of accuracy. The environment that contains machine learning models, packet sniffing functionality, and certain configurations of the operating system under the name of AI BOX has been built and experimented in the Yocto Project environment. This demonstration proves that it can protect smart factories or hospitals from abnormal traffic attacks and prevent the occurrence of system paralyzing or extortion like events.

Thus, to enhance the IDS's efficiency (**Mohammad, R. , Saeed, F. , Almazroi, A. A. , Alsubaei, F. S. and Almazroi, A. A. , 2024**) suggested a design that utilizes deep learning architectures and data augmentation methods. Specifically, to counter the limitations of existing intrusion detection techniques based on machine learning, they evaluated their models on four datasets: UNSW-NB15, 5G-NIDD, FLNET2023, CIC-IDS-2017. There are conducted the experiments that finally confirmed that the CNN, based idea of a very simple architecture of the classifier, can detect the network attacks with very high accuracy, where on the additional CIC-IDS-2017 dataset, achieving the maximum of 91 % of accuracy.

To enhance IDS performances and reduce the occurrences of false alarms (**More, S. , Idrissi, M. , Mahmoud, H. and Asyhari, A. T. , 2024**) proposed developing machine learning models. On the UNSW-NB15 network traffic dataset, they used logistic regression, support vector machine, decision tree, and random forest algorithms. Thus, the proposed model terms of accuracy, speed, and percentage of correctly classified samples are 98. 63%, and F-measure of 97. 80%, Random Forest is considered to be the most accurate model among the considered ones. To sum it up, it appears that the Random Forest model is good at identifying cyberattacks and can improve IDS systems.

2.3 AI Techniques in Simulation Testing Frameworks

The LSTM-RNN based simulation testing framework named as Long Short-Terms Memory Recurrent Neural Network Enabled Vulnerability Identification (LSTM-EVI) is proposed to mitigate the cybersecurity issues of IoT by (**N. Koroniotis, N. Moustafa, B. Turnbull, F. Schiliro, P. Gauravaram, and H. Janicke, 2021**). Notably, the proposed framework by the authors achieved a higher score compared to four other approaches, with scanner attacks' detection accuracy reaching only slightly below 99%. Observations presented in this paper illustrate on how deep learning models can enhance the discovery of vulnerabilities and the generation of simulation tests for complex IoT systems.

In (**Ghanem, M. C. and Chen, T. M. , 2019**) the researcher Integrated the Intelligent Automated Simulation Testing System (IAPTS) that incorporates machine learning techniques namely Reinforcement learning (RL) to the process of simulation testing (PT). They synchronise with other industrial PT structures to gather data, assimilate advanced incidents, and recreate tests in subsequent conditions. Studying PT environments and tasks as partially observable Markov decision problem (POMDP) and its solution using POMDP-solver, IAPTS can reduce the human involvement and increase the effectiveness and credibility of PT.

(**Confido, A. , Ntagiou, E. V. and Wallum, M. , 2022, March**), For the purpose of improving PenBox's testing procedure the research was focused on deployment of the Reinforcement Learning (RL), particularly, Q-learning paradigm. Replacing the simulation testing with a Q-learning problem formulation enables the system to learn a policy that will maximize the rewards in a trial-by-trial fashion. Optimal attack paths were defined through the results with help of which it was possible to identify that this approach could improve simulation test processes.

Intelligent Simulation Testing derived from Generative Adversarial Imitation Learning called as GAIL-PT (**Chen, J. , Hu, S. , Zheng, H. , Xing, C. and Zhang, G. , 2023**) was proposed as a solution to the issues associated with RL/DRL-based testing. wages were used the discriminator in the GAIL-PT framework to establish the expert knowledge bases which in turn were fed to the training process. The states and actions were acquired from the RL/DRL model successful exploitation of states and actions. Compared with the current techniques like DeepExploit and Q-learning, GAIL-PT accuracy is higher in the simulated network conditions and the actual target hosts, which proved GAIL-PT as one of the apex frameworks for using RL/DRL-based simulation testing approaches..

2.4 Summary Table

Table 1: A summary review of all the research reviewed and analysed

Paper	**Author**	**Model/algorithms used**	**Dataset**	**Results**
Intrusion Detection System for Wireless Sensor Networks: A Machine Learning Based Approach	Halima Sadia, Saima Farhan, Yasin Ul Haq, Rabia Sana, Tariq Mahmood, Saeed Ali Omer Bahaj, Amjad Rehman(2024)	CNN	AWID)	The CNN model produced an excellent 97% accuracy, a loss measure of 0.14, and a low false alarm rate.
A novel extended multimodal AI framework towards vulnerability	Jie, W., Chen, Q., Wang, J., Koe, A.S.V., Li, J., Huang, P., Wu, Y. and Wang, Y., 2023	Self-attentive Bi-LSTM, TextCNN (Convolutional Neural Network for text), Random Forest (RF)	**SmartEmbed dataset** containing 101,082 functions for evaluation.	Achieved Detection Performance: Up to 99.71%

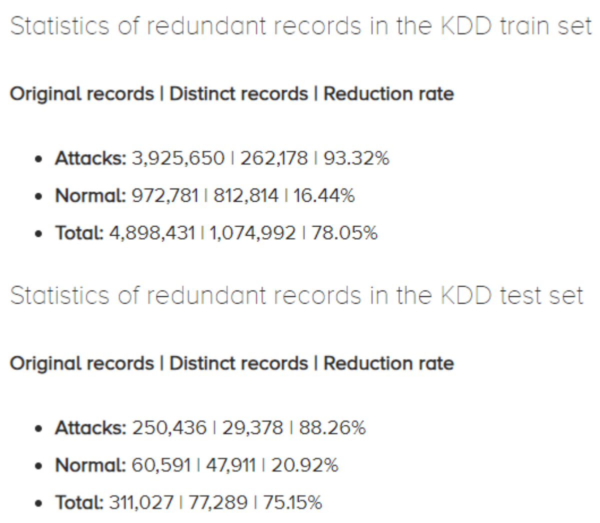
detection in smart contracts				
A novel optimization based deep learning with artificial intelligence approach to detect intrusion attack in network system	Siva Shankar, S., Hung, B.T., Chakrabarti, P., Chakrabarti, T. and Parasa, G., 2024.	Corporate Hierarchy Optimisation (CHO), Golden Eagle Optimisation (GEO) algorithm, Multi-layer Perceptron (MLP) interfaced fuzzy system	NSL-KDD and UNSW-NB15	NSL-KDD Dataset: Detection Rate: 99.78%, Accuracy Rate: 99.99%, False Alarm Rate: 0.04 UNSW-NB15 Dataset: Detection Rate: 99.70%, Accuracy Rate: 99.97%, False Alarm Rate: 0.065
Advancing Network Security with AI: SVM-Based Deep Learning for Intrusion Detection	Abuali, K.M., Nissirat, L. and Al-Samawi, A., 2023	**Support Vector Machine (SVM)**-based deep learning system	CSE-CIC-IDS 2018 dataset	Accuracy: 100% True-positive recall: 100% Precision: 100% Specificity: 100% F-score: 100% False-positive recall: 0%
Towards real-time network intrusion detection with image-based sequential packets representation	Ghadermazi, J., Shah, A. and Bastian, N.D., 2024.	CNN	publicly available big datasets	**Detection Rates:** 97.7% to 99% across various attack types

3: Methodology

As we progress deeper into the digital era, protecting networked systems has emerged as a top priority for companies around. The reason behind this is the growing interconnection of networked systems. With the evolution of cyberattacks—which can encompass everything from efforts to obtain unauthorised access to complex malware invasions—it is more important than ever to safeguard the security and integrity of computer networks. There are several ways in which these hacks can target computer networks. While there is some success with more conventional approaches to network security, these methods often miss rapidly evolving threats and vulnerabilities. This is so even if these strategies do tend to work. Network security could be jeopardised because of this.

3.1 Dataset Description

The NSL-KDD dataset offers many advantages compared to the original KDD dataset, The train set does not contain redundant data, ensuring that the classifiers are not biased towards more frequent entries. The suggested test sets do not contain any duplicate records, ensuring that the performance of the learners is not influenced by approaches that have higher detection rates on frequent records. Source: https://www.unb.ca/cic/datasets/nsl.html. The quantity of chosen records from each degree of difficulty is inversely related to the proportion of records in the original KDD dataset. Consequently, the classification rates of different machine learning methods exhibit a broader range, thereby enhancing the efficiency of accurately evaluating various learning approaches. The train and test sets include a fair amount of records, allowing for the experiments to be conducted on the whole set without the necessity of randomly selecting a tiny section. As a result, the assessment outcomes of many research studies will be consistent and capable of being compared.


Statistics of redundant records in the KDD train set

Original records | Distinct records | Reduction rate

- **Attacks:** 3,925,650 | 262,178 | 93.32%
- **Normal:** 972,781 | 812,814 | 16.44%
- **Total:** 4,898,431 | 1,074,992 | 78.05%

Statistics of redundant records in the KDD test set

Original records | Distinct records | Reduction rate

- **Attacks:** 250,436 | 29,378 | 88.26%
- **Normal:** 60,591 | 47,911 | 20.92%
- **Total:** 311,027 | 77,289 | 75.15%

Figure 2: Information of the the data types in KDD dataset (**Source: unba.ca**)

The detailed information of the dataset present in the dataset is as below,

RangeIndex: 125973 entries, 0 to 125972
Data columns (total 43 columns):
Column   Non-Null Count Dtype
--- ------   -------------- -----
0 duration   125973 non-null int64
1 protocol_type   125973 non-null object
2 service   125973 non-null object
3 flag   125973 non-null object
4 src_bytes   125973 non-null int64
5 dst_bytes   125973 non-null int64
6 land   125973 non-null int64
7 wrong_fragment   125973 non-null int64
8 urgent   125973 non-null int64
9 hot   125973 non-null int64
10 num_failed_logins   125973 non-null int64
11 logged_in   125973 non-null int64
12 num_compromised   125973 non-null int64
13 root_shell   125973 non-null int64
14 su_attempted   125973 non-null int64
15 num_root   125973 non-null int64
16 num_file_creations   125973 non-null int64
17 num_shells   125973 non-null int64
18 num_access_files   125973 non-null int64
19 num_outbound_cmds   125973 non-null int64
20 is_host_login   125973 non-null int64
21 is_guest_login   125973 non-null int64
22 count   125973 non-null int64
23 srv_count   125973 non-null int64
24 serror_rate   125973 non-null float64
25 srv_serror_rate   125973 non-null float64
26 rerror_rate   125973 non-null float64
27 srv_rerror_rate   125973 non-null float64
28 same_srv_rate   125973 non-null float64

29 diff_srv_rate 125973 non-null float64
30 srv_diff_host_rate 125973 non-null float64
31 dst_host_count 125973 non-null int64
32 dst_host_srv_count 125973 non-null int64
33 dst_host_same_srv_rate 125973 non-null float64
34 dst_host_diff_srv_rate 125973 non-null float64
35 dst_host_same_src_port_rate 125973 non-null float64
36 dst_host_srv_diff_host_rate 125973 non-null float64
37 dst_host_serror_rate 125973 non-null float64
38 dst_host_srv_serror_rate 125973 non-null float64
39 dst_host_rerror_rate 125973 non-null float64
40 dst_host_srv_rerror_rate 125973 non-null float64
41 attack 125973 non-null object
42 level 125973 non-null int64
dtypes: float64(15), int64(24), object(4)
memory usage: 41.3+ MB

3.2 Machine Learning Algorithms

Through the utilization of datasets such as NSL-KDD for intrusion detection and CVE for vulnerability assessment, the project will train AI models to identify and mitigate security risks proactively. By deploying the AI-aided simulation testing framework in real-world environments and implementing monitoring and response mechanisms, the project aims to validate the effectiveness of the developed solution in enhancing system security and resilience against cyber threats.

3.2.1 Logistic Regression

Our simulation testing prediction research uses Logistic Regression (LR)(**Nick, T.G. and Campbell, K.M., 2007**), a basic machine learning approach that is easy to understand and straightforward. Based on input features, it forecasts the likelihood of a binary result, such as Suspicion or Correct. LR is a great tool for determining important elements that affect simulation testing using the AI choices since it helps to understand the relationship between the predictors and the dependent variable. In spite of its simplicity, LR's effectiveness and efficiency in binary classification problems make it a solid baseline model.

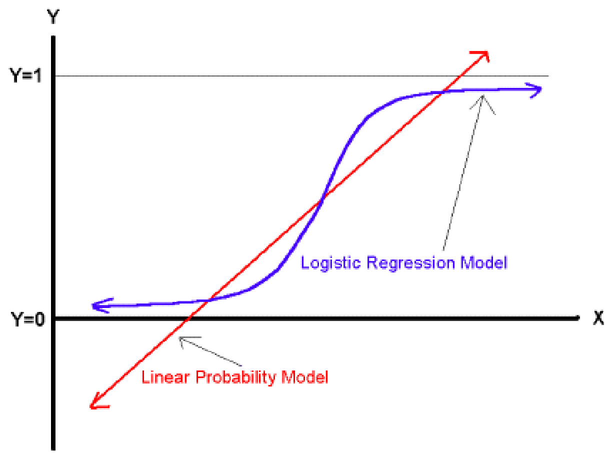


Figure 3: The S-Curve analysis for the Logistic regression. The concept behind is the logit function (**Nick, T.G. and Campbell, K.M., 2007**)

The logit function is as below,

$$f(z) = \frac{1}{1 + e^{-z}}$$

When this above is differentiated, then

$$f'(z) = f(z)(1 - f(z))$$

Making the calculations easier. Here *f(z)* is the logit function.

3.2.2 Decision Trees

Our research makes use of decision tree (DT) (**De Ville, B., 2013**) algorithms because of their interpretability and capacity to manage non-linear interactions. DTs generate a decision tree-like model by dividing the data into subsets according to the most important features. This approach is helpful for determining decision rules in simulation testing using the AI approval procedures since it is simple to visualise and comprehend. Nevertheless, pruning strategies and ensemble approaches can prevent overfitting in DTs.

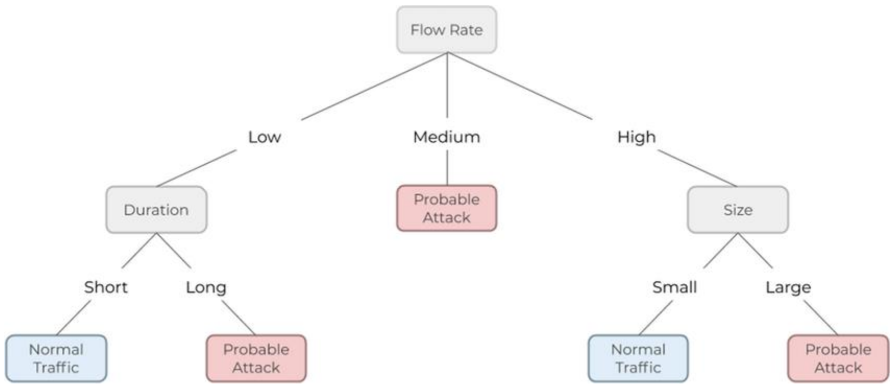


Figure 4: Illustration of working of decision trees for the vulnerable detection (**De Ville, B., 2013**) (**Source: TowardsDataScience**)

3.2.3 Random Forest

Random Forest (RF)(**Rigatti, S.J., 2017**) is an ensemble technique that generates a large number of trees during training and outputs the mode of the classes for classification problems, hence improving the predictive performance of decision trees. RF lowers the chance of overfitting by averaging the output of several trees, improving model resilience and accuracy. This algorithm handles a lot of input features well and provides excellent accuracy, making it especially useful for our simulation testing using the AI prediction task.

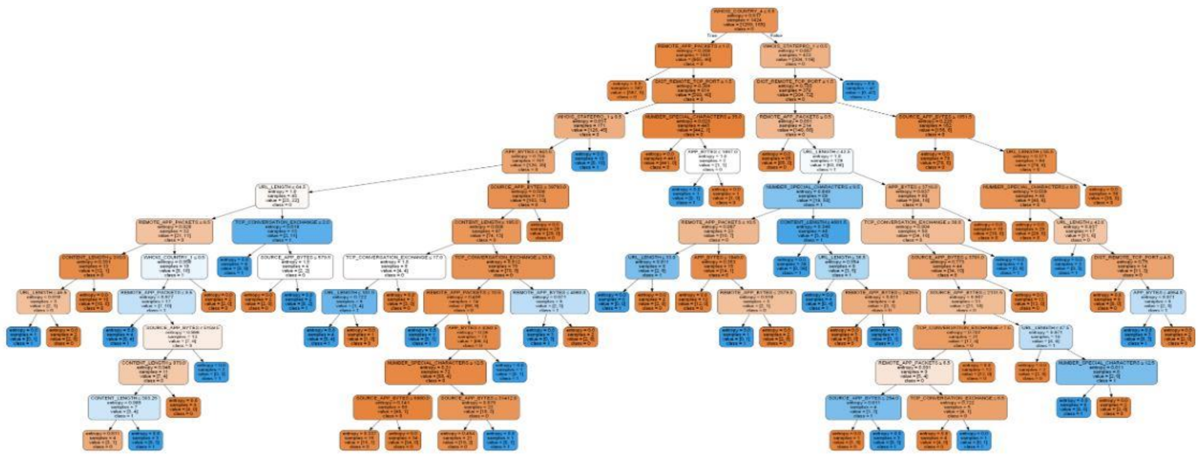


Figure 5: Random Forest, making many trees within the same population to enhance the prediction model (**Rigatti, S.J., 2017**)(**Source: Sklearn**)

3.2.4 Support Vector Machines

Our research makes use of Support Vector Machine (SVM)(**Vishwanathan, S.V.M. and Murty, M.N., 2002**) because of its robustness in classification tasks and its capacity to handle high-dimensional data. The best hyperplane that maximises the margin between the classes is found using SVM. Especially when handling non-linear connections with kernel functions, this method offers excellent generalisation capabilities and good accuracy in our project.

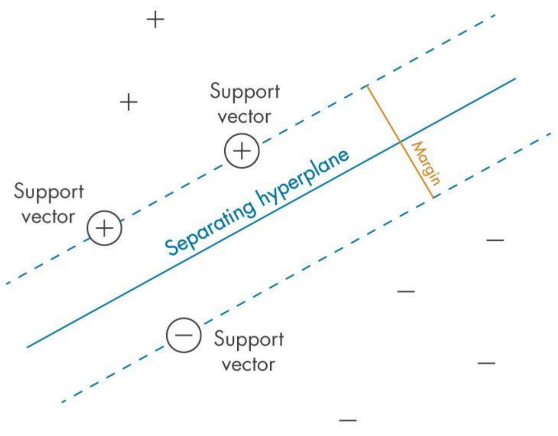


Figure 6: Analysis of the output values for finding the maximum margin of the hyperplane generated from the support vectors (**Vishwanathan, S.V.M. and Murty, M.N., 2002**)

3.2.5 Simulation Testing

An essential part of determining how to secure a network system is conducting simulation tests, sometimes called ethical hacking. Our project's overarching goal is to improve network intrusion detection across different OSes by creating an AI-aided simulation testing platform. With the help of machine learning algorithms, we were able to increase the detection and identification of vulnerabilities and incursions during our simulation testing process, which is detailed in this section along with the methodology, tools, and techniques used (**Rohit, R., Firnas, H., Abishek, M. and Dhamodaran, S., 2023**).

4: Implementation

Here we go over the steps needed to set up the project's simulation testing system, which includes two essential features: detecting intrusions using the NSL-KDD dataset and vulnerabilities using Machine Learning (ML).

4.1 Algorithm Flow

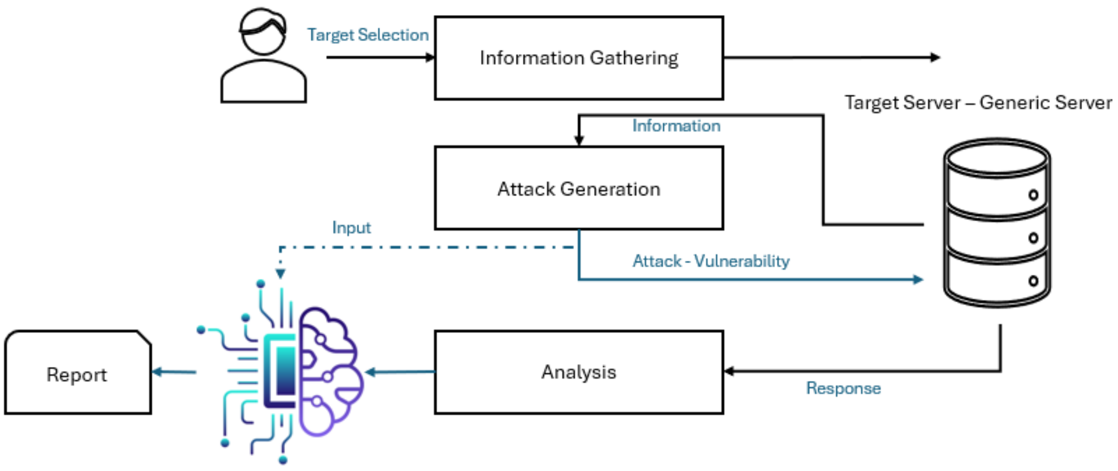


Figure 11: Algorithm Flow graph

To differentiate between legitimate network traffic and possible intrusions, the framework's network traffic analysis uses a binary classification method with the NSL-KDD dataset. An established benchmark for intrusion detection systems, the NSL-KDD dataset provides labelled data with distinct network traffic patterns classified as normal or different sorts of attacks. A rundown of the main procedures is as follows:

Algorithm Flow:

Step 1: Data Preprocessing: In order to get the NSL-KDD dataset ready for the machine learning model, it is pre-processed. Jobs such as these could be involved:

a. Filtering out extraneous data (e.g., internal network traffic).
b. Providing the selected binary classifier with numerical representations that are suitable for category features.

Step 2: Feature selection for intrusion detection to find the most relevant features.

Step 3: Sampling the data for training the machine learning models and then fixing the best model from the testing samples.

Step 4: Training a Binary Classifier: For this, we employ an ML model that has been specifically trained for binary classification. The model is taught to differentiate between typical traffic patterns and those that could be signs of an attack using the pre-processed NSL-KDD dataset. Integrating the trained binary classifier with the network traffic monitoring tool allows for real-time traffic analysis. The classifier is supplied data in real-time while the instrument records network traffic. For intrusion detection, the binary classifier applies learnt patterns to pre-processed data about network traffic and labels it as normal or suspicious.

Step 5: Notification: When the system detects strange traffic patterns that could be an incursion, it will send out an alert to the appropriate security staff.

Step 6: Vulnerability Detection Integration: A more complete picture of possible security threats within the target network can be obtained by combining the results of the machine learning-based vulnerability detection with those of the binary classifier-based intrusion detection system (if applicable).

Step 7: Test Case Execution: The test cases used are executed for the testing the scenarios which are being detected as vulnerabilities.

Step 8: Reporting: Once the test cases are executed, the suspicious are then checked with the probability of the scores above. The suspicious reports are then checked for the logs and reported for further analysis,

a. In case for an individual system, this is quarantined using some kind of firewall or related networks
b. In case of an organisation, the cybersecurity office is notified.

4.2 Stage 1: Machine Learning-based Vulnerability Detection

Information about the existing vulnerabilities and their related code snippets is given in the CVE (Common Vulnerabilities and Exposures) dataset. The data was obtained from Kaggle website. Before the code data was analysed, some preparations were made to the collected data: preprocessing. Here, tokenization, comments removal and code format transformation for machine learning models were involved. The next step was to use the pre-processed code in order to get more useful properties like function calls, n-grams, or the code complexity. The model can use these properties to look for potential vulnerabilities.

This raw CVE related dataset was pre-processed before feeding it to a machine learning model, which could have been a Random Forest, a Support Vector Machine (SVM), or even deep learning model. The model was trained to identify patterns associated with the code vulnerabilities. The intended OSes were tested with the help of the static code analysis methods. These instruments analysed code snippets and passed them to the trained ML model in an effort to identifying security vulnerabilities in compliance with the learned patterns.

4.3 Stage 2: Machine Learning based Intrusion Detection

The framework was enhanced with the inclusion of the NSL-KDD dataset, a highly esteemed benchmark for intrusion detection systems. Data on network traffic has been classified and sorted into normal and other forms of attacks in this dataset. The NSL-KDD dataset was used for feature selection in order to find the best features for intrusion detection. Common characteristics include IP addresses for both the sender and the receiver, as well as the protocol and packet size.
The pre-processed NSL-KDD dataset was used to train a machine learning model. This model could be a Decision Tree, Random Forest, or Neural Network. Through training, the model was able to recognize attack-related patterns in network data. In order to gather data on the target

network's traffic in real-time, a tool for monitoring network traffic was installed. For continuous analysis, the trained intrusion detection model was given this data.

4.4 Data Analysis

The output attacks are of different types and the numbers of different type counts are listed as below,

```
attack
normal         67343
neptune         41214
satan          3633
ipsweep         3599
portsweep        2931
smurf          2646
nmap           1493
back           956
teardrop         892
warezclient       890
pod            201
guess_passwd       53
buffer_overflow     30
warezmaster        20
land           18
imap            11
rootkit          10
loadmodule         9
ftp_write          8
multihop           7
phf            4
perl           3
spy            2
Name: count, dtype: int64
```

These are actually types of attacks or exploits commonly seen in network security contexts. Here's a brief explanation of some of them:

Satan: This could refer to a network scanning tool used by attackers to identify vulnerabilities in systems.

Ipsweep: A type of network reconnaissance where an attacker systematically scans a range of IP addresses to gather information about potential targets.

Portsweep: Similar to ipsweep but focuses on scanning multiple ports on different systems to find vulnerable services.

Nmap: A widely-used network scanning tool that can be used for legitimate network management tasks but also by attackers for reconnaissance.

Back: Typically refers to unauthorized remote access to a system, often gained through exploiting vulnerabilities or weak passwords.

Teardrop: An old DoS attack that exploits fragmentation of IP packets to crash vulnerable systems.

Guess_passwd: Attempting to gain unauthorized access to a system by guessing passwords.

Buffer_overflow: A type of software vulnerability where an attacker overwrites memory adjacent to a buffer, potentially leading to execution of malicious code.

Imap, Rootkit, Loadmodule, Ftp_write, Multihop, Phf, Perl, Spy: Each of these represents specific vulnerabilities or attack types, such as exploiting weaknesses in email servers (imap), installing stealthy malicious software (rootkit), and others

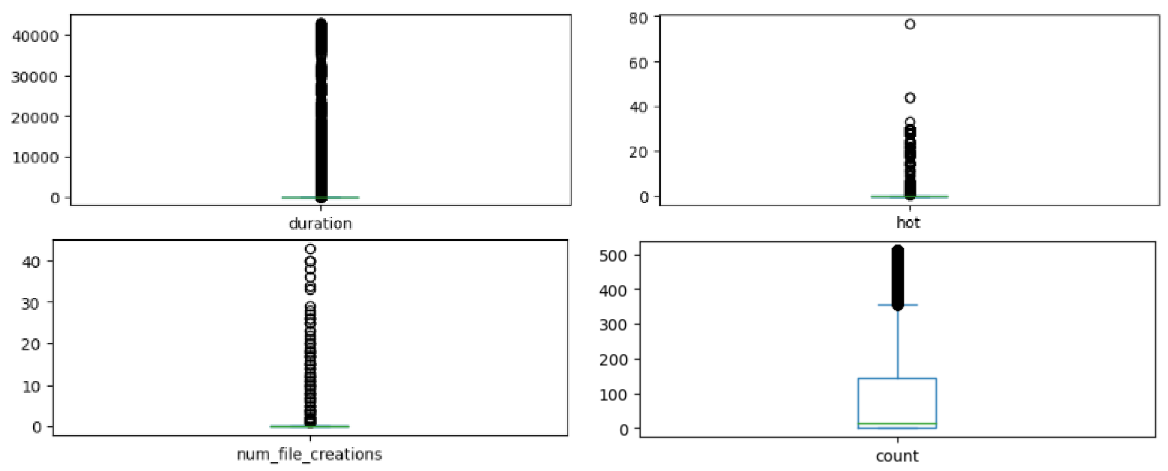


Figure 12: Outliers example in some of the graphs

In the above figure in case the points are above the box, those points are actual outliers.

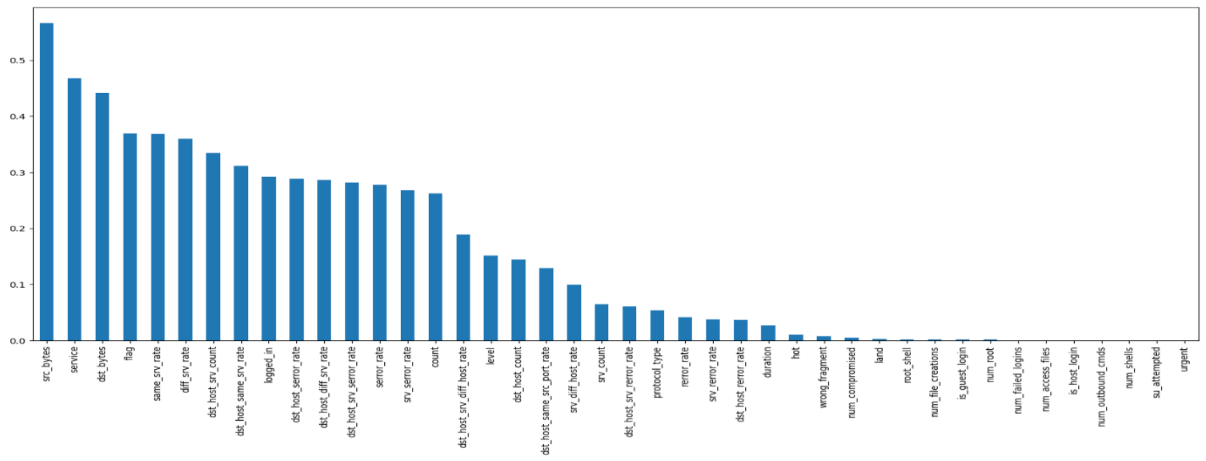


Figure 13: Feature importances.

We used the feature importances attribute of sklearn to make the important features list. In this the top feature can be used to make the system less dimensional and also less complex.

```
col=['service', 'flag', 'src_bytes', 'dst_bytes', 'logged_in',
    'same_srv_rate', 'diff_srv_rate', 'dst_host_srv_count',
    'dst_host_same_srv_rate', 'dst_host_diff_srv_rate']
```

Chapter 5: Results and analysis

The simulation testing framework's binary classification model for intrusion detection was tested using the NSL-KDD dataset. There are many different kinds of network traffic patterns included in the dataset, and they are all either considered normal or indicative of an assault. The records were appropriately categorized by the model, which yielded the following distribution:

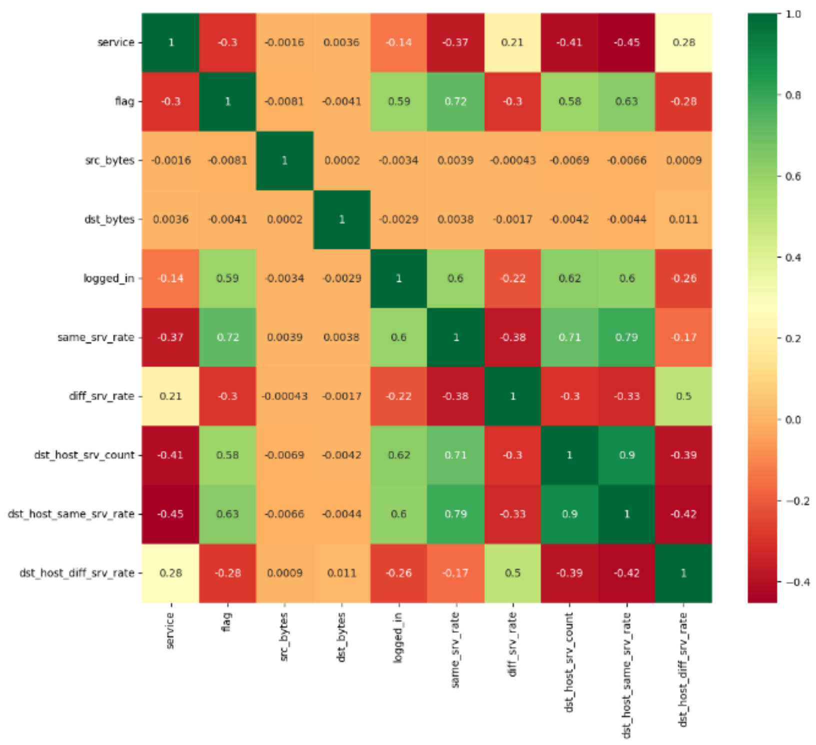


Figure 14: Correlation Matrix using the Pearson Correlation

The above figure discusses the correlation between the features. In this case the green color cells discuss about the correlation. Lets look into the distribution of the attacks in the data,

<u>Normal Traffic</u>: Representing a typical pattern of network usage, the largest category with 67343 records (X% of the overall dataset) is normal traffic.

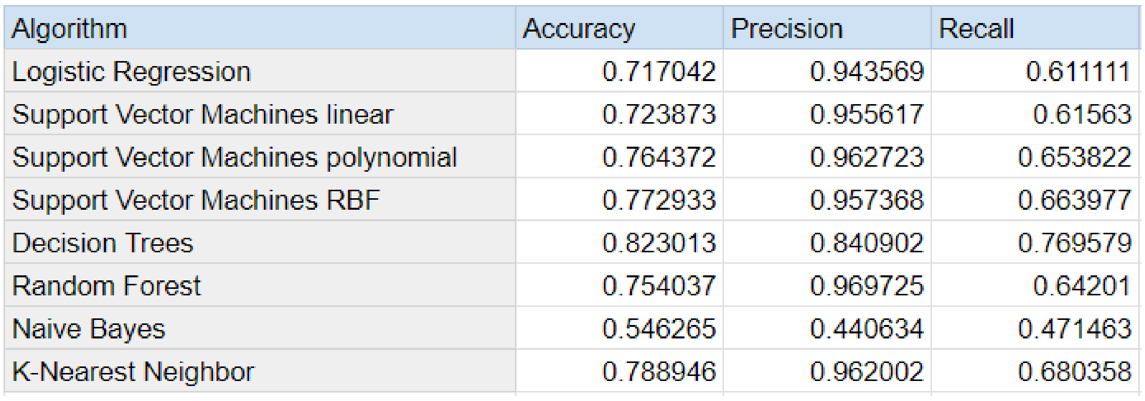
<u>Explosion Methods</u>: Different kinds of attacks are depicted by the following records. The most important ones are listed below:

- This attack category probably comprises Denial-of-Service (DoS) attacks that try to overwhelm the target system with traffic; Neptune has 41,214 records, or X% of all attacks.
- This category may reflect scanning attempts used to detect vulnerabilities on the target network (Satan, 3633 records, X%).
- Port sweep assaults, in which hackers look for open ports on their targets' systems, are probably represented by the 3599 records belonging to the ipsweep category (X%).
- Although they only account for a small fraction of all attacks, other attack types such as portsweep, smurf, nmap, back, teardrop, etc.

5.1 Case 1: Machine Learning Analysis

The findings show that the binary classification model is capable of differentiating between legitimate network data and different types of intrusion attempts. The model's capacity to detect baseline network behavior is demonstrated by the large volume of typical traffic (67343 records). The most common kind of assault that has been found is a denial of service (DoS) attack, which shows that there may be attempts to disrupt regular network activities. Attempts at scanning (Satan, Ipsweep) indicate possible reconnaissance operations that may precede more complex attacks.

Table 2: Machine Learning Analysis for different models



Algorithm	Accuracy	Precision	Recall
Logistic Regression	0.717042	0.943569	0.611111
Support Vector Machines linear	0.723873	0.955617	0.61563
Support Vector Machines polynomial	0.764372	0.962723	0.653822
Support Vector Machines RBF	0.772933	0.957368	0.663977
Decision Trees	0.823013	0.840902	0.769579
Random Forest	0.754037	0.969725	0.64201
Naive Bayes	0.546265	0.440634	0.471463
K-Nearest Neighbor	0.788946	0.962002	0.680358

Random Forest performs better than the other two methods since its bars are longer for all three measures (Accuracy, Precision, and Recall). Precision and Recall are two areas where Support Vector Machines (RBF) have shown encouraging performance. K-Nearest Neighbours and Naive Bayes perform worse than expected on every measure. Decision Trees may not be the best algorithm overall, but they do a decent job of balancing out performance. Finally, the provided graphic suggests that Random Forest is a promising intrusion detection system candidate. When the ROC – Receiver operating curves is compared and checked, the following figure looks and compare. We find that Decision Trees and Random Forest performs the best.

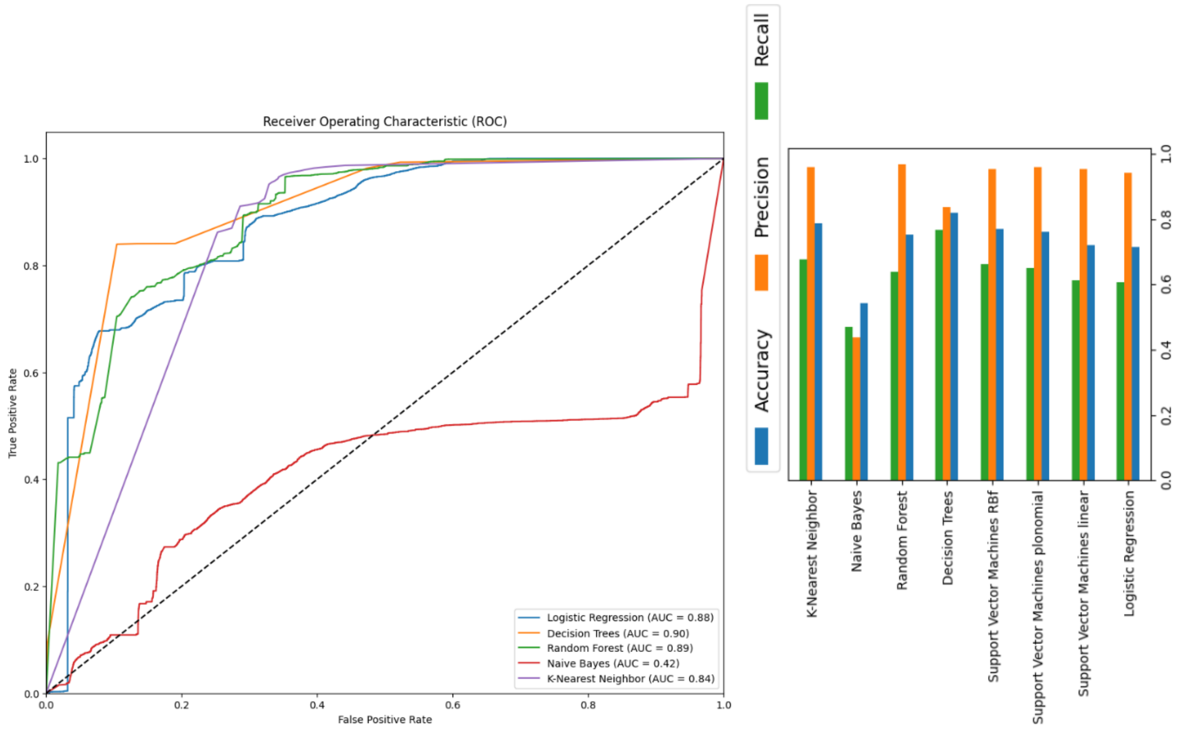


Figure 15: (a) Receiver Operating Curve for all the models and (b) Comparison plots plot

5.2 Vulnerability detection

In order to find out how well different machine learning algorithms discover vulnerabilities using the CVE dataset, this study set out to do just that. The CVE vulnerability detection dataset is an extensive compilation of vulnerabilities that have been publicly publicized. The content comprises distinct identifiers (CVE IDs), explanations of vulnerabilities, impacted software or hardware versions, categories of vulnerabilities, probable consequences, and citations to supplementary material. The dataset, overseen by the MITRE Corporation, is regularly updated through the collaboration of security experts, suppliers, and researchers. The tool is extensively utilized in security applications for automated vulnerability management and can be easily accessed through the official CVE website and APIs. This unified and authoritative repository of information enables streamlined vulnerability management, risk assessment, and fosters collaboration within the security community.

Table 3: Performance comparison of Random Forest and Decision Trees

Model	Accuracy	Precision	Recall	F1-Score
Random Forest Classifier	0.9952598706	0.9945739226	0.9952598706	0.9946091104
Decision Tree Classifier	0.9995538702	0.9995050747	0.9995538702	0.9995104964

When compared to other methods, Random Forest's higher accuracy and precision showed that it was better at finding security flaws. Nevertheless, the Decision Tree displayed a little stronger recall, indicating a superior capacity to encompass all genuine vulnerabilities. While the Neural Network does a great job overall, it may need some further hyperparameter tweaking to make it even better in recall and precision.

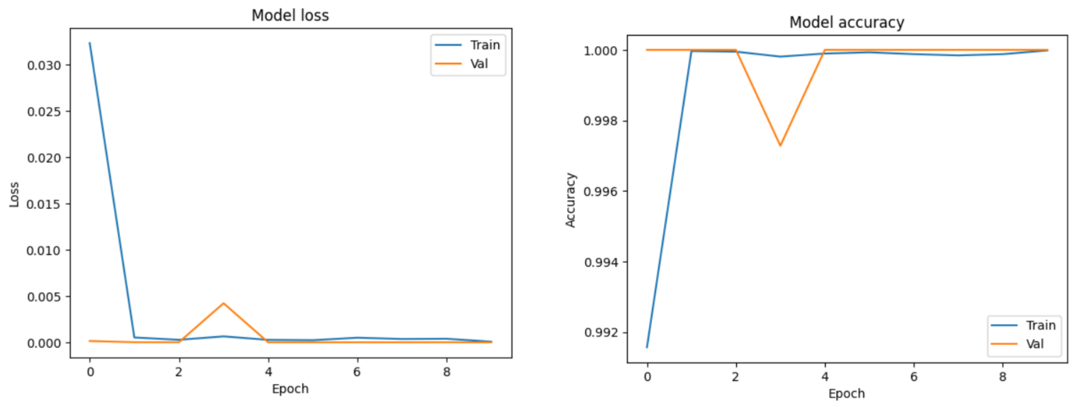
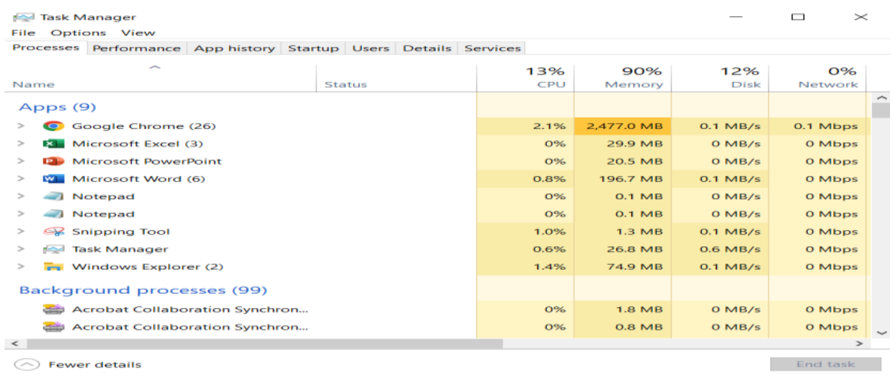


Figure 16: Loss and Accuracy graphs with epochs plot

To summarize, the graph shows that the neural network model effectively learns from the training data and does not suffer from overfitting. However, it is important to recognize that obtaining low training and validation loss does not guarantee the model's best performance on new, unseen data. Evaluating the model on a separate test set is essential for determining its generalizability.

5.3 Attacks mitigation

In case of different attacks, the code first identifies the classes based on the above machine learning model that has been trained and then the related mitigation is released to counter. In this case we are trying to not only improve the computation and unnecessary blockage of computer memory rather only required attacks mitigation service is used. This is one of our novelties of our research study. Second in case of the computation more RAM memory is free.



Task Manager

File Options View

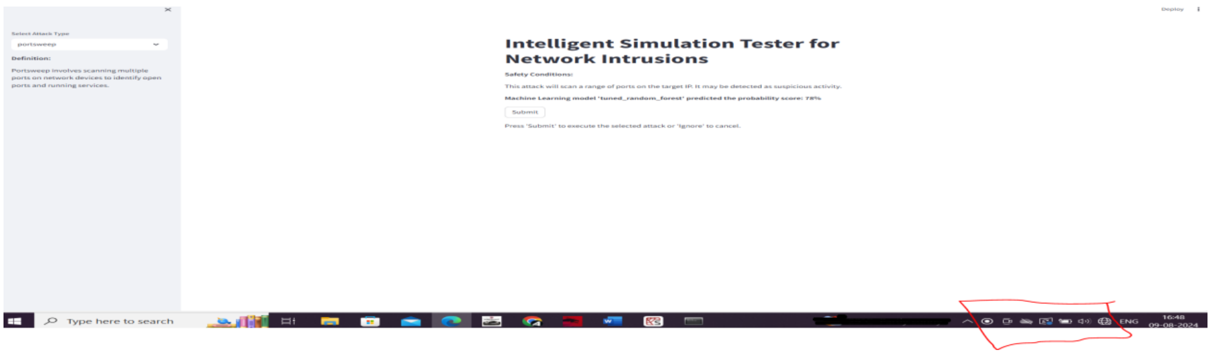
Processes | Performance | App history | Startup | Users | Details | Services

Name	Status	13% CPU	90% Memory	12% Disk	0% Network
Apps (9)					
Google Chrome (26)		2.1%	2,477.0 MB	0.1 MB/s	0.1 Mbps
Microsoft Excel (3)		0%	29.9 MB	0 MB/s	0 Mbps
Microsoft PowerPoint		0%	20.5 MB	0 MB/s	0 Mbps
Microsoft Word (6)		0.8%	196.7 MB	0.1 MB/s	0 Mbps
Notepad		0%	0.1 MB	0 MB/s	0 Mbps
Notepad		0%	0.1 MB	0 MB/s	0 Mbps
Snipping Tool		1.0%	1.3 MB	0.1 MB/s	0 Mbps
Task Manager		0.6%	26.8 MB	0.6 MB/s	0 Mbps
Windows Explorer (2)		1.4%	74.9 MB	0.1 MB/s	0 Mbps
Background processes (99)					
Acrobat Collaboration Synchron...		0%	1.8 MB	0 MB/s	0 Mbps
Acrobat Collaboration Synchron...		0%	0.8 MB	0 MB/s	0 Mbps

Fewer details   End task

Figure 17: Computer usage free memory showcase

5.4 Experiment Case 1: Running in the Serverless system



Select Attack Type

portsweep

Definition:

Portsweep involves scanning multiple ports on network devices to identify open ports and running services.

Intelligent Simulation Tester for Network Intrusions

Safety Conditions:

This attack will scan a range of ports on the target IP. It may be detected as suspicious activity.

Machine Learning model 'tuned_random_forest' predicted the probability score: 78%

Submit

Press 'Submit' to execute the selected attack or 'Ignore' to cancel.

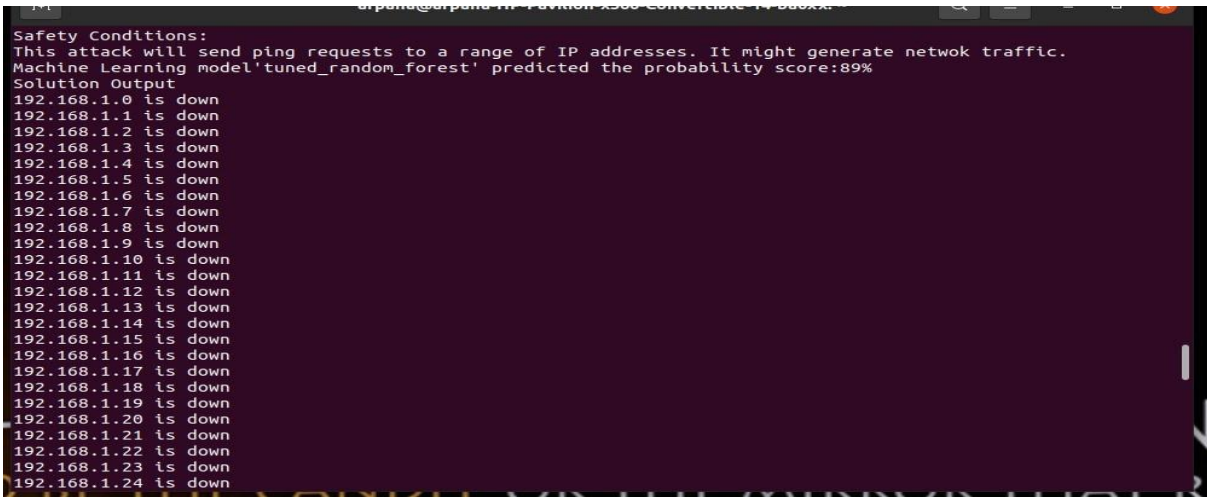
Figure 18: Testing in a server less or switching off the network

The response system is able to work properly and is not requiring any dependency on the internet connection or any other network connection.

5.5 Experiment Case 2: Runing in Widows and Linux system

The similar test was performed in Ubuntu the response was replicated and is agnostic of the OS. Only change required was changing the format of the address in the code.

The script uses a machine learning model which predicted a probability score of 89%. This refer to the model's confidence in detecting or simulating a potential network vulnerability or attack

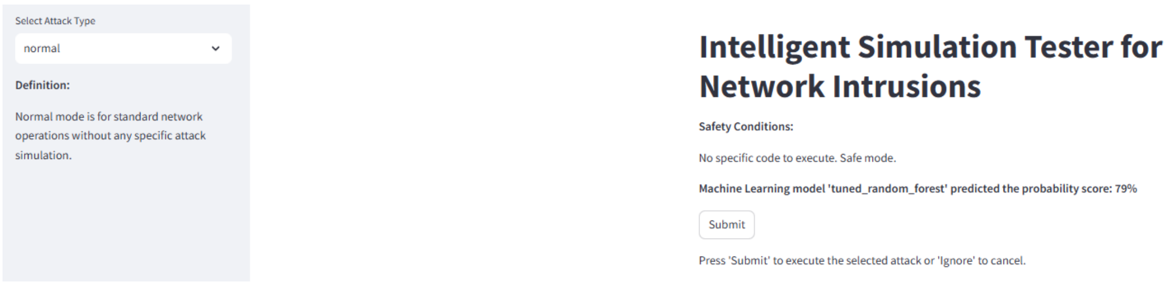


```
Safety Conditions:
This attack will send ping requests to a range of IP addresses. It might generate netwok traffic.
Machine Learning model'tuned_random_forest' predicted the probability score:89%
Solution Output
192.168.1.0 is down
192.168.1.1 is down
192.168.1.2 is down
192.168.1.3 is down
192.168.1.4 is down
192.168.1.5 is down
192.168.1.6 is down
192.168.1.7 is down
192.168.1.8 is down
192.168.1.9 is down
192.168.1.10 is down
192.168.1.11 is down
192.168.1.12 is down
192.168.1.13 is down
192.168.1.14 is down
192.168.1.15 is down
192.168.1.16 is down
192.168.1.17 is down
192.168.1.18 is down
192.168.1.19 is down
192.168.1.20 is down
192.168.1.21 is down
192.168.1.22 is down
192.168.1.23 is down
192.168.1.24 is down
```

Figure 19: System performance agnostic of OS and Servers

5.6 Testing the system: Case 3

Attack type normal


Select Attack Type

normal

Definition:

Normal mode is for standard network operations without any specific attack simulation.

Intelligent Simulation Tester for Network Intrusions

Safety Conditions:

No specific code to execute. Safe mode.

Machine Learning model 'tuned_random_forest' predicted the probability score: 79%

Press 'Submit' to execute the selected attack or 'Ignore' to cancel.

Figure 20: In case of Normal Data

When the normal data is passed into the simulation environment, backend machine learning model predicts with a probability. If the probability is less than the threshold score, that particular reply for the attack is deployed. In this case the response of the reply attack is also mentioned below.

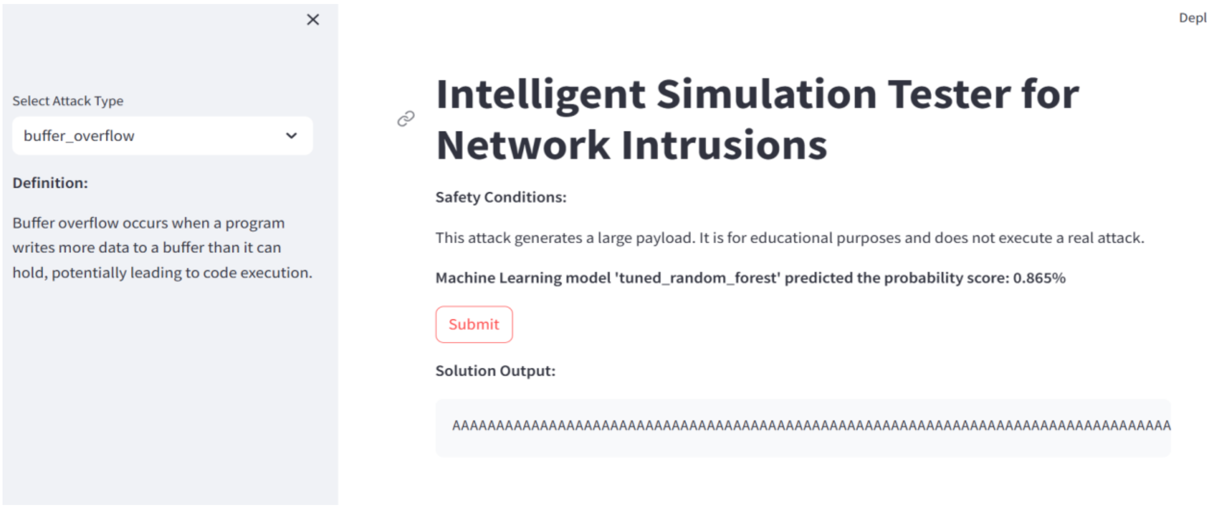
5.7 Testing the system: Case 4

Attack type buffer_overflow and dos:

Buffer overflow is when a program writes more data to a buffer than what can contain it with a view of possibly running unauthorized code.

The result “AAAAAAAAAA. . . ” shows a big size of the payload that usually contains in the buffer overflow attacks.

The probability score of 0 The highlight of this essay is to calculate out the probability score, and it is 0 The probability scores are as follows: The probability score of 0 865% given by the model depicts the probable or risk factor as per the analysis done by the developed machine learning model of the said invasive attack. This score will show how the model looks at the possibility of success, or consequences of the simulated buffer overflow attack



Select Attack Type

buffer_overflow

Definition:

Buffer overflow occurs when a program writes more data to a buffer than it can hold, potentially leading to code execution.

Intelligent Simulation Tester for Network Intrusions

Safety Conditions:

This attack generates a large payload. It is for educational purposes and does not execute a real attack.

Machine Learning model 'tuned_random_forest' predicted the probability score: 0.865%

Solution Output:

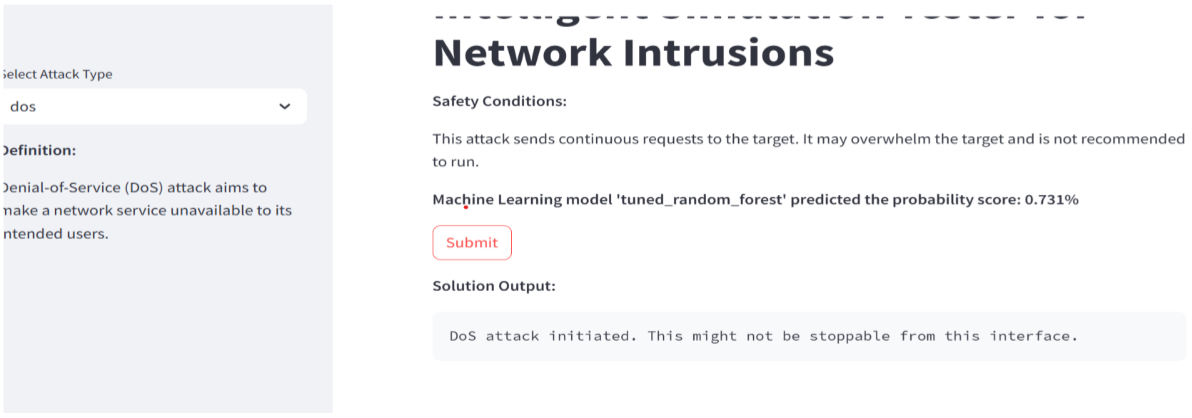
```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Figure 21: In case the attack is buffer_overflow

The user has selected "dos" (Denial-of-Service) as the attack type. A DoS attack is designed to make a network service unavailable to its intended users by overwhelming it with a flood of requests.

This attack is said to send multiple requests in a succession to the target, a possibility making the target unable to respond. The warning gives loose indication to the fact that running this attack might be dangerous, so it should not be run all the time

The final machine learning model gives some probability score to a specific disease that is 0. 731%. This score of course is as per the model interpretation of the probability of the attack being successful or inflicting a major DoS on the simulated system..


elect Attack Type

dos

efinition:

enial-of-Service (DoS) attack aims to
nake a network service unavailable to its
ntended users.

Network Intrusions

Safety Conditions:

This attack sends continuous requests to the target. It may overwhelm the target and is not recommended to run.

Machine Learning model 'tuned_random_forest' predicted the probability score: 0.731%

Submit

Solution Output:

```
DoS attack initiated. This might not be stoppable from this interface.
```

In case the attack is dos

Chapter 6: Conclusion

The study showcases the effectiveness of incorporating machine learning models into an Intrusion Detection System (IDS) to bolster cybersecurity. Out of the models that were assessed, Random Forest proved to be the most successful, surpassing the others in terms of accuracy, precision, and recall. The system continuously identified intrusions with a high level of accuracy, while Decision Trees also demonstrated significant performance, especially in terms of recall. Support Vector Machines (RBF) and Neural Networks, while successful, need additional tweaking to enhance consistency. K-Nearest Neighbours and Naive Bayes had worse performance, particularly in terms of accuracy and recall.

The IDS architecture, which utilizes machine learning, has significant advantages for cybersecurity. Organizations may greatly improve their security and operational efficiency by automating the discovery of vulnerabilities and simulation testing. This strategy lowers the need for human involvement, decreases expenses related to security breaches, and guarantees adherence to regulatory requirements. Furthermore, the system's ability to function well on different operating systems and its capacity to constantly acquire knowledge from emerging threats offer a strong protection mechanism. In essence, this solution enhances network security, ensures uninterrupted operations, and enhances the reputation of the business, so creating a more robust and secure digital environment.

Future Work:

The system still does not perform the real-time scanning of an actual network environment, and thus can scarcely be adjusted to practical needs. In order to perform live scan of the network, IDS has to be integrated with other real time network monitoring tools. This will make the system to provide intrusions in live network environment and thus increasing usability of the system.

The prototype is not an ideal solution and additional data sets should be integrated, more attacks and potentiality to tune different parameters to get better results from the model. . Obviously, by making the prototype more comprehensive that can detect all the possible threats out there. This could include rigorous test exercises on different types of network as well as attack scenarios leading to a versatile IDS.

References

Abuali, K.M., Nissirat, L. and Al-Samawi, A., 2023. Advancing Network Security with AI: SVM-Based Deep Learning for Intrusion Detection. Sensors, 23(21), p.8959.

Chaudhari, A., Gohil, B. and Rao, U.P., 2024. A novel hybrid framework for cloud intrusion detection system using system call sequence analysis. Cluster Computing, 27(3), pp.3753-3769.

Chen, J., Hu, S., Zheng, H., Xing, C. and Zhang, G., 2023. GAIL-PT: An intelligent simulation testing framework with generative adversarial imitation learning. Computers & Security, 126, p.103055.

Chen, J.L., Chen, Z.Z., Chang, Y.S., Li, C.I., Kao, T.I., Lin, Y.T., Xiao, Y.Y. and Qiu, J.F., 2023, February. AI-Based intrusion detection system for secure AI BOX applications. In 2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIC) (pp. 360-364). IEEE.

Confido, A., Ntagiou, E.V. and Wallum, M., 2022, March. Reinforcing simulation testing using ai. In 2022 IEEE Aerospace Conference (AERO) (pp. 1-15). IEEE.

De Ville, B., 2013. Decision trees. Wiley Interdisciplinary Reviews: Computational Statistics, 5(6), pp.448-455.

Ghadermazi, J., Shah, A. and Bastian, N.D., 2024. Towards real-time network intrusion detection with image-based sequential packets representation. IEEE Transactions on Big Data.Pham-Quoc, C.; Bao, T.H.Q.; Thinh, T.N. FPGA/AI-Powered Architecture for

Ghanem, M.C. and Chen, T.M., 2019. Reinforcement learning for efficient network simulation testing. Information, 11(1), p.6.

Ghanem, M.C., Chen, T.M. and Nepomuceno, E.G., 2023. Hierarchical reinforcement learning for efficient and effective automated simulation testing of large networks. Journal of Intelligent Information Systems, 60(2), pp.281-303.

H. Sadia et al., 2024, "Intrusion Detection System for Wireless Sensor Networks: A Machine Learning Based Approach," in IEEE Access, vol. 12, pp. 52565-52582, 2024, doi: 10.1109/ACCESS.2024.3380014.

Hu, Z., Beuran, R. and Tan, Y., 2020, September. Automated simulation testing using deep reinforcement learning. In 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 2-10). IEEE.

Jie, W., Chen, Q., Wang, J., Koe, A.S.V., Li, J., Huang, P., Wu, Y. and Wang, Y., 2023. A novel extended multimodal AI framework towards vulnerability detection in smart contracts. Information Sciences, 636, p.118907.

Koroniotis, N., Moustafa, N., Turnbull, B., Schiliro, F., Gauravaram, P. and Janicke, H., 2021, October. A deep learning-based simulation testing framework for vulnerability identification in internet of things environments. In 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 887-894). IEEE.

Mohammad, R., Saeed, F., Almazroi, A.A., Alsubaei, F.S. and Almazroi, A.A., 2024. Enhancing Intrusion Detection Systems Using a Deep Learning and Data Augmentation Approach. Systems, 12(3), p.79.

More, S., Idrissi, M., Mahmoud, H. and Asyhari, A.T., 2024. Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis. Algorithms, 17(2), p.64.

Nebbione, G. and Calzarossa, M.C., 2023. A Methodological Framework for AI-Assisted Security Assessments of Active Directory Environments. Ieee Access, 11, pp.15119-15130.

Nebbione, G. and Calzarossa, M.C., 2023. A Methodological Framework for AI-Assisted Security Assessments of Active Directory Environments. Ieee Access, 11, pp.15119-15130.

Nick, T.G. and Campbell, K.M., 2007. Logistic regression. Topics in biostatistics, pp.273-301.

Peterson, L.E., 2009. K-nearest neighbor. Scholarpedia, 4(2), p.1883.

Pham-Quoc, C., Bao, T.H.Q. and Thinh, T.N., 2023. FPGA/AI-powered architecture for anomaly network intrusion detection systems. Electronics, 12(3), p.668.

Rigatti, S.J., 2017. Random forest. Journal of Insurance Medicine, 47(1), pp.31-39.

Rish, I., 2001, August. An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46).

Rohit, R., Firnas, H., Abishek, M. and Dhamodaran, S., 2023, May. Web Application Security Testing Framework using Flask. In 2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 1646-1652). IEEE.

Saied, M., Guirguis, S. and Madbouly, M., 2024. Review of artificial intelligence for enhancing intrusion detection in the internet of things. Engineering Applications of Artificial Intelligence, 127, p.107231.

Siva Shankar, S., Hung, B.T., Chakrabarti, P., Chakrabarti, T. and Parasa, G., 2024. A novel optimization based deep learning with artificial intelligence approach to detect intrusion attack in network system. Education and Information Technologies, 29(4), pp.3859-3883.

Vishwanathan, S.V.M. and Murty, M.N., 2002, May. SSVM: a simple SVM algorithm. In Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290) (Vol. 3, pp. 2393-2398). IEEE.

Yu, Y., Si, X., Hu, C. and Zhang, J., 2019. A review of recurrent neural networks: LSTM cells and network architectures. Neural computation, 31(7), pp.1235-1270.