Enhanced Intelligent Data Security Model for Monitoring Cloud Computing Infrastructure Using Machine Learning Algorithm

N. Michael[1*], G. James[2], F. Onuodu[3], U. Okengwu[3]

[1]Department of Mathematics and Computer Science, Ritman University, Ikot Ekpene, Nigeria.
[2]Department of Computing, Topfaith University, Mkpatak, Nigeria.
[3]Department of Computer Science, University of Port Harcourt, Nigeria.

ABSTRACT: Cloud computing has transformed data management with its scalability and cost-effectiveness, but it also introduces significant security risks, including data breaches, unauthorized access, and malicious attacks. Traditional security approaches often fail to detect sophisticated attacks, highlighting the need for intelligent systems to learn and adapt to evolving threats. This study presents a novel data security virtualization model that employs the Random Forest algorithm to enhance the security of cloud computing infrastructures. The model combines network traffic analysis, system log analysis, and virtual machine monitoring to detect and respond to security threats. The study aims to improve the accuracy and efficiency of threat detection and response using machine learning techniques. The proposed model was evaluated using a comprehensive dataset and showed promising results, achieving high accuracy in detecting malicious activity with a precision of 0.99 and recall of 0.99. This research contributes to proactive security measures in cloud computing by integrating advanced machine learning methodologies, providing a novel solution that addresses the limitations of traditional security approaches.

KEYWORDS: Cloud Computing, Data Management, Data Security, Virtualization, Blockchain, Virtualization Architecture.

[Received Jan. 3, 2025; Revised Mar. 1, 2025; Accepted Mar. 29, 2025] Print ISSN: 0189-9546 | Online ISSN: 2437-2110

I. INTRODUCTION

The advent of cloud computing has revolutionized the way organizations store, process, and manage data. Cloud computing offers numerous benefits, including scalability, flexibility, and cost-effectiveness, making it an attractive option for businesses of all sizes (Mell and Grance, 2009). However, this shift to cloud computing also introduces new security challenges, including data breaches, unauthorized access, and malicious attacks (Kumar et. Al., 2020). Traditional security approaches often fall short in detecting sophisticated attacks, highlighting the need for intelligent systems that can learn and adapt to evolving threats (Bhattacharya et. Al., 2019). Virtualization technology has been widely adopted in cloud computing to improve resource utilization and scalability (Chapman and Smith, 2001). However, virtualization also introduces new security risks, such as virtual machine (VM) escape attacks and hypervisor vulnerabilities (Xenakis et. Al., 2019).

Intelligent systems-enhanced data security virtualization models have shown promise in addressing these challenges (Singh et. Al., 2019). These models utilize machine learning algorithms to analyze network traffic, system logs, and other data sources to detect potential security threats (Bhattacharya et. Al., 2019). The Random Forest algorithm, in particular, has been widely used in intrusion detection systems due to its ability to handle large datasets and complex feature interactions (Liu et. Al. 2019).

This research proposes a novel intelligent system-enhanced data security virtualization model that leverages the Random Forest algorithm to monitor cloud computing infrastructure. The proposed model shall utilize a combination of network traffic analysis, system log analysis, and virtual machine monitoring to detect potential security threats. The Random Forest algorithm is used to classify network traffic and system logs as either normal or malicious, while virtual machine monitoring is used to detect VM escape attacks and hypervisor vulnerabilities. Cloud computing has undergone significant evolution since its inception in the 1960s (Mell and Grance, 2009). Initially, cloud computing was primarily used for data storage and processing, but it has since expanded to include a wide range of services, including infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) (Chapman and Smith, 2001). This evolution has been driven by advances in virtualization technology, distributed computing, and network infrastructure.

Despite its many benefits, cloud computing introduces several security challenges, including data breaches, unauthorized access, and malicious attacks (Ryusei et. Al., 2022). These challenges are exacerbated by the multi-tenancy nature of cloud computing, where multiple users share the same physical infrastructure (Anuruddha et. al., 2022). Additionally, cloud computing's reliance on virtualization technology introduces new security risks, such as VM escape attacks and hypervisor vulnerabilities (Arora and Parashar, 2013). Intelligent systems, including machine learning and

*Corresponding author: g.james@topfaith.edu.ng

doi: http://dx.doi.org/10.63746/njtd.v22i3.3261

artificial intelligence, have shown promise in addressing the security challenges associated with cloud computing (Kumar et. al., 2020). These systems can analyze network traffic, logs, and other data sources to detect potential security threats (Kumar et. al., 2020). The Random Forest algorithm, in particular, has been widely used in intrusion detection systems due to its ability to handle large datasets and complex feature interactions (Arthur, 2006; Barker et al., 2009).

Cloud computing, which provides scalable and flexible infrastructure solutions, has become a cornerstone for many enterprises in the age of rapid digital change (Baroncelli et. al., 2010; BBC News, 2013). However, the move to cloud-based services has also brought up serious security issues. Due to the dynamic and complicated nature of cloud settings, traditional data security procedures frequently fall short, increasing vulnerability and possibly resulting in data breaches (Bengio et, al., 2013; Bengio et, al., 2015; Borders et. al., 2009; Boyd et, al., 1987). The main issue is the absence of an efficient and intelligent system that can keep an eye on and secure cloud computing infrastructures constantly (Chellappa, 2013). Current security methods frequently lack proactive threat detection and mitigation because they are reactive (Ciresan et. al., 2012; Cramer, 2002; Deng and Yu, 2014; Gellman, 2009). Furthermore, the identification of possible security threats and abnormal activity is made more difficult by the massive amount of data created in cloud settings (Gurav et. al., 2010; Harnad, 2008; Han et. al., 2014; Han et. al., 2015).

To improve data security virtualization, a strong, intelligent system that makes use of cutting-edge machine learning methods like the Random Forest Algorithm is essential. Real-time monitoring, threat identification, and adaptive responses to changing security risks in cloud infrastructures should all be possible with this approach (Hao et. al., 2015; Ikrohn et. al., 2007; Jin et. al., 2015; Langley, 2011). The objective is to develop a thorough security model that guarantees the availability and integrity of cloud services in addition to safeguarding sensitive data (Liang and Zhang, 2015; Luo et. al., 2011; Marblestone et. al., 2016; Mollah and Vasilakos, 2017). The motivation for this research is to develop an intelligent system-enhanced data security virtualization model that can effectively detect and prevent security threats in cloud computing infrastructure (Olshausen, 1996; Paladi and Michalas, 2016). The proposed model will utilize a combination of network traffic analysis, system log analysis, and virtual machine monitoring to detect potential security threats. The Random Forest algorithm will be used to classify network traffic and system logs as either normal or malicious, while virtual machine monitoring will be used to detect VM escape attacks and hypervisor vulnerabilities.

This research aids in the creation of cutting-edge security protocols designed especially for cloud computing infrastructures. It offers a cutting-edge method for handling contemporary security issues by fusing the Random Forest Algorithm with intelligent systems. Implementing a real-time model that can detect and respond to threats enhances cloud environments' overall security posture, reducing the risk of data breaches and other security incidents. By enhancing the virtualization model for data security, the study helps ensure the confidentiality, integrity, and availability of data stored in the cloud (Olshausen, 1996; Paladi and Michalas, 2016; Sainath et. al., 2013; Schulz and Behnke, 2012). This is particularly important for organizations handling sensitive information, such as financial data, personal records, and proprietary information. However, the power of this research lies in its potential to transform the way cloud computing infrastructures are secured, offering a sophisticated, proactive, and scalable solution to contemporary security challenges.

II. THEORETICAL ANALYSIS

This study looked at the work of Ryusei et al. (2022). The system was developed by Ryusei et al. (2022) and was meant to share the signatures of suspected malware files using blockchain technology. The system was able to share the signatures of suspected files between users, allowing them to rapidly respond to increasing malware threats (Sgandurra and Lupu, 2016; Shi et. al., 2011; Shiraz et. al. 2013). Further, it was able to improve the accuracy of detection and removal of malware by utilizing signatures recorded by the blockchain. A phenomenal technique of the Existing System was the application of blockchain technology in the detection of malware files (Si et. al., 2013; Vaezpour et. al., 2016; Xing and Zhan, 2012). This system was not able to explore its full efficiency due to the lack of a PerpSearch for guilty agents that perpetuate online software piracy and the problems of detection latencies and inaccuracies due to overfitting were also identified in the model. Figure 1 presents the Ryusei et al. (2022) architecture.

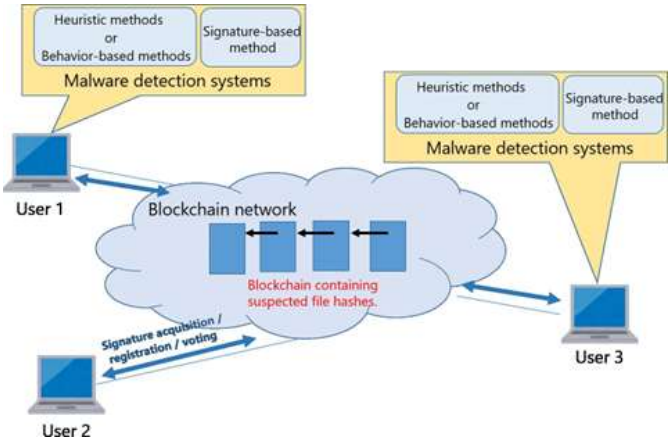

Figure 1: Architecture of the Existing System (Source: Ryusei et. al., 2022)

In summary, their blockchain-integrated approach did not fully offer a promising alternative to conventional malware detection, reducing dependency on centralized antivirus providers and enhancing the robustness of malware defenses. However, while the system successfully addresses some key challenges in malware detection, it lacks real-world testing and scalability solutions that are needed to fully validate its effectiveness and feasibility in practical applications. This study intends to enhance the work of Ryusei et al., (2022) with

an improved virtualization model for data fraud detection using a robust Machine learning technique- Random Forest

III. MATERIALS AND METHODS/METHODOLOGY/EXPERIMENTAL PROCEDURE

This study proposes an intelligent system-enhanced data security virtualization model for cloud computing infrastructure. The model utilizes a combination of network traffic analysis, system log analysis, and virtual machine monitoring to detect potential security threats. A VMware virtual machine monitoring tool was used to monitor and track data to form the dataset. These datasets were pre-processed using feature extraction and normalization techniques. The features in the datasets were selected using the recursive feature elimination techniques. Upon the completion of the feature engineering, the Random Forest algorithm was used to classify network traffic, system logs, and virtual machine monitoring data as either normal or malicious, and the accuracy, precision, recall, F1-score, and ROC-AUC performance metrics were used to cross-validate and evaluate the model's performance on unseen data.

.

A. System Architecture

Figure 2 presents the architecture of the proposed system.

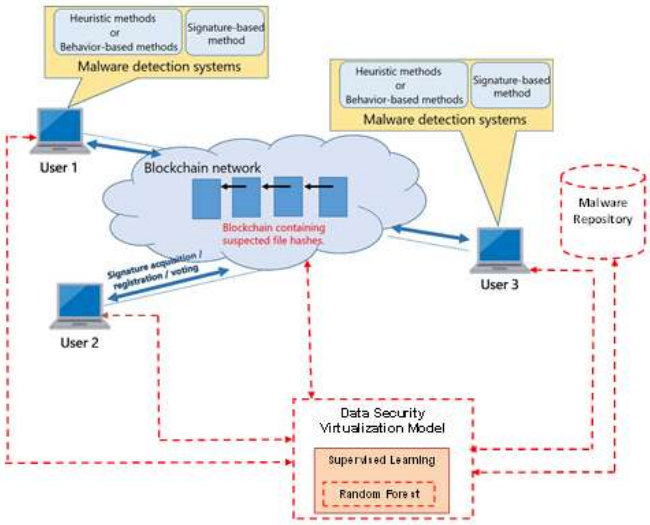

Figure 2: Architecture of the Proposed System

B. Analytical Presentation of Components of the Proposed System

The following are components of the Proposed System

(I) Malware Detection System

This component detects any suspected malware embedded by guilty agents in the Signature File Sharing System. Figure 3 shows the overall architecture of the system and how different components interact.

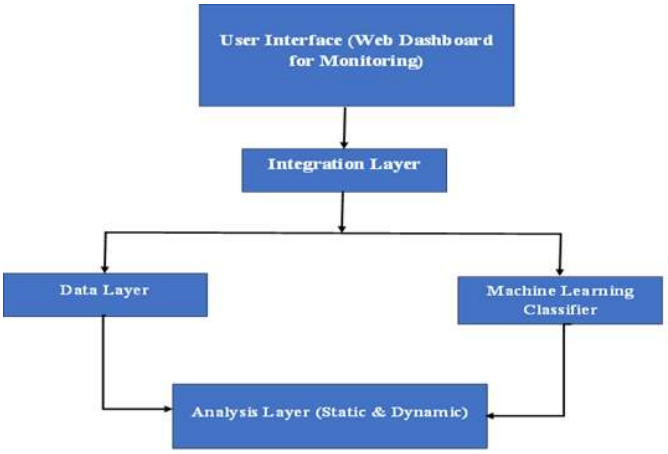

Figure 3: Architecture of malware detection system

In this work, malware detection uses a combination of static and dynamic analysis, which helps improve machine learning-based malware classification.

(a) User Interface (Web Dashboard for Monitoring)

The web dashboard for monitoring is a crucial component that serves as the user interface where users can interact with the system, visualize data, and manage security insights (James et. al., 2024a; James et. al., 2024b). The dashboard is designed to provide real-time information about malware detection, system health, and overall security posture. Table 1 illustrates visually; the key components of the conceptual representation of the web dashboard and Figure 4 illustrates the mockup of the web dashboard visually.

Table 1: Key components of the conceptual representation of the web dashboard

Security Monitoring Dashboard	
User: [Username] [Logout Button]	
Overview	Real-Time Monitoring
Total Files Scanned: **1200**	**Current Scans:** 3
Malware Detected: **45**	**Active Threat:** 2
False Positives: **3**	**Last Scan:** 5 mins ago
Last Update: **12:45 PM**	
Historical Data	**Recent Alerts**
[Graph: Detection Trends]	[Alert 1] Malware detected: abc123 (Critical) [Alert 2] File hash registered: def456 (Low)
File Hash Query: **[_____]**	**[Search Button]**
Result: **[No previous dete]**	

Table 2: Mockup of what the web dashboard

Malware Detection Dashboard			
[Real-Time Status]	[Alerts]	[User Management]	[Logout]

[Summary of Recent Detections]

File Name	File Hash	Status
example.exe	abc123hashvalue	Quarantined
malicious_file.dll	def456hashvalue	Deleted
benign_file.txt	ghi789hashvalue	Clean

[Analysis Reports]

[Static Analysis Results]

File Name	Features Extracted	Result
example.exe	Size, Signature, Headers	Malware
benign_file.txt	Size, Signature	Benign

[Dynamic Analysis Results]

File Name	Behaviour Detected	Result
example.exe	Network Calls, File Access	Malware

[Blockchain Status]

Total Entries:	1000
Last Update:	2024-11-04

The web dashboard is a crucial component of the improved data security virtualization model, providing users with a comprehensive view of malware detection activities and system performance. With real-time monitoring and analysis capabilities, the dashboard empowers users to make informed decisions regarding their cloud computing infrastructure's security.

(b) Integration Layer

The integration layer is essential in the proposed security model for linking malware detection engines (both static and dynamic) with machine learning models and the monitoring dashboard. This layer consolidates data from different sources and provides a unified interface for performing malware analysis and classification. It ensures data flow between components, manages real-time communications, and orchestrates interactions across the system. The message broker within the integration layer is responsible for coordinating communication between different components in the security system, ensuring that data flows smoothly and in real time across static and dynamic analysis engines, the machine learning classifier, the blockchain, and the monitoring dashboard. Below are the details of the message broker, including its roles, practical design, and example configurations.

(c) Data Layer

For this work, the Kafka system was used to implement the message broker. Apache Kafka is a distributed event-streaming platform, making it suitable for handling high-throughput data flows and providing persistence and fault tolerance. The Kafka topics are created to organize communication channels between the components as follows:

```
# Create topics for different data flows
kafka-topics.sh --create --topic static-analysis-results --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
kafka-topics.sh --create --topic dynamic-analysis-results --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
kafka-topics.sh --create --topic classification-results --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
kafka-topics.sh --create --topic blockchain-registration --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
kafka-topics.sh --create --topic dashboard-updates --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
```

Figure 4: Apache Kafka message broker system

(d) Machine Learning Classifier

Aside from the data layer, the next step we took was designing the Machine Learning Classifier. At this point, the machine learning classifier is a critical component of the improved data security virtualization model, tasked with identifying malicious files by analyzing them through a combination of static and dynamic features. The classifier is integrated into the malware detection pipeline to distinguish between benign and malicious files, improving detection accuracy while reducing false positives and false negatives. The classifier begins by extracting static features from files. These features include metadata such as file size, hash values, file type, and embedded code snippets. Static analysis does not execute the file, making it fast and suitable for initial assessments. This includes file size, creation/modification dates, and file type. The strings were pulled out from binary files to find potentially suspicious keywords, URLs, or embedded scripts and extract fields like imported libraries and API calls, often targeted by malware using the code:

```
def extract_file_metadata(file_path): file_info = {} file_info['file_name'] = os.path.basename(file_path)
file_info['file_size'] = os.path.getsize(file_path)
file_info['creation_time'] = os.path.getctime(file_path)
file_info['modification_time'] = os.path.getmtime(file_path) return file_info
```

Using a hypothetical file, this code-generated outputs;

```
file_name: sample_malicious_file.exe
file_size: 51234 bytes
creation_time: 1677845720.0
modification_time: 1677845750.0
md5: a5f4c3c3456f78123456789abcdef012
```

sha1: b4e5c2d4f5678123456789abcdef0123456789
sha256:
c8d3f5a6b789c0123456789abcdef0123456789abcdef0
123456789abcdef0123
strings: ['cmd.exe', 'C:\\windows\\system32',
'malicious_function', 'url_to_bad_site.com',
'password']
entry_point: 0x1000
image_base: 0x400000
imported_dlls: ['KERNEL32.dll', 'USER32.dll']
imported_symbols: ['CreateFileA', 'ReadFile', 'WriteFile', 'CloseHandle'].

These extracted features are then served as input for the machine learning model, which uses them to classify the file as benign or malicious based on its training. This detailed static feature extraction enhances the classifier's understanding of each file, particularly when combined with dynamic analysis in cases where static indicators alone are insufficient. Files that cannot be conclusively classified through static analysis are subjected to dynamic analysis. Here, the file is executed in a controlled sandbox environment, where API calls behaviours, system modifications behaviours, network connections behaviours, and memory changes behaviours are observed. These behaviours provide the classifier with contextual insights, which are essential for identifying sophisticated malware. The learning curves for different hyperparameters were implemented which makes it possible to identify the best-performing configuration, balancing bias, and variance to improve the model's generalization. First, the max_depth (Maximum depth of each decision tree) of 50 was used to prevent overfitting. Then the n_estimators, min_samples_split, min_samples_leaf, max_features, bootstrap, oob_score, criterion, random_state, and class_weight were equally used to compare to determine the most essential for identifying sophisticated malware.

(e) Analysis Layer (Static & Dynamic)

The Analysis Layer in a malware detection model for cloud infrastructure consists of both Static and Dynamic Analysis. This layer's primary role is to comprehensively analyze files, detect potential malware, and produce features that enhance the accuracy of machine learning classifiers. Static analysis involves examining a file without executing it by capturing details such as file type, size, creation/modification timestamps, and permissions. Metadata often hints at unusual patterns that can be indicative of malware (e.g., unexpectedly large files or unusual file types).

(II) Users

Three components represent this process that involves three end-user terminals for communicating with the Signature File Sharing System.

(a) The Input and Output Design

The input design of the proposed system encompasses the system initialization platform, the new user register platform, the login platform, and the data input platform. The function of the new user register platform is to enable new users to input registration details to be allocated login details such as username and password. The function of the login platform is to enable a registered user to login to the system. The function of the data input platform is to enable system scans for malware detection and remediation. Figure 5 shows the input design of the proposed system.

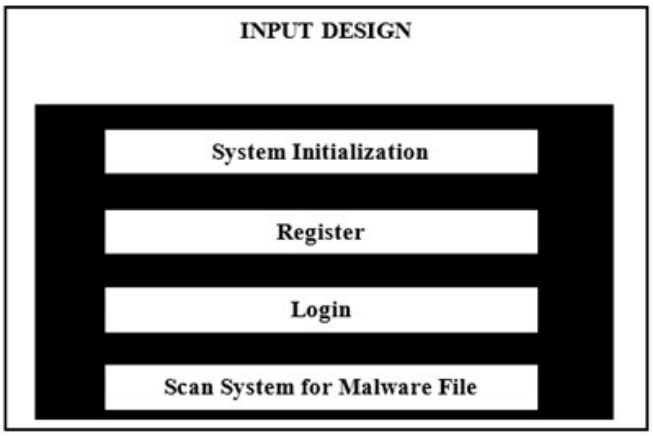

Figure 5: High-Level Input Design of the Proposed System

The output design of the proposed consists of a data display tool that gives out the result of the entered input, once it is processed. The output design of the proposed system includes the input platform, the operation platform, the control platform, and the system scan platform. Figure 6 shows the high-level output design of the proposed system.

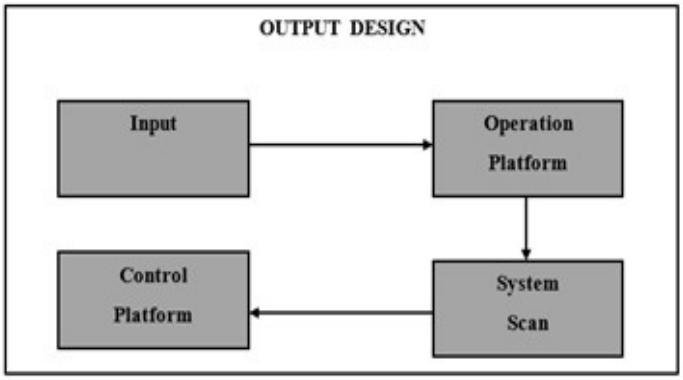

Figure 6: High-Level Output Design of the Proposed System

(b) The Rule-base Design

The rule-based design of the proposed system uses a set of prewritten rules to make decisions and solve problems. Developers create rules based on human expert knowledge that enable the system to process input data

and produce a result. The rule-based design of the proposed system requires only small amounts of simple data to perform tasks and repetitive processes. This makes them easy for developers to create, use, and debug. Figure 7 shows the rule-based design of the proposed system.

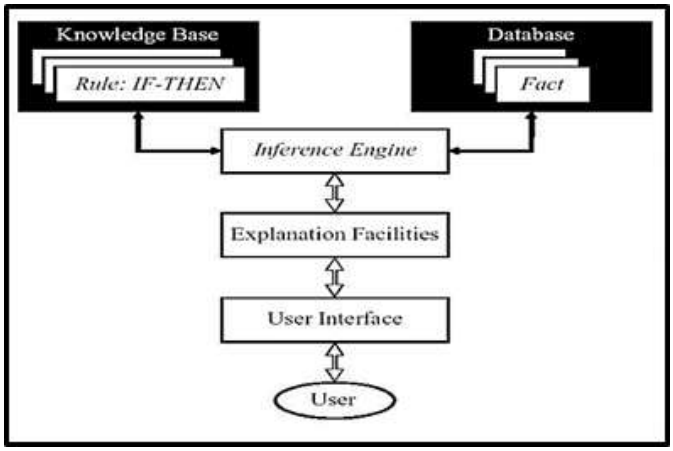

Figure 7: Rule-base Design of the Proposed System

(c) Blockchain Network

This component represents a distributed ledger system that contains suspected file hashes. This is part of the Cloud Computing Infrastructure, in which malicious files can be shared or distributed. Figure 8 shows the Blockchain address registration.

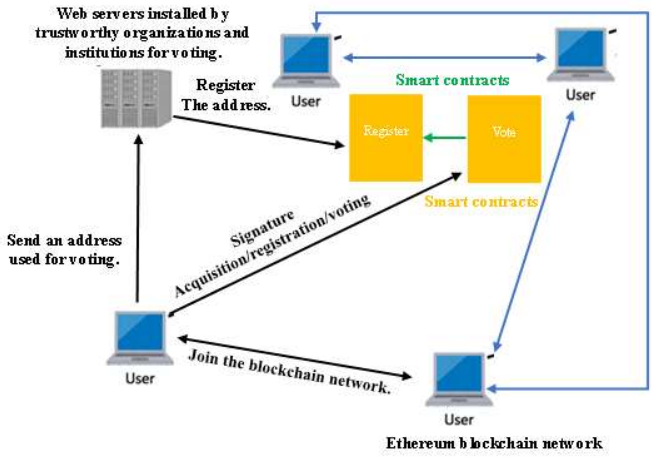

Figure 8: Rule-base Design of the Proposed System

Here we describe data such as file hash values and the number of votes to be stored on the blockchain. Data stored on the blockchain can be represented as a record, and its details are shown in Figure 8. As can be seen from Figure 8, the record is represented by the following five elements:

i. Suspected file hash value
ii. Number of votes for "malicious"
iii. Number of votes for "benign"
iv. Addresses of users who voted "malicious"
v. Addresses of users who voted "benign"

The numbers of votes for "malicious" and "benign" are used to calculate the degree of maliciousness in the elimination decision formula (section IV-E) to determine whether to remove the file. The recording of user addresses prevents the same user from illegally voting more than once. Here, the user address is not an IP address but the address used on the blockchain, such as "0xca35b7d915458ef540ade6068dfe2f44e8fa-733c".

When the hash value of the downloaded file exists on the blockchain, the user's detection system determines whether to remove the file based on the maliciousness degree given by the elimination decision formula. That is, when equation (1) is satisfied, the file is not deleted, and when equation (2) is satisfied, the file is deleted.

$$M_d \leq M_t, \quad (1)$$

$$M_d > M_t, \quad (2)$$

$$0 \leq M_d \leq 1.$$

When Vm + Vb ≥Vt: The user's system uses only the voting result on the blockchain and calculates the malicious- ness degree with equation (3).

$$M_d = \frac{V_m}{V_m + V_b} \quad (3)$$

When Vm + Vb < Vt: The user's system calculates maliciousness degree with expression (4), the results of voting on the blockchain, and its malware detection results by heuristic or behaviour-based methods. Here, it is assumed that the malware detection system outputs 1 when the file is malware and 0 when it is benign. That is, Dr ∈ {0, 1}.

$$M_d = \frac{V_m}{V_m + V_b} + R_v + D_r + R_s \quad (4)$$

Where Rv and Rs are defined by the following expressions:

$$R_v = \frac{V_m + V_b}{V_t} \quad (5)$$

$$R_s = 1 - Rv \quad (6)$$

(d) Data Security Virtualization Model

This component represents an automated method of recognizing a person based on a physiological or behavioural characteristic using logistic regression. Among the features measured are face, fingerprints, hand geometry, handwriting, iris, retinal, vein, and voice. Biometric data are separate and distinct from personal information.

(e) Malware Repository/Data Collection

This section discusses the means of collecting all the information for training and testing the Proposed System and managing it in a way that maximizes the speed and comprehensiveness with which critical information can be extracted, analyzed, and used (Gabriel et. al., 2024; James et. al., 2024c). A total of 10000 datasets were collected making up of collections of network traffic, system logs,

and virtual machine monitoring data, with corresponding labels indicating whether the data was normal or malicious. The collection process involved real-world network traffic, system logs, and virtual machine monitoring data. The data was generated using various tools and techniques, such as TCPdump Network traffic generators to simulate network traffic, Logstash System log generators to simulate system logs, and VMware Virtual machine monitoring tools to simulate virtual machine monitoring data.

The data was labeled as normal or malicious based on the characteristics of the data, such as unusual network traffic patterns, suspicious system log entries, or anomalous virtual machine behaviour respectively. Table 3 shows the analytical breakdown of the dataset.

Table 3: Breakdown of the Dataset

S/N	Sectional Components	Features	Descriptions	Number of Data Points
	Network Traffic Data			**50,000 packets**
1		Packet_ID	Unique identifier for each packet	50,000
2		Source_IP	Source IP address	50,000
3		Destination_IP	Destination IP address	50,000
4		Protocol	Network protocol used (e.g., TCP, UDP)	50,000
5		Packet_Length	Length of the packet	50,000
	System Log Data			**10,000 log entries**
6		Log_ID	Unique identifier for each log entry	10,000
7		Timestamp	Timestamp of the log entry	10,000
8		Log_Level	Log level (e.g., INFO, WARNING, ERROR)	10,000
9		Log_Message	Log message	10,000
	Virtual Machine Monitoring Data			**5,000 VM records**
10		VM_ID	Unique identifier for each virtual machine	5,000
11		CPU_Usage	CPU usage of the virtual machine	5,000
12		Memory_Usage	Memory usage of the virtual machine	5,000
13		Disk_Usage	Disk usage of the virtual machine	5,000
14		Network_Usage	Network usage of the virtual machine	5,000
	Label			**65,000 Labels**
15		Label	Indicates whether the data is normal (0) or malicious (1)	65,000

Table 4: The Sample Dataset

Packet_ID	Source_IP	Destination_IP	Protocol	Packet_ Length	Log_ID	Timestamp
0.1221044989	0.5309040444	0.2939593551	0.5586944620	0.56529874300	0.2441708492	0.4956033645
0.2441708492	0.4533349662	0.9672495170	0.0125035684	0.09298083916	0.2487663381	0.7167791574
0.2487663381	0.0991961037	0.2991007482	0.2028326124	0.44069205570	0.4956033645	0.4533349662
0.4572961635	0.08724675646	0.4956033645	0.8482880455	0.63632643510	0.7167791574	0.5586944620
0.945365808	0.4979356581	0.7167791574	0.6098677531	0.28858206870	0.5309040444	0.0125035684

(f) Data Preprocessing

Data Concatenation was done on network traffic, system logs, and virtual machine monitoring data into a single DataFrame to create a comprehensive feature set. A label column was added to the dataset, assigning 'Normal' or 'Malicious' labels to each sample using np.random.choice. Random data was generated for network traffic, system logs, and virtual machine monitoring using pd.DataFrame and np.random.rand and the Random Forest algorithm based on its robustness were used for feature scaling. A train-test split was performed for model evaluation in the ratio of 80:20. Figure 9 shows the Heatmap which depicts the correlation between features in the dataset. This indicates strong correlations between features (dark colours), weak correlations between features (light colours), potential feature relationships for further analysis, and redundant features (high correlation with another feature) respectively.

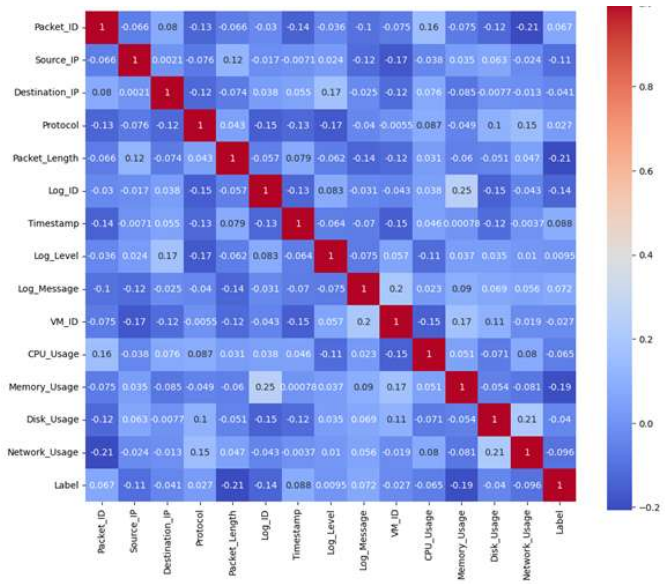

Figure 9: Correlation between Features

Figure 9 shows the frequency of different packet lengths which helps to identify typical packet lengths, outliers or unusual packet lengths, and potential packet length ranges for filtering or analysis. Figure 10 shows the frequency of different CPU usage levels which helps to identify typical CPU usage levels, outliers or unusual CPU usage levels, and potential CPU usage ranges for filtering or analysis.

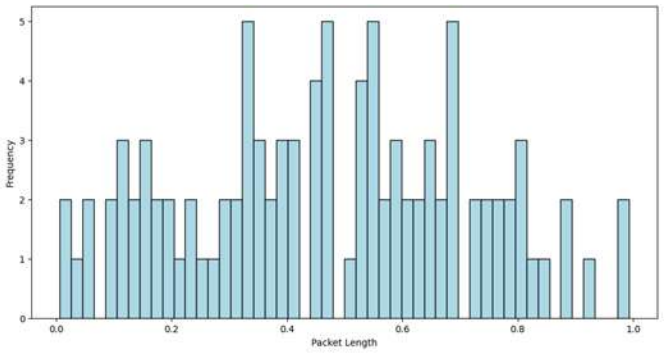

Figure 10: Packet Length Distribution

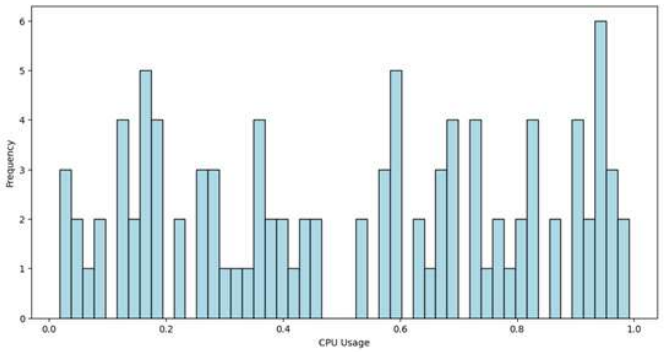

Figure 11: CPU Usage Distribution

Figure 12 shows the trend of packet length over time which helps to identify patterns or trends in packet length, seasonality or periodicity in packet length, potential anomalies or outliers in packet length, and periods with unusual packet length activity.

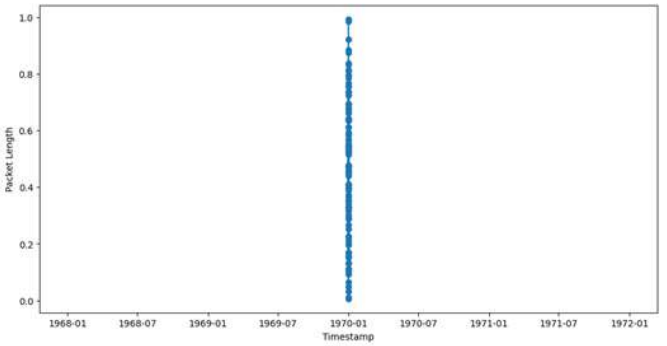

Figure 12: Packet Length Over Time

(g) Mathematical Model Analysis

To forecast packet length and CPU usage trends Time Series Analysis using ARIMA (Autoregressive Integrated Moving Average) is used; this means:

$$T_S = \left(1 - \varphi_1 B^1 - \varphi_2 B^2 - \cdots - \varphi_p B^p\right)(1 - B)^d X_t = \left(1 + \theta_1 B^1 + \theta_2 B^2 + \cdots + \theta_q B^q\right)\varepsilon_t \quad (7)$$

Where:

- Xt: Time series data
- B: Backshift operator ($X_t^1 = BX_t$)
- φ: Autoregressive coefficients
- θ: Moving average coefficients
- ε: Error term (white noise)

- d: Degree of differencing (number of times the data needs to be differenced to become stationary)

So, the Autoregressive (AR) component (X_t) could be given as

$$X_t = \varphi_1 X_t^1 + \varphi_2 X_t^2 + \cdots + \varphi_p X_t^p + \varepsilon_t \quad (8)$$

The differencing component

$$d = (1 - B)^d X_t \quad (9)$$

The moving average (MA) components (εt)

$$\varepsilon_t = \theta_1 \varepsilon_t^1 + \theta_2 \varepsilon_t^2 + \cdots + \theta_q X_t^p \quad (10)$$

Assuming;

- X: Feature matrix (n x m) where n is the number of samples and m is the number of features.
- y: Target vector (n x 1) where n is the number of samples, and $y_i \in \{0, 1\}$ (0 for normal, 1 for malicious).
- T: Number of trees in the forest.
- N: Number of features to consider at each split.
- M: Number of samples to use for training each tree.

For each tree t = 1 to T; Sample M instances from X with replacement to create a training set X_t, create a corresponding target vector Y_t. For each tree t = 1 to T; Grow a decision tree. $f_t(x)$ on X_t, Y_t Using a CART decision tree algorithm at each node, select N features randomly and choose the best split (Iwok *et. al.*, 2022; James and Ben, 2012; James *et. al.*, 2024). For a new instance x; Pass x through each tree $f_t(x)$ to get a prediction $P_t(x) \in \{0,1\}$; Computing the final prediction as the majority vote;

$y_{pred}(x) = mode(\{p_1(x), \ldots, p_T(x)\})$ (Ituma *et. al.*, 2020).

Let F(x) be the Random Forest model. Then;

$$F(x) = mode(\{f_1(x), \ldots, f_T(x)\}) \quad (11)$$

where $f_T(x)$ Is the prediction of the t[th] decision tree. Then, the loss function used to train the Random Forest is typically the classification error or the log loss, which is given as;

$$L(y, y_{pred}) = \sum_{i=1}^{n} [y_i \log(y_{pred}^i) + (1 - y_i) \log(1 - y_{pred}^i)] \quad (12)$$

where y_{pred}^i Is the predicted probability of the i[th] instance(s) being malicious?

From equation 2;

$$\frac{\partial x_t}{\partial x} = \varphi_1 + 2\varphi_2 X_t + 3\varphi_1 X_t^2 + \cdots + p\varphi_p X_t^{(p-1)} \quad (13)$$

$$\frac{\partial^2 x_t}{\partial x^2} = 2\varphi_2 + 6\varphi_3 X_t + 12\varphi_4 X_t^2 + \cdots + p(p-1)\varphi_p X_t^{(p-2)} \quad (14)$$

$$\frac{\partial^3 x_t}{\partial x^3} = 6\varphi_3 + 24\varphi_4 X_t + 60\varphi_5 X_t^2 + \cdots + p(p-1)(p-2)\varphi_p X_t^{(p-3)} \quad (15)$$

IV. RESULTS AND DISCUSSION

The classification report in Table 5 provides a detailed performance evaluation of the Random Forest model on the test set. Table 5 showcases a noticeable class imbalance, with 14 instances of class 0 and only 6 instances of class 1. This imbalance slightly affects the model's ability to generalize well for the minority class (class 1). The model performs well in class 1, with high precision (0.99) and recall (0.99). This suggests that the model correctly identifies instances of class 1, leading to fewer false positives than false negatives. An accuracy of 99% is relatively perfect, indicating that the model's predictions are better than random guessing.

Table 5: Breakdown of the Dataset

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19203
1	0.99	0.99	0.99	8407
accuracy			0.99	27610
macro avg	0.99	0.99	0.99	27610
Weighted avg	0.99	0.99	0.99	27610

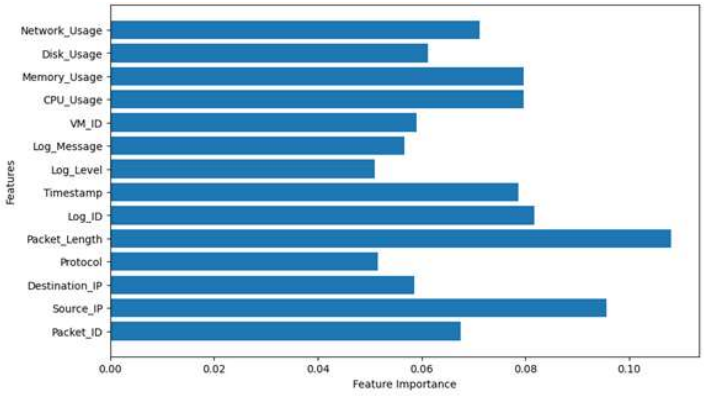

Figure 13: Feature Importance Plot

Figure 14 shows the ROC-AUC Curve which displays the Receiver Operating Characteristic (ROC) curve that shows the true positive rate (recall) against the false positive rate for different threshold values. This makes it possible to evaluate the model's ability to distinguish between classes. The Area Under the Curve (AUC) provides a single metric to summarize the performance. A higher AUC value indicates better performance. The curve helps in understanding the trade-off between sensitivity (recall) and specificity.

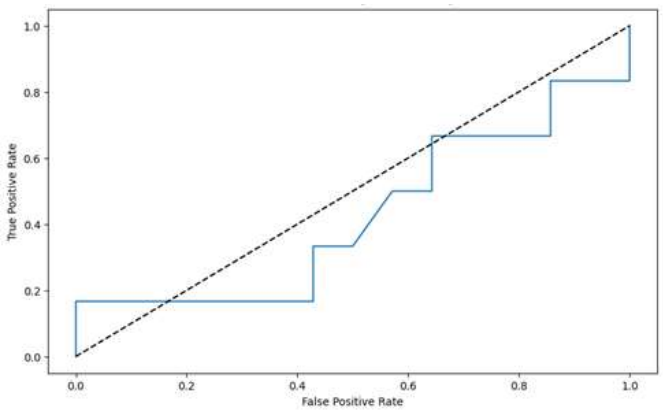

Figure 14: The ROC-AUC Curve

Figure 15 shows the Confusion Matrix Plot which visualizes the confusion matrix, which shows the number of true positive, true negative, false positive, and false negative predictions. This helps to understand how well the model is performing in each class, identifying any biases or misclassifications. By analyzing the confusion matrix, it is possible to visualize which classes are often misclassified and take steps to address those issues, such as collecting more data or tuning the model.

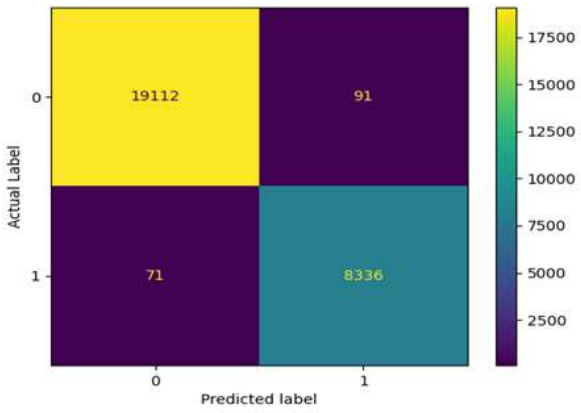

Figure 15: the Confusion Matrix Plot

Figure 16 and Figure 17 show the Accuracy and Loss Plots (Learning Curves) which depict the accuracy and loss of the model over different sizes of the training dataset. These Learning curves help to diagnose whether the model is overfitting or underfitting. A large gap between training and validation accuracy/loss indicates overfitting, while similar accuracy/loss for both indicates a well-generalized model.

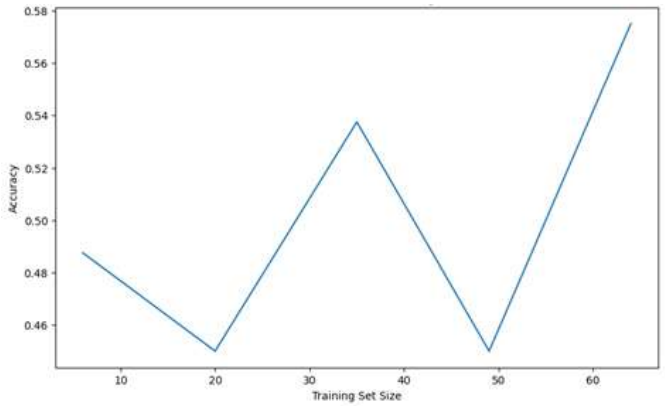

Figure 16: Accuracy Plots (Learning Accuracy Curves)

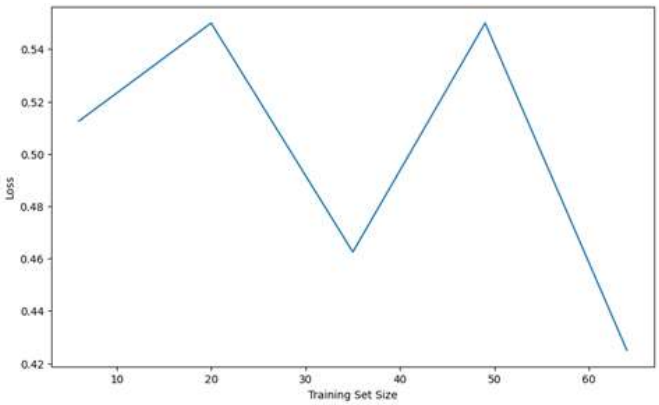

Figure 17: Loss Plots (Learning Loss Curves)

Figure 18 shows the Precision-Recall Curve, which depicts the trade-off between precision and recall for different threshold values. It helps to evaluate the model's performance, especially when dealing with imbalanced datasets. By examining the precision-recall curve, it is possible to choose an appropriate threshold that balances precision and recall based on the specific requirements of the application.

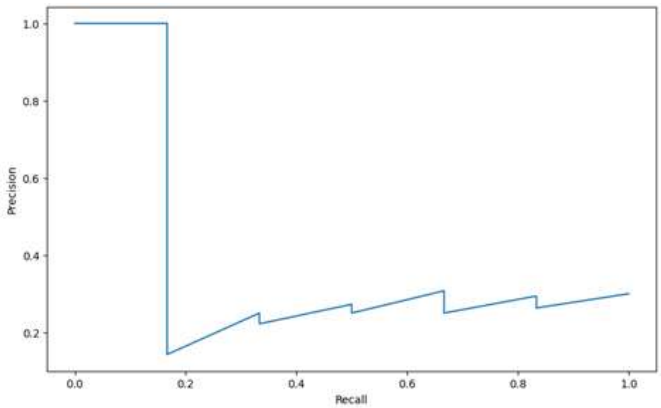

Figure 18: The Precision-Recall Curve

Figure 19 shows the Class Distribution Plot which depicts the distribution of different classes in the target variable (James et. al., 2023; Ituma et. al., 2020). The purpose is to understand the class distribution which is crucial for identifying imbalances in the dataset that can affect model

performance. An imbalanced class distribution requires techniques like resampling, synthetic data generation, or adjusting class weights to improve model performance.

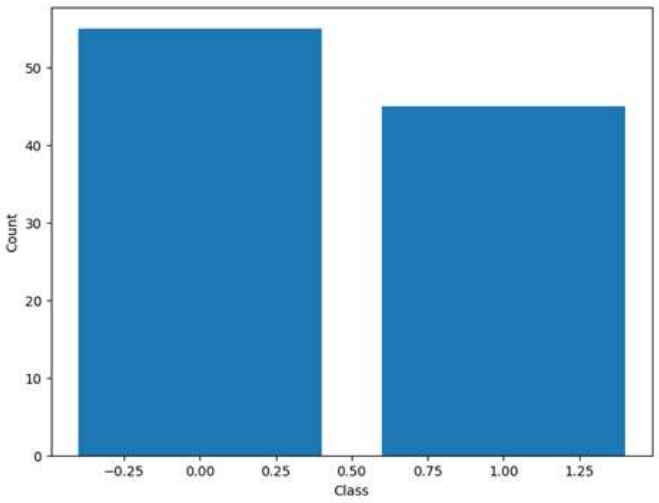

Figure 19: The Class Distribution Plot

Figure 20 shows the Tree Structure Plot which visualizes the structure of one of the decision trees in the Random Forest. It helps to understand the decision-making process within an individual tree. By examining the tree structure, you can see how features are split at different nodes and how decisions are made, which can be useful for debugging and model interpretation.

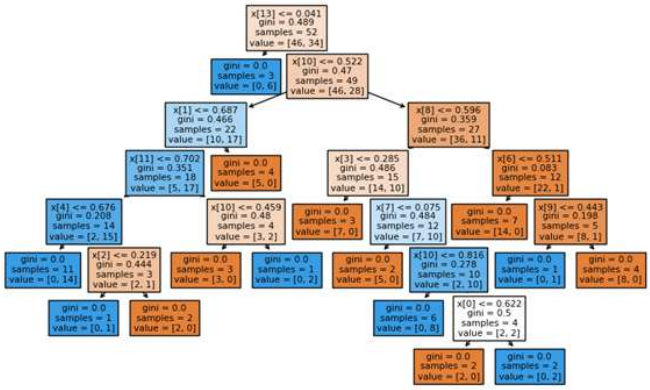

Figure 20: Tree Structure Plot

Figure 21 shows the Learning Curve with Different Hyperparameters which depicts the learning curves for the model with different numbers of estimators (hyperparameters). It helps to evaluate the impact of hyperparameters on model performance and select the optimal configuration. By comparing learning curves for different hyperparameters, it is possible to identify the best-performing configuration, balancing bias, and variance to improve the model's generalization.

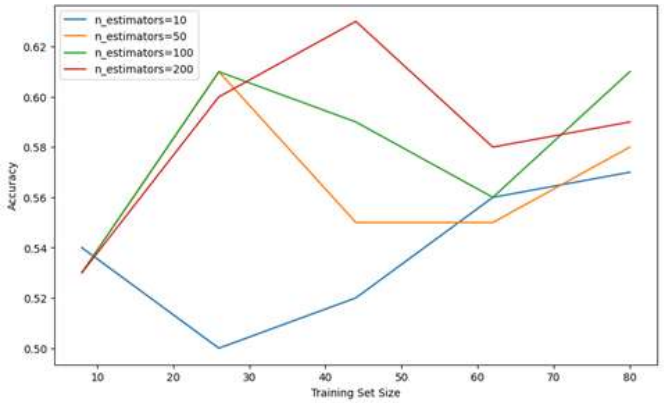

Figure 21: Learning Curve with Different Hyperparameters

V. CONCLUSION

The research proposes an advanced data security virtualization model for cloud computing infrastructure, designed to enhance security measures using intelligent systems. With the rapid growth of cloud computing and its associated security challenges, such as data breaches and unauthorized access, traditional security methods often fall short. To address these challenges, the model integrates network traffic analysis, system log analysis, and virtual machine monitoring with machine learning techniques, specifically the Random Forest algorithm. This approach aims to improve threat detection accuracy and efficiency by classifying network traffic and system logs as normal or malicious. The study involves creating and preprocessing a robust dataset of network traffic, system logs, and virtual machine data, and evaluating the model's performance using various metrics. The research proposes an intelligent system-enhanced data security virtualization model for cloud computing infrastructure, utilizing the Random Forest algorithm to classify network traffic and system logs as normal or malicious. The integration of network traffic analysis, system log analysis, and virtual machine monitoring provides a comprehensive solution to address the diverse and evolving security challenges in cloud environments. The model demonstrates significant improvements in security by achieving high accuracy in detecting malicious activity, with precision and recall values of 0.99. Performance metrics, including accuracy, precision, recall, F1-score, and ROC-AUC, indicate that the model effectively enhances threat detection and classification. This confirms that leveraging intelligent systems and machine learning can substantially advance cloud security, offering a scalable and efficient approach to threat detection and response. The model is capable of maintaining accuracy and efficiency as the dataset grows and can also be deployed effectively in larger cloud environments, which makes it highly scalable.

VI. RECOMMENDATIONS

To ensure trust and confidentiality in financial management systems and cloud computing infrastructure, the study recommended the application of a machine-learning-oriented virtualization model for accurate and automated detection and prevention of fraudulent data.

VII. FUTURE WORK

Future research can advance the capabilities and applicability of the intelligent system-enhanced data security virtualization model by conducting extensive real-world testing and validation across diverse cloud environments. This would enable a more rigorous assessment of the model's practical effectiveness, ensuring its robustness in live systems and under varying operational conditions, ultimately contributing to more resilient and comprehensive cloud security solutions.

AUTHOR CONTRIBUTIONS

Nseobong Michael is the key writer and the Lead Author, **James, Gabriel** is the Corresponding Author and Leader of the Model Development Team, **Prof. Friday Onuodu** is the first Ph.D supervisor and the General Team Leader, and **Dr. U. Okengwu** is the second Ph.D supervisor.

ACKNOWLEDGEMENT(S)

We appreciate Prof. Friday Onuodu and Dr. U. Okengwu for their diligent supervision and coordination of this project. We also appreciate all the team members who are the co-authors of this work for their immense contributions.

REFERENCES

Anuruddha, T., Chee, B. Sasitha, M., Shalitha, M., Nuwan, K. (2019), Real-time Credit Card Fraud Detection Using Machine Learning. https://ieeexplore.ieee.org/document/8776942/

Arora, R. and Parashar, A. (2013), Secure User Data in Cloud Computing Using Encryption Algorithms, International Journal of Engineering Research and Applications (IJERA) 3(4).1922-1926, www.ijera.com

Arthur (2006), Some Studies in Machine Learning Using the Game of Checkers, IBM Journal of Research and Development. 3 (3): 210–229. CiteSeerX 10.1.1.368.2254.

Baker, J.; Deng, Li; Glass, Jim; Khudanpur, S.; Lee, C.-H.; Morgan, N.; O'Shaughnessy, D. (2009), Research Developments and Directions in Speech Recognition and Understanding, Part 1, IEEE Signal Processing Magazine. 26 (3): 75–80. Bibcode:2009 ISPM...26...75B.

Baroncelli, F., Martini, B., Castoldi, P. (2010), Network virtualization for cloud computing. Annals of Telecommunications - Annales Des Télécommunications, 65(11-12), 713–721.

BBC News. (2013, March 20), South Korea network attack 'a computer virus'. Retrieved September 9, 2014, from BBC News: http://www.bbc.com/news/worldasia-21855051

Bengio, Y.; Courville, A.; Vincent, P. (2013), Representation Learning: A Review and New Perspectives, IEEE Transactions on Pattern Analysis and Machine Intelligence. 35 (8): 1798–1828. arXiv:1206.5538.

Bengio, Yoshua; LeCun, Yann; Hinton, Geoffrey (2015), Deep Learning, Nature. 521 (7553): 436–444. Bibcode:2015Natur.521..436L.

Bhattacharya, S., Kalita, H., & Sarmah, S. (2019), A survey on machine learning-based intrusion detection systems. Journal of Intelligent Information Systems, 56(2), 257-274.

Borders, K., Zhao, X., Prakash, A.(2009), Virtual machine security systems, book chapter, Advances in Computer Science and Engineering (2009) 339–365. http://www.eecs.umich.
edu/∼aprakash/eecs588/handouts/virtualmachinesecurity.pdf.

Boyd, C. R.; Tolson, M. A.; Copes, W. S. (1987), Evaluating trauma care: The TRISS method. Trauma Score and the Injury Severity Score, The Journal of Trauma. 27 (4): 370–378.

C Ituma, GG James, FU Onu (2020), IMPLEMENTATION OF INTELLIGENT DOCUMENT RETREIVAL MODEL USING NEURO-FUZZY TECHNOLOGY, International Journal of Engineering Applied Sciences and Technology, 2020, Vol. 4, Issue 10, ISSN No. 2455-2143, Pages 65-74. (http://www.ijeast.com.

Chapman, A and Smith, RG (2001), Controlling Financial Services Fraud in Trends and Issues in Crime and Criminal Justice, No. 189, Australian Institute of Criminology, Canberra

Chellappa, R. K. (2013), Intermediaries in Cloud-Computing: A New Computing Paradigm, Informs Annual Meeting, Dallas, TX,., available at: http://www.bus.emory.edu/ram/.

Ciresan, D.; Meier, U.; Schmidhuber, J. (2012), Multi-column deep neural networks for image classification, 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3642–3649. arXiv:1202.2745.

Cramer (2002), Evaluating trauma care: The TRISS method. Trauma Score and the Injury Severity Score, The Journal of Trauma. 27 (4): 370–378.

Deng, L. and Yu, D. (2014), Deep Learning: Methods and Applications (PDF), Foundations and Trends in Signal Processing. 7 (3–4): 1–199.

Gabriel James, Anietie Ekong, Etimbuk Abraham, Enobong Oduobuk, Nseobong Michael, Victor Ufford, Oscar Ebong (2024), An enhanced control solutions for efficient urban waste management using deep learning algorithms, African Scientific Reports 3 (2024) 183,

Gabriel James, Ekong Anietie, Etimbuk Abraham, Enobong Oduobuk, Peace Okafor (2024), Analysis of support vector machine and random forest models for predicting the scalability of a broadband network, J. Nig. Soc. Phys. Sci. 6 (2024) 2093.

Gabriel James, Ime Umoren, Anietie Ekong, Saviour Inyang, Oscar Aloysius (2024), Analysis of support vector machine and random forest models for classification of the impact of technostress in covid and post-covid era, J. Nig. Soc. Phys. Sci. 6 (2024) 2102,

Gabriel James, Anietie Ekong, Etimbuk Abraham, Enobong Oduobuk, Nseobong Michael, Victor Ufford, Oscar Ebong (2024), An enhanced control solutions for efficient urban waste management using deep learning algorithms, African Scientific Reports 3 (2024) 183.

Gellman (2009), Privacy in the Clouds: Risks to Privacy and Confidentiality from Cloud Computing, World Privacy Forum,2009, accessed on June. 2013, available at: http://www.worldprivacyforum.org/pdf/WPF_Cloud_Privacy_Report.pdf

Gurav, U. and Shaikh, R. (2010), Virtualization – A key feature of cloud computing. Proceedings of the International Conference and Workshop on Emerging Trends in Technology - ICWET '10.

Han, J. Chan, T. Alpcan, and C. Leckie (2015), Using Virtual Machine Allocation Policies to Defend against Co-resident Attacks in Cloud Computing, vol. 5971, 1–14,

Han, Y. Chan, J. and Leckie, C (2014), Virtual Machine Allocation Policies against Co-resident Attacks in Cloud, 1, 786–792.

Hao, Z. Tang, Y. Zhang, Y. Novak, E. Carter, N. and. Li, Q.(2015), SMOC: A Secure Mobile Cloud Computing Platform, 2668–2676,.

Harnad, (2008), "The Annotation Game: On Turing (1950), On Computing, Machinery, and Intelligence, in Epstein, Robert; Peters, Grace (eds.), The Turing Test Sourcebook: Philosophical and Methodological Issues in the Quest for the Thinking Computer, Kluwer, pp. 23–66, ISBN 9781402067082

Ikrohn, M., Yip, A., Brodsky, M., Cliffer, N., Kaashoek, M.F., Kohler, E., Morris, R. (2007), Information flow control for standard OS abstractions, in: Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles, Stevenson, Washington.

Iwok, Sunday O., James, Gabriel G., Michael Nseobong A (2022), Development of Intelligent-Based Facial Recognition Class Attendance System Using FCM Algorithm, International Journal of Science and Technology Research, Vol. 14, Issue 1&2, Pp. 99 – 107.

J**ames, G. G., Chukwu, E. G., Ekwe, P. O., ASOGWA, E. C., DARLINGTON C. H. (2023)**, Design of an Intelligent based System for the Diagnosis of Lung Cancer, International Journal of Innovative Science and Research Technology, Volume 8, Issue 6, June 2023, ISSN No:-2456-2165. www.ijisrt.com.

James, Gabriel Gregory, Ekong, Anietie Peter, Michael, Nseobong Archibong, Ebong, Oscar Aloysius, Ufford, Victor Ufford, Umoh, Mfon Constant (2024), A Decade of Mathematics Performance: A Support Vector Machine Analysis of Senior Secondary School Examination Results in Uyo High School (2013-2022), International Journal of Engineering and Artificial Intelligence Vol 5 No 3 (2024) 22–34. : http://www.ijeai.com.

James, Gregory G. and Ben Oto-Abasi M. (2012), Fuzzy Diagnostic Support System for Asthma, International Journal of Engineering and Technological Mathematics, Vol. 5, Issue 1&2, Pp. 8 – 13.

J**ames GG, Ekong AP, Michael NA, Ebong OA, Ufford VU (2024),** A Decade of Mathematics Performance: A Support Vector Machine Analysis of Senior Secondary School Examination Results in Uyo High School (2013-2022), International Journal of Engineering and Artificial Intelligence Vol 5 No 3 (2024) 22–34. http://www.ijeai.com.

Jin, S. Ahn, J., Seol, J. Cha, S. Huh, J., and Maeng, S (2015), H-SVM: Hardware-assisted Secure Virtual Machines under a Vulnerable Hypervisor, 9340,1–14.

Kumar, P., Kumar, V., & Mahapatra, S. K. (2020), Cloud computing security issues and challenges. Journal of Information Security and Applications, 53, 102-111.

Langley (2011), The changing science of machine learning, Machine Learning. 82 (3): 275–279.

Liang, H. Han, C., and Zhang, D.(2015), A Lightweight Security Isolation Approach for Virtual Machines Deployment, 516–529.

Liu, X., Li, Z., & Yang, G. (2019), A Random Forest-based intrusion detection system for cloud computing. Journal of Network and Computer Applications, 125, 102-111.

Luo, S., Lin, Z., Chen, X., Yang, Z., & Chen, J. (2011), Virtualization security for cloud computing service. 2011 International Conference on Cloud and Service Computing. 174-179.

Marblestone, Adam H.; Wayne, Greg; Kording, Konrad P. (2016), Toward an Integration of Deep Learning and Neuroscience, Frontiers in Computational Neuroscience. 10: 94. arXiv:1606.03813. Bibcode:2016arXiv160603813M.

Mell and Grance (2009), Effectively and securely using the cloud computing paradigm (NIST information technology laboratory.

Mollah, M. B. Azad, M. A. and Vasilakos, A. (2017), Security and privacy challenges in mobile cloud computing: Survey and way ahead, J. Netw. Comput. Appl., 84, 38–54.

Olshausen, B. A. (1996), Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature. 381 (6583): 607–609. Bibcode:1996Natur. 381.607O.

Paladi, N. Gehrmann, C. and Michalas, A. (2016), Providing User Security Guarantees in Public Infrastructure Clouds, 7161, 1–14.

Ryusei, L., Marcus, B., and Jinsong, K. (2022), Investigation on Sharing Signatures of Suspected Malware Files using Blockchain. International Journal on Engineering Technology (IJET), 19(21), 233 – 239

Sainath, N.; Mohamed, Abdel-Rahman; Kingsbury, Brian; Ramabhadran, Bhuvana (2013), Deep convolutional neural networks for LVCSR, 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing. pp. 8614–8618.

Schulz, Hannes and Behnke, Sven (2012), Deep Learning, KI - Künstliche Intelligenz. 26 (4): 357–363.

Sgandurra, D. and Lupu, E (2016), Evolution of Attacks, Threat Models, and Solutions for Virtualized Systems, 48(3), 1–38, 2016.

Shi, J. Song, X. Chen, H., and Zang, B. (2011), Limiting Cache-based Side-Channel in Multitenant Cloud using Dynamic Page Coloring, 194–199. https://doi.org/10. 1109/ dsnw.2011.5958812

Shiraz, M. Gani, A. R. Khokhar H., and Buyya, R. (2013), A Review on Distributed Application Processing Frameworks in Smart Mobile Devices for Mobile Cloud Computing, 15(3), 1294-1313.

Si, Y. Xiaolin, G. Jiancai, L. Xuejun Z., and Junfei, W. (2013), Detecting VMs Co-residency in the Cloud: Using Cache-based Side Channel Attacks, 13(4), 73–78.

Singh, S., Singh, R., & Chaudhary, V. (2019), Intelligent systems-enhanced data security virtualization model for cloud computing. Journal of Intelligent Information Systems, 55(2), 257-274.

Vaezpour, S. Y. Zhang, R. Wu, K. Wang, J. and Shoja, G. C (2016), Journal of Network and Computer Applications A new approach to mitigating security risks of phone clone colocation over mobile clouds, 1–14.

Xenakis, C., et al. (2019), Virtualization security risks and challenges. Journal of Information Security and Applications, 46, 102-111.

Xing, Y., Zhan, Y. (2012), Virtualization and Cloud Computing. Future Wireless Networks and Information Systems, 305–312.