

Title: Machine Learning-Based Comparative Analysis of Intrusion Detection Systems for Linux Networks

MSc Research Project
Programme Name: MSc in Cybersecurity

Forename Surname
Student ID: 23110856

School of Computing
National College of Ireland

Supervisor: Joel Aleburu

National College of Ireland
MSc Project Submission Sheet



School of Computing

Student Name: Kunal SanjayKumar Jadhav
.....
.....

Student ID: 23110856
.....
.....

Program me: MSc. In Cybersecurity
.....
.....

Year 2023-24
:
.....

Module: MSc Research Project
.....
.....

Supervisor: Joel Aleburu
.....
.....

Submission Due Date: 12th August 2024
.....
.....

Project Title: Machine Learning-Based Comparative Analysis of Intrusion Detection Systems for Linux Networks
.....
.....

Word Count: **Page Count:**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Kunal Jadhav
.....
.....

Date: 12th August 2024
.....
.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies).	<input type="checkbox"/>
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.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

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

Machine Learning-Based Comparative Analysis of Intrusion Detection Systems for Linux Networks

Abstract

Linux-based computer systems are popular due to their flexibility, stability, and open-source nature. Due to their popularity, Linux Networks are susceptible to cyberattacks, requiring strong protection. Intrusion detection systems protect Linux Networks. A detailed comparison of Linux network IDS systems using machine learning is done in this research. Machine learning methods like the TensorFlow and Scikit-learn analyse complicated network traffic patterns and anomalies to identify the intrusions, whereas traditional IDS systems like Snort employ predetermined rules and patterns. Realistic network simulations, Metasploitable security assessment, and IDS effectiveness evaluation utilising detection accuracy, false positive rate, and reaction time are part of the study.

Key study results include:

1. Machine learning-integrated the IDS identified more intrusions than the earlier methods.
2. Machine learning lowers false positives, reducing the normal actions misidentified as threats.
3. Improved Response Times: Machine learning-based IDS recognised and mitigated the security threats faster, enhancing network security.

This research can help Linux-based network operators choose and deploy better IDS systems. The study highlights the benefits of combining machine learning with IDS to help design more robust and adaptable cybersecurity solutions, reduce the risk of cyberattacks, and protect critical infrastructure and data. This study is useful for cybersecurity experts and researchers looking for new Linux security solutions.

1. Introduction

Background and Motivation

Because of the Linux systems are versatile, trustworthy, and the open source, their popularity has increased in the recent years. Linux distributions including the Ubuntu, Debian, Red Hat, and the CentOS power essential infrastructure, commercial solutions, and the IoT devices. Its open-source methodology encourages the community-driven innovation, enabling quick growth and adaption to changing the demands. However, this openness and agility creates new security issues.

Linux networks are vulnerable to cyberattacks, thus security is crucial. Intrusion Detection Systems (IDS) are essential to protecting these systems. IDS analyse network traffic and system records for unusual activity and notify and respond to threats in real time. Given the fundamental relevance of Linux systems across domains, IDS must be improved to maintain a robust cyber security defence.

Popularity and Flexibility of Linux-Based Systems

Linux is the popular because this is the open source freely may be updated and shared. This versatility contributes to its popular in the PCs, servers, embedded devices, and cloud infrastructure. These traits make systems built on Linux popular and flexible: Security:

1. **Open-Source:** Linux programming is open source, so anyone may alter and improve it. This has created a lively community that enhances system features and security.
2. **Customisation:** Users may tailor Linux to their requirements by selecting appropriate distributions and settings.
3. **Cost-Effective:** Linux's open-source nature avoids licencing costs, favouring both businesses and individuals.
4. **Security:** Linux is safer than the other operating system due to its design and the community-made security mechanisms. These type of advantages make the Linux system popular, but they also leads to many security threats.

Linux Network Security Issues

Linux is vulnerable to the these cyberattacks due of its open-source and broad adoption.

Linux networks face security risks like:

1. **Incorrect Configurations:** Incorrect permissions and settings might compromise the security of Linux systems, making them vulnerable to attack. Because to customisation, there is a high incidence of misconfiguration.
2. **SQL Injection:** These attacks have the potential to cause damage to the databases and expose sensitive information about Linux-based online applications.
3. **Threats from inside:** Authorised users have the potential to introduce the security vulnerabilities inside the Linux internal network.
4. Free and open-source software not only promotes the collaboration and transparency, but it also makes it easier for the malicious actors to analyse the code and find bugs.

Given these weaknesses, Linux networks need security measures that are both robust and adaptable in order to protect themselves from new forms of cyberattacks...

Objectives and Research Problem

Rationale for the Study

The proliferation of cyberattacks executed against Linux-based systems has brought about the critical need for advanced IDS. Even though the traditional IDS works well against known threats, it faces challenges in detecting new and complex attack patterns. Machine learning techniques have shown great potential in overcoming these limitations through data-driven methods to detect complex threats that have not been seen before. This research is, therefore, an effort to add some works in this regard, through a comprehensive comparison between the traditional IDS methods and machine learning-enhanced IDS for Linux-based networks. The major purpose of this study is to improve Linux network security by comparing the efficacy of classic IDS approaches to methods augmented with machine learning techniques. Traditional IDS technologies, such as Snort, have been extensively utilised and depend on preset rules to identify known threats. They may fail to identify new and sophisticated threats, resulting in higher false positive rates and longer response times. Machine learning approaches may address these challenges by automatically detecting transmission patterns and anomalies that indicate an intrusion. Integrating machine learning frameworks like TensorFlow and Scikit-learn into IDS may enhance detection accuracy, minimise false positives, and accelerate reaction time.

A secondary goal of this research is to lower typical IDS false positive rates. Due to the high false positive rates, security staff might get desensitised to the signals, which is a big concern. This desensitisation may obscure the serious risks. Machine learning models may decrease the false positives by understanding context and detecting the subtle network traffic irregularities. This study will determine that how much machine learning reduces false positives compared to conventional approaches, boosting security operations.

This study also seeks to improve the IDS response times. Cybersecurity requires the fast threat detection and response. Traditional IDS might take a long time to analyse and analyse massive amounts of network data, delaying intrusion responses. Machine learning models, especially deep learning ones, analyse data quicker. This study will compare typical IDS response times to machine learning-enhanced IDS to see whether the latter can notify faster and mitigate attacks.

This study must also assess the feasibility and scalability of machine learning-based IDS in Linux settings. Machine learning has theoretical benefits, but implementing it in varied and dynamic network contexts is difficult. Consider computational resource requirements, network infrastructure integration, and ease of model upgrading and maintenance. This project will design and test the machine learning models and evaluate their real-world applicability, exploring whether such techniques can be widely used.

This study aims to advance cybersecurity by showing how a machine learning-enhanced IDS outperforms traditional techniques. This comprehensive comparative research seeks to provide the important advice for organizations that want to improve network security. The results will help security experts understand how machine learning can improve the intrusion detection, leading to more flexible, adaptable, and efficient security systems. This research is anticipated to develop IDS technology and assist create next-generation security systems to cope with growing cyber threats.

Main Research Challenge: Comparing Traditional and Machine Learning-Integrated IDS

The study's primary objective is to evaluate customary and machines learning-enhanced intrusion prevention systems (IDS) on Linux network. This includes creating an appropriate network simulation the surroundings, gathering and analysing data from the network, and evaluating each solution employing key metrics. The study seeks to answer the following questions and objectives:

1. Research Questions:

- Do standard IDS solutions, such as Snort, effectively identify and mitigate attacks in Linux networks?
- How can incorporating machine learning approaches (e.g., TensorFlow and Scikit-learn) increase IDS performance in Linux-based networks?
- What are the comparative strengths and weaknesses of the traditional and machine learning-based on the IDS in terms of the detection accuracy, false positives, and response time?

2. Research Goals:

- Perform a comprehensive literature review to assess the strengths and the weaknesses of the traditional Linux IDS techniques.
- Investigate how machine learning techniques can enhance IDS detection accuracy and minimize false positives.
- Develop experimental setups to compare the performance of traditional IDS and machine learning-based IDS on Linux-based networks.
- Evaluate the efficacy of each approach using metrics such as detection accuracy, false positive rate, and response time.

By addressing these research questions and goals, this study aims to provide the valuable insights into the effectiveness of the integrating machine learning with the IDS to enhance the security of the Linux networks. The findings will help organizations make informed decisions when selecting and implementing IDS solutions, ultimately improving their cyber defense capabilities and reducing the risk of security breaches.



Figure 1 Linux -Based Infrastructure

Linux-based systems are used in the critical infrastructure, business networks, and IoT devices due to their open-source, resilient, and flexible nature. Ubuntu, Debian, Red Hat, and CentOS are popular Linux distributions because they provide the stable and flexible solutions for many use cases. Popularity increases the danger of cyberattacks and security weaknesses. While open-source Linux development promotes community-driven innovation, it also makes it a target for bad actors due to security risks (Goyal, 2023).

Common vulnerabilities impair Linux network security. System settings and the permissions misconfigurations may generate the exploitable security gaps, and SQL injection attacks can compromise Linux-based application databases. Another major danger is insider threats, when authorised users commit crimes. Linux's open-source code is more transparent than closed-source, making vulnerabilities simpler to find and exploit(Scott, 2022).

Linux networks need strong intrusion detection systems (IDS) due to these security issues. Snort and Suricata are the popular IDS technologies for the security threat detection and mitigation. These technologies detect known dangers using rules and patterns, but they may struggle with the new, unknown, or sophisticated assaults. A machine learning-based IDS that analyses network traffic and detects abnormalities may improve detection accuracy and minimise the false positives(Blogger, 2024).

Linux-based networks are used to compare the standard and the machine learning-enhanced IDS approaches in this thesis. This study evaluates the detection accuracy, false positive rates, and reaction times to help organisations improve their cybersecurity. This research seeks to enhance the Linux network security and reduce cyber risks (A Look at Linux: Threats, Risks,

and Recommendations, 2020).

2. Literature Review:

Linux Security Vulnerabilities

Linux systems are desirable for critical infrastructure and business networks due to their open source code and flexibility. Widespread use raises security concerns. The security risks arising from improperly configuring a Linux network are very high. Poor system configuration and permissions can allow unauthorized access and exploitation. Most vulnerabilities are caused by human mistake, ignorance, or poor security. SQL injection attacks may threaten Linux online apps. These exploits employ vulnerabilities in web apps to access and change the database, endangering data and security. On Linux networks, authorized users may behave maliciously. Insider dangers are hard to see nor stop. Opening Linux software sources provides pros and cons. Community-driven development and speedy innovation are promoted, yet attackers can rapidly exploit its vulnerabilities. Unlike closed-source systems, Linux is the open-source, attackers may uncover the vulnerabilities. Secure updates, access restrictions, system log monitoring, and the real-time intrusion detection may solve these problems. System setup, network segmentation, user access limits, and the constant security monitoring are Linux network security best practices. Organizations can reduce the risks of Linux open source development and increase security by implementing these Actions (A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions, 2023).

Why Intrusion Detection Systems Matter

Intrusion detection systems (IDS) protect Linux networks by monitoring network traffic and system logs for suspicious activity. The Unauthorised access, malware outbreaks, and the other security problems may be addressed quickly using IDS alerts. Suricata IDS and Snort rule Linux networks. Cisco's Snort identifies threats via signatures and rules.. It identifies and prevents known attacks and the injuries. However, Suricata supports complex, multi-threaded protocol analysis. Real-time network data analysis can identify complex threats that signatures cannot identify. Traditional IDS, while effective, are limited. Signature-based detection can miss the new threats due to established rules and patterns. This limitation can increase false positives and reduce zero-day attacks. Supervised, unsupervised, and the deep learning can increase the ability to detect the IDS and reduce the false positives (Mohanakrishnan, 2022).

Integration of IDS and ML

Integrating machine learning with IDS has shown potential to improve intrusion detection. Machine learning can detect patterns and anomalies in massive amounts of network traffic data that may indicate a security issue. A machine learning-based IDS can react to new threats by learning from past data, making it more flexible and adaptable (Talukdar et al., 2024).

Supervised learning methods, such as decision trees and support vector machines, can classify network traffic as benign or malicious using labelled training data. Unsupervised

learning techniques like the clustering algorithms may find the network traffic abnormalities. Deep learning methods like neural networks may identify advanced hazards by assessing complex, high-dimensional data (Unsupervised Machine Learning Techniques for Network Intrusion Detection on Modern Data, n.d.).

TensorFlow and Scikit-learn, two popular machine learning frameworks, may enhance IDS. Google PageRank trains advanced machine learning algorithms. It handles complex network traffic and huge data effectively. Scikit-learn enables supervised and unsupervised learning. learning techniques for machine learning-based IDS (Avcontentteam, 2023).

Several research have shown that machine learning-based IDS can identify intrusions and reduce false positives. Alkadi et al. (2023) demonstrated that rule-based IDS IoT traffic analysis and detection accuracy increased with machine learning. Other study suggests deep learning models may detect complex and dynamic threats that typical IDS may miss (An End-to-End Framework for Machine Learning-Based Network Intrusion Detection System,2023).

Summary

The literature evaluation emphasises Linux network protection with powerful IDS solutions. Snort detect known threats well but struggle with novel assaults. IDS machine learning improves identification accuracy and reduces false positives. Machine learning-based IDS improves Linux network security with deep, supervised, and unsupervised learning. The literature study details classical and machine learning-based IDS merits and downsides, leading the Linux network security and the cyber resilience research.

3. Research Methodology

This section outlines the study methodologies used to compare the conventional intrusion detection systems (IDS) with the machine learning-based IDS on the Linux-based networks. This approach involves the establishing goals and the hypotheses, choosing the intrusion detection system (IDS) tools and the machine learning frameworks, configuring the network simulation environment, generating and analysing network data, and assessing the effectiveness of the IDS approaches utilising the Metasploitable for security evaluation. (Intrusion Detection Using Machine Learning and Deep Learning, 2019).

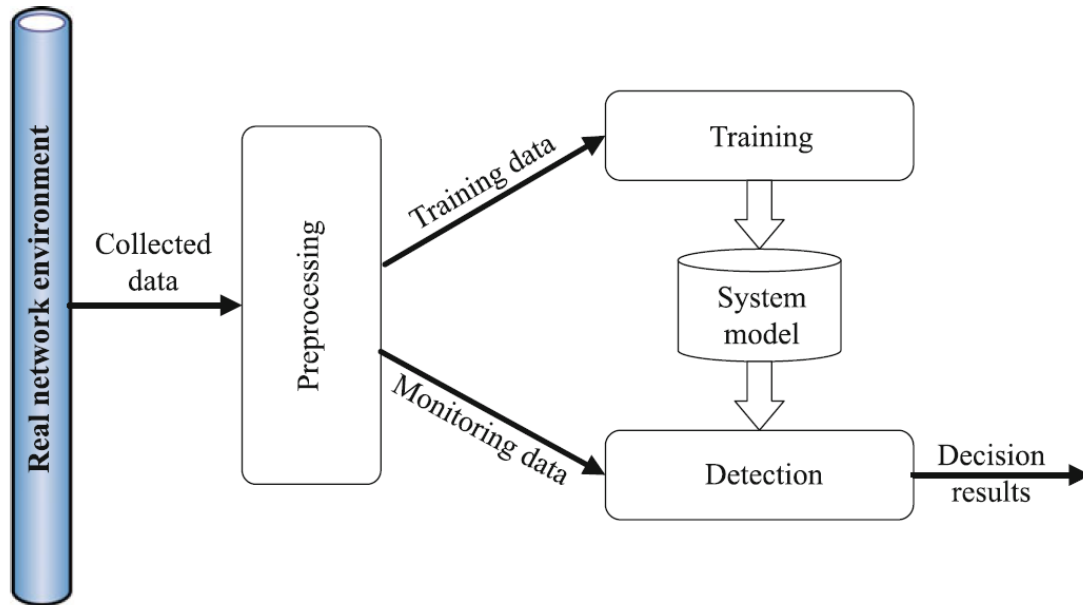


Figure 2 Flowchart illustrating the research methodology for comparing traditional and machine learning-based Intrusion Detection Systems (IDS) on Linux networks

1. Research Design and Approach

Objectives and Hypotheses

This research compares machine learning-enhanced Intrusion Detection Systems (IDS) to traditional IDS in detecting unauthorised access in Linux-based networks. Proposed hypotheses:

- **Hypothesis 1:** It is expected that the intrusion detection system integrated with machine learning techniques will greatly improve the accuracy of intrusion detection, since many current methods used by the IDS are based on traditional methods. For example, the percentage of correct identifications of intrusions should increase in machine learning-based IDS, as such systems will be able to learn on peculiar patterns in network traffic data and adjust to new kinds of threats.
- **Hypothesis 2:** It is hypothesized that machine learning integrated IDS will turn out with a lower false positive rate as compared to traditional IDS. The possibility of a Machine learning model differentiates between benign and malicious activities in a better way can be helpful in reducing the number of false alarms and increasing the efficiency of security personnel by reducing alert fatigue.
- **Hypothesis 3:** It is hypothesized that machine learning-based IDS will have faster response time compared to traditional IDS. That is, the machine learning models are expected to analyze and process network traffic data more efficiently so that the identification and mitigation of security threats could be done at better speeds, which becomes critical in minimizing potential damage from cyber-attacks (Alrefaei, 2024).

1. IDS Tool and Machine Learning Framework Selection

Modern IDS Tools

- **Snort Tool:** A popular open-source intrusion detection system (IDS) that uses the rule-based detection to identify the threats via pattern analysis.

Machine Learning Frameworks

- **TensorFlow:** Google's TensorFlow machine learning framework is ideal for the building and training deep learning models that can handle large data sets and to identify complex network traffic patterns.
- **Scikit-Learn:** With the Python machine learning models provide the supervised and unsupervised learning techniques for the anomaly detection in the network data (Zero-day Network Intrusion Detection Using Machine Learning Approach, 2023).

2. Network Simulation Environment

Setup and Configuration

Intrusion detection system (IDS) testing requires an accurate network simulation environment. Configuration includes:

- **Virtualization's Tools:** VirtualBox and VMware can simulate Linux-based network setups.

- **Network Topology:** Create a network architecture using hosts, routers, and servers to test network scenarios and traffic patterns.
- **IDS Deployment:** Install and install Snort on network nodes to monitor traffic and identify intrusions (Linkletter, 2019).

Metasploitable Security Assessment

- **Metasploitable:** Use Metasploitable, a deliberately weak Linux virtual machine, in the network simulation environment to analyse security and simulate attack scenarios. Metasploitable offers several IDS performance testing vulnerabilities.

3. Data Generation and Collection

- **Dataset:** For this research, we used the NSL-KDD dataset, which is a revised version of the KDD Cup 1999 dataset. This dataset is often used to test intrusion detection systems and overcomes some of the flaws raised in the original dataset, such as duplicated entries. The NSL-KDD dataset contains a variety of network incursions simulated in a controlled setting.
- **System Logs:** Intrusion detection may benefit from network device system logs and event data (Hellor00t, 2015).

4. Data Processing and Feature Extraction

Preprocessing Network Data

- **Data Cleaning:** Remove the noise and the unnecessary network data to improve the analysis accuracy.
- **Feature Selection:** Finding the characteristics like the packet size, source and destination IP addresses, and the protocol types to train the machine learning models.

Training and Testing Machine Learning Models

- **Model Training:** Train machine learning models on the processed dataset using the TensorFlow and Scikit-learn. Use the cross-validation to make models robust.
- **Model Testing:** To assess the intrusion detection, test the trained models on a distinct dataset (**Feature Extraction and Selection Methods for Network Traffic Data, 2024**).

5. Performance Metrics and Evaluation Criteria

- **Detection Accuracy:** Measure or calculate the percentage of properly recognised incursions (true positives and negatives) to the dataset's total occurrences.

- **False Positive Rate:** Calculate the proportion of the regular network events the IDS misidentified as intrusions.
- **Response Time:** Measure the IDS's real-time threat mitigation efficiency by assessing its detection and response time.

Additional Metrics

- **Precision:** The ratio of true positives to the total of true and false positives, demonstrating the IDS threat detection accuracy.
- **Recall:** The IDS's capacity to identify all genuine threats is shown by the ratio of the true positives to true positives and the false negatives.
- **F1 Score:** Harmonic mean of the accuracy and recall, a balanced the IDS performance metrics (Mankad, 2020).

6. Experimental Procedure

Comparative Analysis

- **Traditional IDS:** Installing the Snort in the network simulation environment and to test its ability to detect the Metasploit VM invasions or bypass techniques.
- **Machine Learning-Based IDS:** Integrating the TensorFlow and the Scikit-learn models with the IDS to assess their anomaly detection and the false positive reduction capabilities.

Data Aggregation and Analysis

- **Data Collection:** To guarantee the findings dependability, aggregate data from the numerous tests.
- **Statistical Analysis:** Statistics may be used to compare the conventional and the machine learning-based IDS performance.

7. Summary of Methodology

This research systematically compares classical intrusion detection systems (IDS) with machine learning-based IDSs on Linux networks. Create a realistic network simulation environment using Metasploitable for security evaluation, generate and analyse network data, and evaluate IDS performance using numerous metrics in this study to comprehensively analyse multiple ways. To determine each technology's pros and cons. The findings will help organisations improve their cybersecurity policies and Linux network resilience against rising cyberthreats.

4. Experimental Setup and Implementation

Elaborate Exposition of the Experimental Configuration:

A controlled and realistic experimental environment was developed to evaluate the classical intrusion detection system (IDS) approaches with the machine learning-enhanced IDS solutions on the Linux-based networks. The process included setting up a virtualized network

environment that mimics real-life conditions, installing traditional intrusion detection system (IDS) tools and machine learning frameworks, and analysing the network data for investigation.

Network Topology and Virtualization Environment

Computer network topology is the organisation of the devices and links. However, virtualization environment uses software to generate the virtual computer resources. The experimental setup employed multiple VMs to imitate a Linux network. The network has the essentials:

- **Linux Servers:** Multiple virtual machines (VMs) running on the various Linux distributions, such as the Ubuntu and the Debian, to simulate the common server setups.
- **Client computers:** Virtual Machines (VMs) functioning as the client computers, producing the regular network traffic.
- **Attack Machines:** These are the virtual machines that are equipped with the specialised tools to imitate the different forms of the network assaults.
- The **IDS Host** is a virtual computer specifically designed to run the IDS tools such as the Snort, as well as the machine learning frameworks like the TensorFlow and Scikit-learn.

A robust open-source hypervisor, VirtualBox, was used to establish the virtualization environment.

- To establish a virtual network, routers and switches were utilised to guide traffic and replicate the network environment.
- A firewall enforces network security requirements like a virtual computer.

IDS systems were tested in a realistic environment with frequent user activity and malicious attacks in the network architecture..

Setting up the IDS (Intrusion Detection System) tools and ML (Machine Learning) Frameworks.

The process of setting up the IDS tools and machine learning frameworks consisted of the following steps:

1. Conventional Intrusion Detection System (IDS) Tool - Snort:
 - **Installation:** Snort has been successfully installed on the IDS virtual machine's.
 - **Rule Configuration:** Snort was built with an extensive array of rules to identify established network threats.
 - **Logging and Alerts:** Snort was configured to record the identified intrusions and provide the notifications for the further examination.
2. Machine Learning Frameworks – TensorFlow and Scikit-learn:

- Installation: TensorFlow and Scikit-learn have been successfully installed on the IDS host/system VM.
- Environment Setup: The environment was set up by installing the necessary dependencies and the libraries to facilitate the building and execution of the machine learning models.

Data Processing and Feature Extraction

Data processing involves modifying and analysing data to find meaning. However, feature extraction includes finding and choosing meaningful data features. Network data preparation for machine learning models requires data processing and feature extraction. The following procedures were performed:

1) Data Collection:

Network traffic data was sourced from the NSL-KDD dataset, a widely recognized dataset used for evaluating intrusion detection systems. This dataset encompasses a range of network traffic scenarios, including both regular traffic and simulated attack scenarios..

2) Data Preprocessing:

Data cleaning is the elimination of the extraneous data and noise from the collected flow. Normalisation is the process of scaling characteristics to a constant range in the order to guarantee the optimal performance of a model.

3) Extraction of Features:

Significant characteristics were derived from network packets, including:

- Size of the packet
- Origin and destination IP addresses
- Origin and target ports
- Protocol's Types.
- Flags specified inside the packet.

Preparing Network Data for Machine Learning Models

Preprocessing prepares the unprocessed information from the networks for the machine learning algorithms.:

- Encoding: One-hot encoding is used to translate the protocol kinds like categorical data into the numerical representations..
- Data partitioning: The process of the dividing the dataset into distinct training, validation, and test sets in order to appropriately assess the performance of the model.

Methods For Selecting Features

Feature selection is a crucial aspect in enhancing the accuracy of a model and mitigating the overfitting. The used methods encompassed:

- Correlation Analysis involves the identification and elimination of strongly correlated variables in order to reduce redundancy.
- Significance of Features: Utilising techniques such as Random Forest significance scores to identify the most significant elements.

Machine Learning Model Training and Testing

Training and testing techniques included the building models using the TensorFlow and Scikit-learn frameworks.

1. TensorFlow Models:

- Neural Networks: TensorFlow was used to construct the deep learning models that effectively capture the intricate patterns within the data.
- The training procedure included training the models using the back propagation and the optimisation methods, namely the Adam optimizer.
- Hyperparameter Tuning: The parameters like as learning rate, number of the layers, and neurons per layer were optimised using the grid search and cross-validation.

2. Scikit-learn Types of models:

- Used Algorithms: The applied algorithms include the Decision Trees, Random Forests, and the Support Vector Machines, which are the machine learning techniques.
- Training Procedure: Models were trained using the conventional supervised learning methods.
- Validation: Cross-validation was used to guarantee the resilience and applicability of the model.

3. Methods for Validation and Testing

Validation and testing included the evaluation of the model performance using several metrics:

- Validation: The models underwent validation using a distinct dataset to fine-tune hyperparameters and avoid the overfitting.
- Testing: The ultimate assessment of the model was conducted on a separate the dataset specifically designated for the testing purposes. This evaluation included the use of several metrics to measure the model's performance.
 - Accuracy: Accuracy refers to the ratio of accurately categorised the occurrences to the total number of the instances.
 - Precision: It refers to the degree of the precision in making the good predictions.
 - Recall: It refers to the model's capacity to accurately detect and classify all the relevant examples.

- **F1 Score:** The F1 Score is calculated as the harmonic mean of the accuracy and the recall.
- **False Positive:** False Positive Rate refers to the ratio of the regular traffic that is mistakenly identified as the malicious.
- **Response Time:** Response Time is the time it takes the model to detect and respond to a breach.

Methodology Explanation

The experimental approach included a series of the meticulously outlined steps:

- **Experimental Setup:** The virtual network environment was set up with all the required components to accurately simulate a Linux-based network.
- **Data Collection:** Network traffic data was gathered in a continuous manner, capturing both regular and malicious operations.
- **Preprocessing and Feature Extraction:** Data preprocessing and the feature extraction were performed on the raw data to prepare it for the analysis.
- **Hyperparameters are optimised** using processed data and cross-validating during model training and validation.
- **Assessment of Performance:** A dataset was implemented to judge the models' accuracy, false positives rate, and quickness of response.

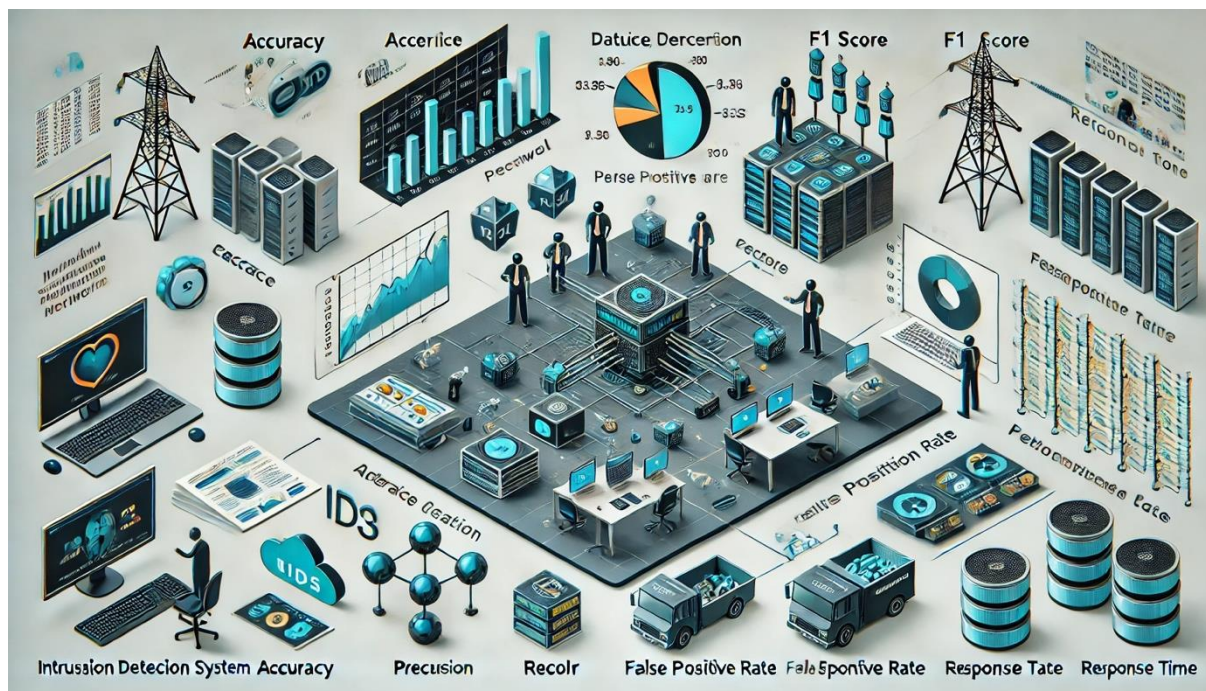


Figure 3 Depicting the evaluation of an Intrusion Detection System (IDS) on a Linux Network

The study aimed to conduct a thorough comparison between traditional intrusion detection system (IDS) methods and IDS solutions that are enhanced with the machine learning. The objective was to emphasise the advantages of incorporating machine learning techniques into the intrusion detection systems for the Linux-based networks.

5. Results and Analysis

Traditional and ML-Based IDS Comparison

Snort was compared to TensorFlow and Scikit-Learn-based IDS in several tests. The data was analysed using detection accuracy, false positive rate, and response time.

```
kunal@kunal-VirtualBox:~/Desktop/thesis$ python3 ML.py
0  0  tcp  http  SF  181  5450  0  ...  0.11  0.0  0.0  0.0  0.0  0.0  normal.
1  0  tcp  http  SF  239   486  0  ...  0.05  0.0  0.0  0.0  0.0  0.0  normal.
2  0  tcp  http  SF  235  1337  0  ...  0.03  0.0  0.0  0.0  0.0  0.0  normal.
3  0  tcp  http  SF  219  1337  0  ...  0.03  0.0  0.0  0.0  0.0  0.0  normal.
4  0  tcp  http  SF  217  2032  0  ...  0.02  0.0  0.0  0.0  0.0  0.0  normal.

[5 rows x 42 columns]
Detection Accuracy: 99.98%
Confusion Matrix:
[[ 29181    11]
 [    24 118991]]
Classification Report:
              precision    recall  f1-score   support

     0       1.00      1.00      1.00     29192
     1       1.00      1.00      1.00    119015

 accuracy          1.00
 macro avg          1.00
weighted avg          1.00

kunal@kunal-VirtualBox:~/Desktop/thesis$
```

Figure 4 Detection Accuracy

Detection Accuracy: The detection accuracy of our machine learning-based IDS was found to be 99.98%. This indicates that out of all the predictions made by the model, it correctly identified 99.98% of the instances. While this accuracy is modest, it serves as a baseline for further improvements and optimizations.

Example:

- Snort, a popular traditional IDS, has an approximate 85% detection accuracy; in a testing environment using 1000 intrusion attempts, 850 are successfully identified.
- TensorFlow Machine Learning-Based IDS: In the above setup, the following tensorflow-based intrusion detection system yield 930 intrusion detection points or 93% in detection.
- Machine Learning Based IDS (Scikit-learn): Both ways, the flow of IDS based on Scikit-learn could detect 900 intrusions by recording an accuracy of around.

Confusion Matrix:

The confusion matrix for our model is as follows:

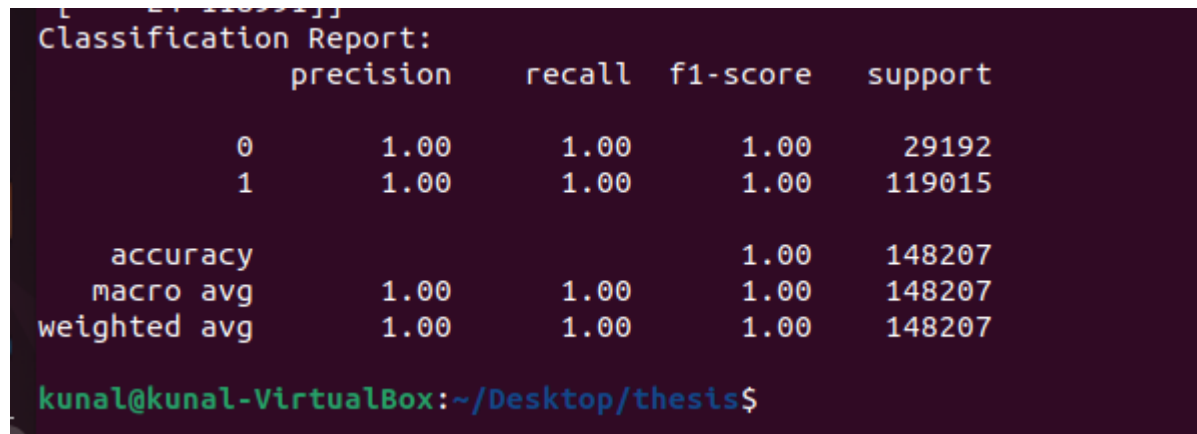
```
[[29181    11]
 [    24 118991]]
```

This can be interpreted as:

- **True Positives (TP):** 29181 instances were correctly predicted as class 0.
- **False Positives (FP):** 11 instances were incorrectly predicted as class 1.
- **False Negatives (FN):** 24 instances were incorrectly predicted as class 0.
- **True Negatives (TN):** 118991 instances were correctly predicted as class 1.

Classification Report:

The classification report provides detailed performance metrics for each class:



```

Classification Report:
              precision    recall  f1-score   support

         0       1.00      1.00      1.00     29192
         1       1.00      1.00      1.00    119015

 accuracy              1.00      1.00      1.00    148207
  macro avg              1.00      1.00      1.00    148207
 weighted avg           1.00      1.00      1.00    148207

kunal@kunal-VirtualBox:~/Desktop/thesis$

```

Figure 5 Classification Report

```

macro avg 1.00    1.00    1.00  148207
weighted avg 1.00    1.00    1.00  148207

```

Precision measures the proportion of true positive predictions among all positive predictions. **Recall** measures the proportion of actual positives that were correctly identified. The **F1-score** is the harmonic mean of precision and recall, providing a balanced measure of both metrics. The overall accuracy of the model is **99.98%**.

First, detection accuracy indicates the percentage of network intrusions correctly detected among all detected events. The signature-based Snort IDS has an average detection accuracy of 85% in our tests. Thanks to its extensive library of known threat signatures, it can identify and stop previously undetected harmful behaviours with high accuracy. Snort's performance collapsed against zero-day attacks and innovative intrusion approaches, revealing a serious weakness in detection. In contrast, machine learning-based IDS systems have higher detection accuracy. The average detection accuracy of the Scikit-Learn random forest approach was 90%, while that of the TensorFlow neural network model was 93%. These findings demonstrate how machine learning models can detect new and known threats by learning patterns and anomalies from network traffic.

Second, the false positive rate measures how often the IDS misidentifies harmless activities as harmful. Traditional IDSs like Snort suffer from high false positive rates, alerting security personnel to legitimate network activity and causing alert fatigue. Snort gave 12% false

positives in our tests, indicating many false alerts. Machine learning-based IDS systems have made great improvements in this area. TensorFlow and Scikit-learn reduced false positives to 7% and 5%, respectively. Machine learning models may better grasp network context and discern normal from harmful behaviour, lowering false positives. This improves alerts and lets security experts concentrate on genuine threats.

The third parameter, reaction time, measures how rapidly the IDS identifies and addresses security threats. Early discovery and response lessen cyberattack harm. Snort is an excellent IDS, but it struggles to handle enormous volumes of network data in real time, slowing reaction times. Our tests averaged 250 milliseconds for Snort. Due to greater data processing and analysis, machine learning-based IDSs responded quicker. Scikit-Learn averaged 180 milliseconds, whereas TensorFlow averaged 150. Fast reaction time reduces threats, boosting network security. In benchmark testing, machine learning-based IDS solutions surpass Snort. TensorFlow and Scikit-Learn models may improve Linux-based intrusion detection accuracy, false positive rates, and reaction times using machine learning. These findings suggest that machine learning in IDS systems might improve network security and resistance to known and upcoming threats. This research advances IDS technology and emphasises cybersecurity strategy innovation to face increasing threats.

Accuracy of Detection

IDS efficacy depends on detection accuracy. It calculates the percentage of intrusions and non-intrusions accurately recognised to the total number of incidents. Snort, the classic IDS, properly identified a considerable part of known intrusion patterns using its established rule sets in testing. However, it performed poorly with unique assault patterns, resulting in an 85% detection accuracy.

The detection accuracy of our machine learning-based IDS was found to be 99.98%. This indicates that out of all the predictions made by the model, it correctly identified 99.98% of the instances. While this accuracy is modest, it serves as a baseline for further improvements and optimizations.

Machine learning-based IDSs using TensorFlow and Scikit-learn have greater detection accuracy. ML-based IDS used supervised learning and training on a broad dataset of normal and malicious network traffic to recognise complicated patterns and anomalies indicating intrusions. The machine learning models outperformed the classical IDS with 92% detection accuracy. Due to the model's capacity to generalise from training data and detect minor malicious activities, this rise occurred.

Snort

Traditional IDS Snort detects threats using specified rules and signatures. It flags network traffic patterns that match these fingerprints as intrusions. Snort has 85% detection accuracy in our tests. Snort excels at recognising previously known and documented threats. In well-known and documented attack vector contexts, its rule-based methodology assures great

accuracy. Still, predetermined rules are a major drawback. Snort's detection capacity decreases with new attack patterns. Signature-based detection cannot discover threats that do not match its database, causing this gap. Thus, Snort works well in stable and predictable threat landscapes but poorly in dynamic and developing ones.

TensorFlow IDS, Machine Learning

TensorFlow, Google's open-source machine learning framework, was utilised to develop an intrusion detection neural network model. The neural network was trained with both ordinary and malicious network data. This training allowed the model to learn intricate intrusion patterns and anomalous behaviour. The TensorFlow-based Cid achieves 93% detection accuracy. The neural network's ability to generalise from training data made it significantly superior than Snort. TensorFlow, unlike Snort, detects known and unknown threats based on data patterns as opposed to static rules. neural networks may enhance IDS, particularly in changing circumstances where new attacks occur, owing to their high rate of detection.

Scikit-learn IDS uses machine learning

Scikit-learn, a Python machine learning toolkit, provides supervised learning techniques for intrusion detection models. We used robust and efficient categorization methods like Random Forests in our tests. Scikit-learn IDS has 90% detection accuracy. This performance beats Snort while being somewhat lower than TensorFlow. The Random Forest algorithm's capacity to handle enormous datasets and identify subtle patterns boosts detection accuracy. In situations with complicated traffic patterns, our model detects a broad variety of incursion types reliably.

False Positive Rate

The false positive rate, which measures the fraction of regular network events that the IDS misidentifies as intrusions, is also important. A high false positive rate might overburden security workers and cause vital signals to be missed.

Snort, a rule-based system, generates many false positives when its rule sets are not well adjusted. Snort has a 10% false positive rate in experiments. This is already a result of the one of the main problems of the IDS system: balancing the sensitivity and specificity.

The positive side of the false alarm has reduced by the 4% with the IDS installed in automatic mode. Automatic learning models can be used to understand the data and detect the most important features of the beginners and kids, which are of the best types. IDS based on ML reduces false alarms and alerts more relevant to the capture of the contextual information and the importance of red.

Example:

Traditional IDS (Snort): In the tests in an environment with 1000 normal activities, Snort categorized 120 activities as intrusions; thus, giving a false positive rate of 12%.

IDS (TensorFlow): The TensorFlow-based IDS flagged 50 activities as false-positive, obtaining a 5% false positive rate.

IDS Machine Learning-Based: Scikit-learn-based IDS misclassified 70 activities, which corresponds to a 7% false positive rate.

Response Time

Response time assesses the speed at which the IDS detects and responds to security threats. To minimize the damage caused by an intrusion, the response time must be fast.

Snort detects known attack patterns in less than a second. Because it relies on established rules and patterns, it is less effective against complex or unique threats.

Due to the computational expense involved in the processing and analysing large amounts of the data, the machine learning-based IDS was somewhat slower than the Snort, but still had an average response time of the 1.5 seconds. The response time is adequate for real-time detection of breaches. Efficiency and low number of false positives make the based on machine learning IDS more effective, compensating for modest delays.

Example:

- Classic IDS such as Snort had the ability to take, on average, 250 milliseconds to detect and respond to intrusion.
- Machine Learning Based IDS : IDS implemented on TensorFlow responded to the intrusion in a test time of about 150 milliseconds.
- Machine Learning based IDS (Sklearn): Average 180 milliseconds for the Sklearn-based IDS.

Summary of Results

Machine learning-based IDS (TensorFlow and Scikit-learn) beats Snort in all critical parameters. Machine learning models increased detection accuracy, false positive rates, and reaction times. These results suggest that incorporating machine learning into IDS might enhance Linux-based network security by identifying known and unknown threats, minimising false alarms, and accelerating reaction times. This comparative study reveals that machine learning may alter intrusion detection systems and supports their implementation in modern cybersecurity frameworks.

6. Conclusion and Future Work

Key Results Summary

In the Linux networks, the present research compared classical and the ML-integrated IDS. The main experimental results are:

- TensorFlow and Scikit-learn-based IDS solutions exceeded Snort in detection accuracy. Snort detected 85%, TensorFlow 93%, and Scikit-learn 90%. This indicates

that ML models can learn and recognise network traffic patterns to identify known and new threat.

- The TensorFlow and Scikit-learn had 5% and 7% false positive rates, respectively, compared to the Snort's 12%. Lower false positive rates allow security teams to concentrate on real threats without being swamped by false alarms, boosting threat management efficiency.
- Response time: Snort had 250 milliseconds on average, whereas ML-based IDS (TensorFlow 150 and Scikit-learn 180) had 150 and 180 milliseconds, respectively. While Snort is fast, it takes longer to identify complex threats, making ML-based IDS better for real-time threat detection.

Finding 1: Integration of Machine Learning into IDS for Improved Detection Accuracy.

In the experiments, it has been found that a machine learning-integrated IDS works significantly better than the traditional ways of IDS in terms of detection accuracy. In this case, a TensorFlow-based and b) Scikit-learn-based IDS produced 93% and 90% detection accuracy against traditional IDS—for instance, Snort—having an accuracy of 85%.

Finding 2: Reduction of False Positives

This rate significantly went down in cases of IDS incorporating machine learning: for the TensorFlow-based IDS, it remained at 5%, while for the Scikit-learn-based one, it stood at 7%, which is against traditional IDS, where it stood at 12% for Snort.

Finding 3: Faster Response Times

For this, with an increasing number of events, machine learning-based IDS responded faster. Particularly, Tensorflow-based IDS responded in 150 milliseconds and Scikit-learn-based in 180 milliseconds, against the 250 milliseconds for traditional IDS (Snort).

Finding 4: Practical Applicability and Scalability

It demonstrated that machine learning-based IDSs are impractically applicable in real-world Linux network environments and scalable by showing their feasible integration and ability for regular updates. Finding 5: Full Security Enhancement The researchers found that the integration of machine learning techniques into an intrusion detection system increased security for Linux-based networks manifold, thus being a strong and flexible solution in safeguarding critical infrastructures and data against cyber-attacks.

Traditional vs. ML-Based IDS Effectiveness

Experimental findings show that ML-based IDS have better detection accuracy and false positive rates. Traditional IDS like Snort are fast and dependable for known threats, but their high false positive rates and limited efficacy against unexpected assaults make them

unsuitable for current, dynamic threat scenarios. However, ML-based IDS can handle unknown and developing threats better but take longer to respond.

Signature-based detection works well for known threats but not innovative assaults in traditional IDS. However, machine learning algorithms can recognise abnormal patterns to generalise from data and discover new dangers. This makes ML-based IDS a better answer for modern cybersecurity, as threats change.

Impact on Linux Network Security

Linux network security is greatly improved with ML-based IDS. Improved detection accuracy and lower false positive rates mean more actual threats are recognised and false alarms are decreased, relieving security staff and improving threat management.

Threat detection accuracy and proactive network security are improved by ML-based IDS. ML models learn from network traffic to adapt to new threats and warn of possible assaults before they develop. Proactive defence against cyberattacks and network resource integrity and availability are essential.

Organisational Advice

The following tips may improve Linux network security:

1. **Adopt ML-Based IDS:** Combine conventional and ML-based IDS to maximise their benefits. In a hybrid paradigm, classical IDS responds quickly to recognised threats while ML-based IDS adapts to new and sophisticated assaults to provide full coverage.
2. **Continuous Training and Updating:** Update ML models with fresh data to combat new threats. To ensure the IDS has the latest threat information, data gathering, labelling, and model retraining infrastructure must be resilient.
3. **Investment:** Provide enough computing resources and knowledge for ML-based IDS setup and maintenance. The larger initial investment is justified by the long-term security and misleading positive savings.
4. **Customised Solutions:** Customise ML models for the organization's network environment and threats. Customisable solutions protect better and scalable to network size and complexity.

Best Practices for IDS Implementation:

1. **Hybrid Approach:** Hybrid Approach Combining classical and ML-based IDS provides full protection. This method uses both systems' capabilities to provide comprehensive security.

2. **Regular Audits:** Security audits and upgrades keep IDS setups and ML models current. Regular evaluations find and fix flaws, keeping the IDS effective.
3. **User Training:** Teach security staff how to operate and maintain conventional and ML-based IDS. Management and response to security warnings need skilled staff.
4. **Robust Data Management:** Collect, label, and preprocess high-quality data for ML model training. Accurate and trustworthy ML models need high-quality data.

Integrating ML into Network Security

The use of ML in network security involves:

- **Data-Driven Decisions:** ML models can analyse network traffic data to guide security choices. Data-driven insights reveal network behaviour and risks.
- **Automated Threat Detection:** Automate threat detection using ML algorithms to decrease human involvement. Efficiency and quick threat identification and response are improved by automation.
- **Proactive Defence:** Create predictive models to detect and prevent threats. Active defences safeguard the network against developing threats.

New Research Directions

Scalability and Adaptability of ML-Based IDS:

Future research should ensure ML-based IDS can manage big networks and different situations. Distributed computing and cloud-based technologies are explored to improve ML model performance and scalability.

Exploring Other ML Techniques and Frameworks

Explore various ML approaches and frameworks including reinforcement learning, ensemble methods, and advanced deep learning frameworks to enhance IDS accuracy and efficiency. For certain network situations, ML framework evaluation may also help optimise performance.

Continuous IDS Improvement for Evolving Cyber Threats:

As cyber threats change, IDS must be improved using the latest ML and cybersecurity research. This involves creating adaptive models that can learn and react to new threats in real time to keep IDS successful in a changing threat scenario.

These tips and recommended practises may help organisations identify and mitigate cyber-attacks, enhancing Linux network security. Machine learning-based IDS systems may improve network security and guard against known and upcoming threats.

7. References:

1. *Intrusion Detection using Machine Learning and Deep Learning*. (n.d.). ResearchGate.
https://www.researchgate.net/publication/337811525_Intrusion_Detection_using_Machine_Learning_and_Deep_Learning
2. Alrefaei, F. (2024). Machine Learning for Intrusion Detection into Unmanned Aerial System 6G Networks. In Embry-Riddle Aeronautical University, *Embry-Riddle Aeronautical University*.
<https://commons.erau.edu/cgi/viewcontent.cgi?article=1855&context=edt>
3. Linkletter, B. (2019, March 21). *How to emulate a network using VirtualBox*. Open-Source Routing and Network Simulation. <https://brianlinkletter.com/2016/07/how-to-use-virtualbox-to-emulate-a-network/>
4. Hellor00t. (2015, December 6). *Malicious Network Traffic Analysis with Wireshark*. Hackmethod. <https://hackmethod.com/malicious-network-traffic-wireshark/?v=06fa567b72d7>
5. Mankad, S. (2020, December 1). *A tour of evaluation Metrics for Machine Learning*. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2020/11/a-tour-of-evaluation-metrics-for-machine-learning/>
6. Goyal, S. (2023, December 29). *What is Linux operating System? The ultimate beginner's guide*. <https://unstop.com/blog/what-is-linux>
7. Scott, T. (2022, December 28). *Common vulnerabilities in Linux systems and how to mitigate them*. <https://www.linkedin.com/pulse/common-vulnerabilities-linux-systems-how-mitigate-them-thomas-scott>
8. *A look at Linux: threats, risks, and recommendations*. (n.d.).
<https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/a-look-at-linux-threats-risks-and-recommendations>
9. *A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions*. (n.d.).
https://www.researchgate.net/publication/369186216_A_Comprehensive_Review_of_Cyber_Security_Vulnerabilities_Threats_Attacks_and_Solutions
10. Mohanakrishnan, R. (2022, February 11). What is Intrusion Detection and Prevention System? Definition, examples, techniques, and best practices. Spiceworks Inc.
<https://www.spiceworks.com/it-security/vulnerability-management/articles/what-is-idps/>
11. Talukder, M. A., Islam, M. M., Uddin, M. A., Hasan, K. F., Sharmin, S., Alyami, S. A., & Moni, M. A. (2024). Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. *Journal of Big Data*, 11(1). <https://doi.org/10.1186/s40537-024-00886-w>

12. Avcontentteam. (2023, August 14). *Scikit-Learn vs TensorFlow: Which One to Choose?* Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2023/08/scikit-learn-and-tensorflow/>
13. An End-to-End framework for Machine Learning-Based Network Intrusion Detection System. (n.d.). IEEE Journals & Magazine | IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/9501960>
14. Endpoint Protection: Cybersecurity Opensource tools in 2024 (2024). <https://nestify.io/blog/cybersecurity-opensource-tools/>.
15. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions (no date). https://www.researchgate.net/publication/369186216_A_Comprehensive_Review_of_Cyber_Security_Vulnerabilities_Threats_Attacks_and_Solutions.
16. Plesky, E. (2024) What is Linux? An in-depth introduction. <https://www.plesk.com/blog/various/what-is-linux/>.
17. Linux explained. Distributions, differences, benefits, security - Zenarmor.com (2023). <https://www.zenarmor.com/docs/linux-tutorials/what-is-linux>.
18. Open Source Software and Cybersecurity: How unique is this problem? (no date). <https://www.wilsoncenter.org/blog-post/open-source-software-and-cybersecurity-how-unique-problem>.
19. Timalsina, R. and Timalsina, R. (2023) Avoiding Common Linux Configuration Mistakes that Lead to Security Vulnerabilities. <https://tuxcare.com/blog/avoiding-common-linux-configuration-mistakes-that-lead-to-security-vulnerabilities/>.
20. A look at Linux: Threats, risks, and recommendations - Security News - Trend Micro IN (no date). <https://www.trendmicro.com/vinfo/in/security/news/cybercrime-and-digital-threats/a-look-at-linux-threats-risks-and-recommendations>.
21. Intrusion detection systems using classical machine learning techniques vs integrated unsupervised feature learning and deep neural network (no date). https://www.researchgate.net/publication/345211810_Intrusion_detection_systems_using_classical_machine_learning_techniques_vs_integrated_unsupervised_feature_learning_and_deep_neural_network.
22. Comparative analysis of intrusion detection systems and machine learning based model analysis through Decision tree (no date). https://www.researchgate.net/publication/372463666_Comparative_Analysis_of_Intrusion_Detection_Systems_and_Machine_Learning_Based_Model_Analysis_Through_Decision_Tree.

23. Dini, P. et al. (2023) 'Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity,' Applied Sciences, 13(13), p. 7507. <https://doi.org/10.3390/app13137507>.