IEEE ComSoc | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 14 April 2024; revised 31 October 2024, 20 January 2025, and 4 March 2025; accepted 24 April 2025.
Date of publication 28 April 2025; date of current version 12 May 2025.

The associate editor coordinating the review of this article and approving it for publication was E. Bou-Harb.

Digital Object Identifier 10.1109/TMLCN.2025.3564912

AI-Powered System for an Efficient and Effective Cyber Incidents Detection and Response in Cloud Environments

MOHAMMED ASHFAAQ M. FARZAAN[1], MOHAMED CHAHINE GHANEM[2,3], AYMAN EL-HAJJAR[4], AND DEEPTHI N. RATNAYAKE[5]

[1]Priam Cyber AI Ltd., WC2H 9JQ London, U.K.
[2]Cybersecurity Institute, University of Liverpool, L69 7ZX Liverpool, U.K.
[3]Cyber Security Research Centre, London Metropolitan University, N7 8DB London, U.K.
[4]Cyber Security Research Group, University of Westminster, W1B 2HW London, U.K.
[5]Cybersecurity and Computing Systems Research Group, University of Hertfordshire, AL10 9EU Hatfield, U.K.
CORRESPONDING AUTHOR: M. C. GHANEM (mohamed.chahine.ghanem@liverpool.ac.uk)

ABSTRACT The growing complexity and frequency of cyber threats in cloud environments call for innovative and automated solutions to maintain effective and efficient incident response. This study tackles this urgent issue by introducing a cutting-edge AI-driven cyber incident response system specifically designed for cloud platforms. Unlike conventional methods, our system employs advanced Artificial Intelligence (AI) and Machine Learning (ML) techniques to provide accurate, scalable, and seamless integration with platforms like Google Cloud and Microsoft Azure. Key features include an automated pipeline that integrates Network Traffic Classification, Web Intrusion Detection, and Post-Incident Malware Analysis into a cohesive framework implemented via a Flask application. To validate the effectiveness of the system, we tested it using three prominent datasets: NSL-KDD, UNSW-NB15, and CIC-IDS-2017. The Random Forest model achieved accuracies of 90%, 75%, and 99%, respectively, for the classification of network traffic, while it attained 96% precision for malware analysis. Furthermore, a neural network-based malware analysis model set a new benchmark with an impressive accuracy rate of 99%. By incorporating deep learning models with cloud-based GPUs and TPUs, we demonstrate how to meet high computational demands without compromising efficiency. Furthermore, containerisation ensures that the system is both scalable and portable across a wide range of cloud environments. By reducing incident response times, lowering operational risks, and offering cost-effective deployment, our system equips organizations with a robust tool to proactively safeguard their cloud infrastructure. This innovative integration of AI and containerised architecture not only sets a new benchmark in threat detection but also significantly advances the state-of-the-art in cybersecurity, promising transformative benefits for critical industries. This research makes a significant contribution to the field of AI-powered cybersecurity by showcasing the powerful combination of AI models and cloud infrastructure to fill critical gaps in cyber incident response. Our findings emphasise the superior performance of Random Forest and deep learning models in accurately identifying and classifying cyber threats, setting a new standard for real-world deployment in cloud environments.

INDEX TERMS Cyber incident, digital forensics, artificial intelligence, machine learning, cloud security, Incident response, NSL-KDD, UNSW-NB15, CIC-IDS-2017.

I. INTRODUCTION

IN RECENT years, the proliferation of cyber attacks targeting organisations across various industries has reiterated the critical need for robust incident response capabilities. According to the UK government's Cybersecurity Breaches Survey in 2024 [1], the 2024 Report on the Cybersecurity Posture of the United States by the Office of the National Cyber Director, the ENISA Threat Landscape (ETL)

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

Report 2024 [47] and IBM's Cost of a Data Breach Report [2], a significant percentage of businesses and charities have experienced breaches or attacks, with alarmingly low adoption rates of formal incident response capabilities (IR). [3]. Consequently, there is a pressing demand for organisations to invest in incident response capabilities to protect against data breaches and cyber threats [4]. Organisations with well-tested Incident Response (IR) capabilities and high levels of AI and ML integration for threat detection and response demonstrated substantially lower data breach costs, as highlighted by IBM's 2024 Cost of Data Breach report [2]. This shows how essential it is for organisations to embrace AI and ML technologies to improve their cybersecurity posture [40].

This study investigates how AI contributes to cybersecurity and explores the potential to apply it in cloud environments to address associated challenges. It proposes a novel system that uses AI and machine learning techniques to enhance cybersecurity within cloud environments. The proposed system includes three main components; a network traffic classifier, a web intrusion detection system (WIDS) and a malware analysis system.

The proposed system stands out through its integration of real-time feature engineering, modular containerisation for scalability, and lightweight agents for efficient log collection in the Web Intrusion Detection System (WIDS). By dynamically adjusting contamination ratios based on domain-specific scenarios, it significantly reduces false positives while maintaining high accuracy in diverse cloud environments. In addition, the system leverages a honeypot subsystem for proactive threat detection, continually updating the models with real-world attack data. This seamless multi-cloud deployment strategy, supported by Docker and Kubernetes, ensures adaptability to emerging threats, setting it apart from traditional reactive security solutions like Splunk and CrowdStrike.

The network classifier uses real-time traffic capture to analyse ongoing network activity for anomalies potentially indicative of malicious behaviour. The NSL-KDD dataset [10],UNSW-NB15 [12], and CIC-IDS-2017 [11], which are widely recognised benchmarks for network traffic analysis, collectively serve as the foundational datasets for training and evaluating our classifier [36]. By effectively classifying incoming traffic in real time based on this rich set of features, the classifier can significantly improve network security by enabling the prompt identification and mitigation of potential cyberattacks [13].

The Web Intrusion Detection System (WIDS) focusses on detecting suspicious behaviour in web traffic to prevent unauthorised access [4]. This is achieved by extracting informative features from standard HTTP server logs. The key innovation of this design lies in its real-time deployment and distributed data collection using lightweight agents on web servers. This ensures efficient log collection and minimises the impact on individual servers [27]. Anomaly detection employs the Isolation Forest algorithm, which is effective in high-dimensional datasets commonly encountered in security applications. To reduce false positives, the application triggers alerts only when the number of detected anomalies exceeds a predefined threshold, which can be optimised by the network administrator after assessing and factorising the associated risks, based on the assumption that real-world attacks often involve rapid bursts of activity [6].

The Malware Analysis system streamlines the process of analysing suspicious files to determine if they are malicious [26]. It achieves this by first extracting key features such as string, Import Address Table (IAT), and callback servers from training binaries and then using them to train a model. The model adopts a combined model architecture to mitigate false positives. It uses the Random Forest model as the primary model, complemented by a secondary model, the Keras TensorFlow model [20]. Both models were trained on a comprehensive dataset obtained from VirusTotal.com. The system follows a logical flow, in which uploaded files undergo initial processing and classification. If the initial model predicts a high likelihood of malicious content, the file is classified as ''Malicious'', and a detailed analysis report is generated. A secondary deep learning model is invoked for precise prediction of files with uncertain classification [21].

This paper represents a concerted effort to explore the practical application of AI techniques in the domain of digital forensics, with a specific focus on developing an AI-enabled cyber incident investigation framework tailored for deployment in cloud environments. Using the capabilities of AI and machine learning (ML), this research seeks to enhance the efficacy and efficiency of digital forensics processes, allowing organisations to better detect, analyse and mitigate cyber threats in cloud infrastructures. Through comprehensive investigations, this work delves into three distinct AI and ML applications of digital forensics: Network Traffic Classification, Web Intrusion Detection, and Malware Analysis Systems. These applications are meticulously integrated within leading cloud platforms such as Google Cloud and Microsoft Azure to facilitate forensic operations effectively [22].

The findings derived from this research shed light on several critical aspects of AI-driven digital forensics. Firstly, the suitability of the Random Forest algorithm emerges prominently for classification tasks, demonstrating robust performance in distinguishing between various network behaviours and identifying potential threats. Furthermore, integrating deep learning models opens new horizons in Malware Analysis, underscoring the potential for enhanced accuracy and efficacy in digital forensic tasks. In addition, this research underscores the effectiveness and scalability of cloud environments as hosting platforms for AI and ML systems. By harnessing cloud infrastructures' computational power and flexibility, organisations can significantly enhance their digital forensic capabilities, thereby overcoming the constraints of traditional on-premises solutions.

Furthermore, the exploration of container technology underscores its pivotal role in facilitating the deployment and scalability of AI and ML-driven digital forensics

systems within cloud environments. The agility and resource efficiency offered by containerisation presents compelling advantages for organisations seeking to streamline their forensic operations and adapt dynamically to evolving cyber threats. In conclusion, this research presents a novel and pragmatic approach to combating cybercrime in cloud environments, leveraging the synergistic potential of Artificial Intelligence and cloud resources.

II. RESEARCH QUESTIONS AND CONTRIBUTION

This research represents an important contribution to the cybersecurity domain, as well as a complete proposal of an AI-enabled cyber incident response system tailored for cloud environments. Unlike traditional systems, the proposed system capitalises on the strengths of ML models such as Random Forest and Deep Learning to enhance detection accuracy and efficiency during incident response within the cloud infrastructures; it has provided real-time analysis and classification of cyber attacks with promising results, which vary from Network Traffic Classification. Furthermore, the innovative use of container technology, which ensures scalability and operational efficiency, allows the current and further growth in the rapidly emerging sophistication level of these cyber threats in a digital world. Therefore, this contribution can underpin AI's potential to revolutionise cybersecurity and establish a robust, scalable framework to ensure its adoption across different cloud platforms—a standard-setting incident response methodology. In this research, our research questions are formulated as follows.

RQ1 How would the integration of AI within cyber incident response systems precisely enhance detection and response capabilities against emerging cyber threats in cloud environments?

RQ2 How would a unified AI-led system that includes a traffic classifier, malware analysis, and web intrusion detection enhance the effectiveness and efficiency of cyber incident investigations within cloud settings?

RQ3 How do cloud platforms like Google Cloud and Microsoft Azure actuate scalability and versatility in the deployment of AI-led systems, and what is the contribution of a T-pot [35] in model development and an ELK Stack in log gathering and visualisation in proactive threat detection?

III. RELATED WORK

This section synthesises research related to digital forensics and incident response systems in cloud environments and the integration of Artificial Intelligence (AI) and Machine Learning (ML) within these domains. The selected papers shed light on various methodologies, frameworks, and technologies to enhance cyber forensic capabilities and address emerging challenges in cloud computing security.

A. INCIDENT RESPONSE AND INVESTIGATION IN CLOUD ENVIRONMENTS

Several works have been done to address incident detection and response in cloud environments. Stelly and Roussev [18] introduced SCARF, a container-based software framework designed to enable digital forensic processing at cloud scale. Their work contributes to the field by advocating for the use of containers to address critical issues in digital forensics, providing practical insight into integration capabilities and performance considerations. However, the absence of experiments in actual cloud environments limits the assessment of the scalability and full potential of SCARF.

Hemdan and Manjaiah [23] presented a cloud forensics investigation model centred on Digital Forensics as a Service (DFaaS). Their contribution lies in emphasising the deployment of forensic servers within the cloud service providers' infrastructures to enhance evidence collection and analysis. Although the performance and features of the model show promise, its reliance on proprietary cloud environments restricts its general applicability to public cloud deployments.

Dykstra and Sherman [24] introduced FROST, a trusted digital forensics tool specifically designed for the OpenStack cloud computing platform. A key contribution of FROST is its focus on ensuring evidence integrity, enabling the reliable acquisition of virtual disks and API logs. However, its limited compatibility with OpenStack platforms presents challenges for broader applicability across heterogeneous cloud infrastructures.

Edington and Kishore [33] proposed a comprehensive forensics framework for cloud computing featuring a central forensic server and an external forensic monitoring plane. The framework addresses key challenges in cloud forensics and provides valuable insights into evidence management. However, its reliance on on-premise resources and lack of validation in actual cloud environments call for further testing to ensure its robustness and scalability.

Ben-Shimol et al. [49] propose an innovative ontology-based framework that converts serverless audit logs into a unified activity knowledge graph, thereby powering situational awareness tools such as an incident response dashboard and a criticality of asset risk assessment framework. Their method capitalizes on the static analysis of log data to reveal contextual relationships within managed serverless environments. On the other hand, our research ventures beyond log transformation by deploying a comprehensive AI-powered system that integrates real-time network traffic classification, web intrusion detection, and post-incident malware analysis using deep learning models. This dynamic approach, underpinned by containerized deployment and cloud-based GPU acceleration, not only enables immediate threat detection but also scales efficiently to accommodate evolving cyber threats in diverse cloud environments.

RCInvestigator [50] presents an interactive framework that transforms cloud audit logs into a knowledge graph, enabling experts to manually explore and annotate the root causes of anomalies through a multi-stage, human-machine collaborative process. This visual approach provides deep, interpretive insights into anomaly origins but depends heavily on expert involvement for its reasoning steps. By contrast, our research adopts a fully automated strategy that integrates real-time network traffic classification, web intrusion detection, and

hybrid malware analysis using deep learning models and containerized deployment.

In summary, while these works have made significant contributions to digital forensics and incident response, they often fall short in terms of scalability, integration across diverse platforms, or practical validation in real-world cloud settings. These gaps highlight the need for solutions like the one proposed in this research, which aim to address these limitations by integrating advanced AI techniques and containerisation for scalable and robust incident response in cloud environments.

B. AI AND ML IN DIGITAL FORENSICS AND INCIDENT RESPONSE

Various studies have explored integrating AI and ML into the Digital Forensics and Incident Response (DFIR) process. Zewdie and Girma [25] proposed a hybrid AI/ML model for IoT security in cloud environments, combining supervised, unsupervised, and reinforcement learning to analyze cloud-based big data efficiently. Their key contribution is leveraging deep learning for feature extraction and classification while emphasizing the need for large datasets like CIADA and Packt. However, their approach lacks a detailed evaluation of zero-day attack handling and computational efficiency on resource-constrained devices. In contrast, our research introduces real-time feature engineering and containerized deployment, ensuring scalability for large-scale cloud security applications. By optimizing deep learning models for cloud-based GPUs and TPUs, our system enhances both efficiency and threat detection capabilities.

Baptista et al. [42] proposed a novel approach to malware detection by transforming binary files into images and using Self-Organizing Incremental Neural Networks (SOINN) for analysis. While their method reports promising accuracies for file types such as PDFs and DOCs, its generalizability to other malware formats and the computational cost of image processing for real-time applications remain uncertain. In contrast, our malware analysis system extracts key features from binaries and employs a hybrid model combining a Random Forest classifier with a deep learning model (Keras TensorFlow) to achieve robust detection across diverse malware types. This approach not only delivers superior accuracy (up to 99%) but also leverages containerized deployment for scalable, real-time operation in cloud environments.

Du et al. [37] investigated the application of AI in digital forensics, emphasizing its potential to automate evidence processing and reduce case backlogs. Their study highlights the promise of AI-based tools in expediting investigations but provides limited details on the specific techniques and their performance across diverse forensic tasks. Our research presents an AI-driven cyber incident detection and response system tailored for cloud environments. Unlike Du et al.'s broad approach, our system leverages containerized microservices for network traffic classification, web intrusion detection, and malware analysis ensuring real-time, scalable processing and robust computational performance in dynamic, cloud-based settings.

Balushi et al. [39] underscore the potential of machine learning in digital forensics by reviewing various algorithms such as Support Vector Machines, Decision Trees, K-Nearest Neighbors, Naïve Bayes, Principal Component Analysis, Logistic Regression, Singular Value Decomposition, and Apriori for automating evidence processing and improving investigative accuracy. While their paper provides a broad overview of these techniques across diverse forensic scenarios, it stops short of a detailed examination of the specific implementation mechanisms and a critical comparison of each algorithm's strengths and weaknesses. Our system not only implements a hybrid machine learning framework for real-time cyber incident detection and response but also rigorously evaluates our models in terms of accuracy, computational efficiency, and scalability within cloud environments. This in-depth analysis bridges the gap by offering actionable insights into the selection and deployment of ML techniques for practical digital forensics.

Qadir and Varol [28] highlighted the crucial role of machine learning in addressing challenges in digital forensics, proposing applications such as link analysis and fraud detection. Despite its insightful analysis, the paper lacked empirical validation of the proposed techniques. Additionally, it overlooked potential drawbacks associated with using machine learning in this context, such as the substantial amount of training data required and the possibility of bias within the algorithms. In general, the paper provides a jumping-off point for exploring the potential of machine learning in digital forensics.

Hilmand et al. [29] conducted a survey study on the application of ML in digital forensics, providing information on various algorithms used for tasks such as access controls and image distortion detection. The authors discussed various applications of ML in the field without delving into the specific strengths and weaknesses of each application. Additionally, the paper did not address the potential drawbacks of using ML, such as algorithm overheads and inherent biases.

Rughani [32] proposed a digital forensic framework that uses artificial intelligence to improve tool performance and minimise user interaction. However, it remains unclear how the framework would address the handling of entirely new types of cybercrime that are not included in its training data. Although the suggested framework shows potential as a viable solution, it still needs a thorough evaluation and validation of the results it claims to achieve. In contrast, our work incorporates a dynamic update mechanism that continuously integrates new threat data and is rigorously validated under real-world conditions, thereby enhancing adaptability and robustness against emerging cybercrimes.

Dunsin et al. [31] developed a multi-agent framework for digital investigations, highlighting reduced time for integrity checks of evidence files. Despite promising results, the framework would benefit from validation in various cloud environments. Our research addresses this gap by implementing and thoroughly evaluating our framework across diverse

TABLE 1. Summary of related works.

Reference	Data Source	Contribution	Novelty and Approach
Stelly & Roussev. [18]	Experimental Data	Containerisation is used to encapsulate individual executable modules . ExifTool and OpennSFW are used as worker modules	Propose a novel container-based framework which integrates existing forensics tools into a processing pipeline as worker modules.
Nanda & Hansen. [30]	Cloud Resources	Forensics as a Service with VM snapshots	Implement a Forensic as a Service (FaaS) solution, enabling digital forensics to be conducted efficiently through a cloud-based Forensic Server.
Dykstra & Sherman [24]	Virtual Disks, API logs	Openstack cloud platform for cyber incident response	Suggest a set of three novel forensic tools designed for the OpenStack cloud platform, ensuring the rustworthy acquisition of virtual disks, API logs, and guest firewall logs.
Philip et al. [41]	DNS logs	Multi-agent system with Decentralised Model	Multi-agent model for forensics investigation in domains where devices are often distributed across a wide area.
Rughani. [32]	Disk Images	Automate the Acquisition, Analysis and Presentation of Data for forensics	framework to optimise speed and performance in investigating cyber crimes and minimising user interactions.
Irina Baptista et al. [42]	Malicious and Benign files.	Malware detection based on binary visualisation using Neural Networks.	Describe a new approach to malware detection that combines machine learning with a creative method of visualising malware as images.
Temechu et al. [25]	Log files from CAIDA and Packt.	Data pre-processing, feature extraction using CNNs, and classification using SVMs	hybrid Machine Learning (ML) approach for anomaly detection in IoT and cloud environments using Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs) to address security threats.
Our Work	Network Traffic, HTTP Server logs, .exe files	Real-Time feature Engineering for Classification with Docker containers and Kubernetes in cloud environments using deep learning model to reduce false positives.	system with multiple applications deployed to defend against cyber threats and respond to incidents. The system can interact with large amounts of data by scaling and predicting with higher accuracy.

cloud platforms, ensuring its robustness and adaptability in real-world scenarios.

Dunsin et al. [16] explored the use of reinforcement learning (RL) to enhance malware investigation during cyber incident response. Their approach focuses on optimizing forensic decision-making through RL-based automation, improving efficiency and response times. While their method demonstrates potential, challenges such as the explainability of RL models, computational costs, and scalability in real-world deployments remain underexplored. Our research differs by integrating a multi-layered AI-driven cyber incident response system that includes post-incident malware analysis. Unlike reinforcement learning-based approaches, our system leverages a hybrid model combining Random Forest and deep learning (Keras TensorFlow) for malware detection, containerized deployment for scalability, and real-time feature extraction to ensure rapid and precise threat identification in cloud environments.

The reviewed literature showcased the growing significance of AI and ML in improving digital forensics and incident response capabilities while highlighting the need for empirical validations and practical implementations to realise their full potential in cloud environments. Table 1 summarises the most relevant related works and provides a comparison with our proposed system, demonstrating its unique advantages . Unlike other related works, our system integrates a honeypot environment, enabling proactive threat detection. Additionally, the system's versatility is demonstrated through its ability to operate on multiple public cloud platforms and process diverse data sources. Incorporating SIEM visualisation provides a holistic view of security events, enhancing overall system effectiveness. By combining these innovative features, the proposed system presents a more robust and adaptable solution to address contemporary security challenges [17].

IV. METHODOLOGY AND IMPLEMENTATION

In this section, we present the methodology employed in this research and outline the systematic approach used to achieve the study objectives. This section covers the design, development, and deployment of the system in detail.

A. SYSTEM DESIGN AND DEVELOPMENT

1) OVERALL SYSTEM

Our research proposes a novel AI-powered system with a three-tier architecture for efficient cyber threat detection and investigation. This architecture leverages containerisation technology to isolate and deploy various functionalities across three distinct environments: Production, Honeypot and DFIR, as illustrated in Figure 3. The production environment securely hosts critical infrastructure that the customer needs, ensuring production data's integrity, availability, and confidentiality. It also securely mirrors network traffic to the DFIR Environment VPC for analysis by AI models. The honeypot environment, a core component of our system's innovation, uses a T-Pot honeypot to strategically attract and deceive attackers [35]. This deception facilitates the collection of valuable training data for our AI models that continuously learn.

The DFIR environment acts as the central hub for analysis. It houses a suite of security applications consisting of trained models that will perform predictions on new data points, including a network attack classifier that performs real-time classification on network traffic. Furthermore, a storage bucket in the DFIR environment leverages the Malware

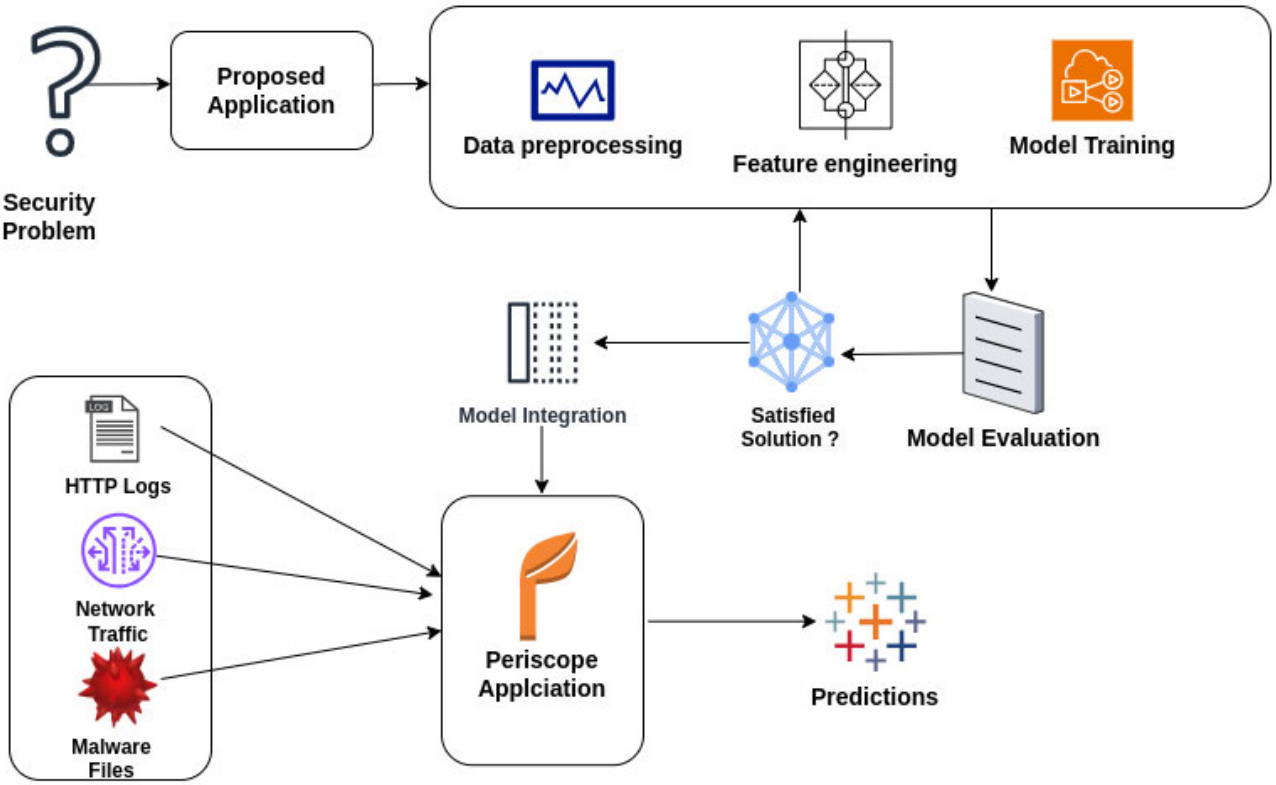

FIGURE 1. **Framework for data management and analysis.**

Analysis system (hosted on a separate subnet) to analyse suspicious files and perform static analysis. Subnet 3, considered the nerve centre, hosts the ELK stack (Elasticsearch, Logstash, and Kibana) for centralised storage and analysis of logs generated by the ML models. These logs enriched with insights from production and honeypot environments empower analysts to identify patterns and anomalies that might indicate potential threats. The final subnet, Research and Development, acts as a bridge between the honeypot environment and the system. Labelled training data from the honeypot and the computing power provided by the cloud facilitate a continuous model training and deployment pipeline, ensuring our AI models stay up-to-date with evolving threats.

Figure 1 shows the proposed framework for data analysis employed in our end-to-end AI system construction, which served as the guiding structure for this research. This ten-step process, beginning with defining the business problem and culminating in the deployment of the trained model, was utilised to develop the AI system presented here. Following data selection and collection, the framework emphasised data pre-processing and feature engineering to prepare the information for model training and evaluation. An iterative loop was adopted, where model performance was assessed and potentially required revisiting earlier stages in the framework for refinement. In subsequent subsections, we also investigate various techniques for deploying AI applications within a secure and efficient architecture. Table 8 summarises the specific security problems addressed and the corresponding algorithms employed.

The data flow diagram presented in Figure 2 shows the connectivity and multiple pipelines implemented in the system. The system design allows for centralised management through the cloud infrastructure, enabling efficient monitoring and analysis. This architecture supports ongoing research and development, contributing to improved threat detection and response capabilities as shown in Figure 3.

2) THE NETWORK TRAFFIC CLASSIFIER

Securing critical infrastructure is paramount and network security plays a vital role in this endeavour. Attackers often exploit vulnerabilities within network systems, making network traffic analysis a crucial tool for defence. Network traffic is a rich source of data that contains valuable information on the ongoing activity of the network illustrated in Table 2. To effectively identify and mitigate potential threats, we propose the development of a network traffic classifier.

The algorithm 1 illustrates the functioning of Network Traffic Classification. This classifier will leverage real-time network traffic capture, allowing for online analysis of network activity.

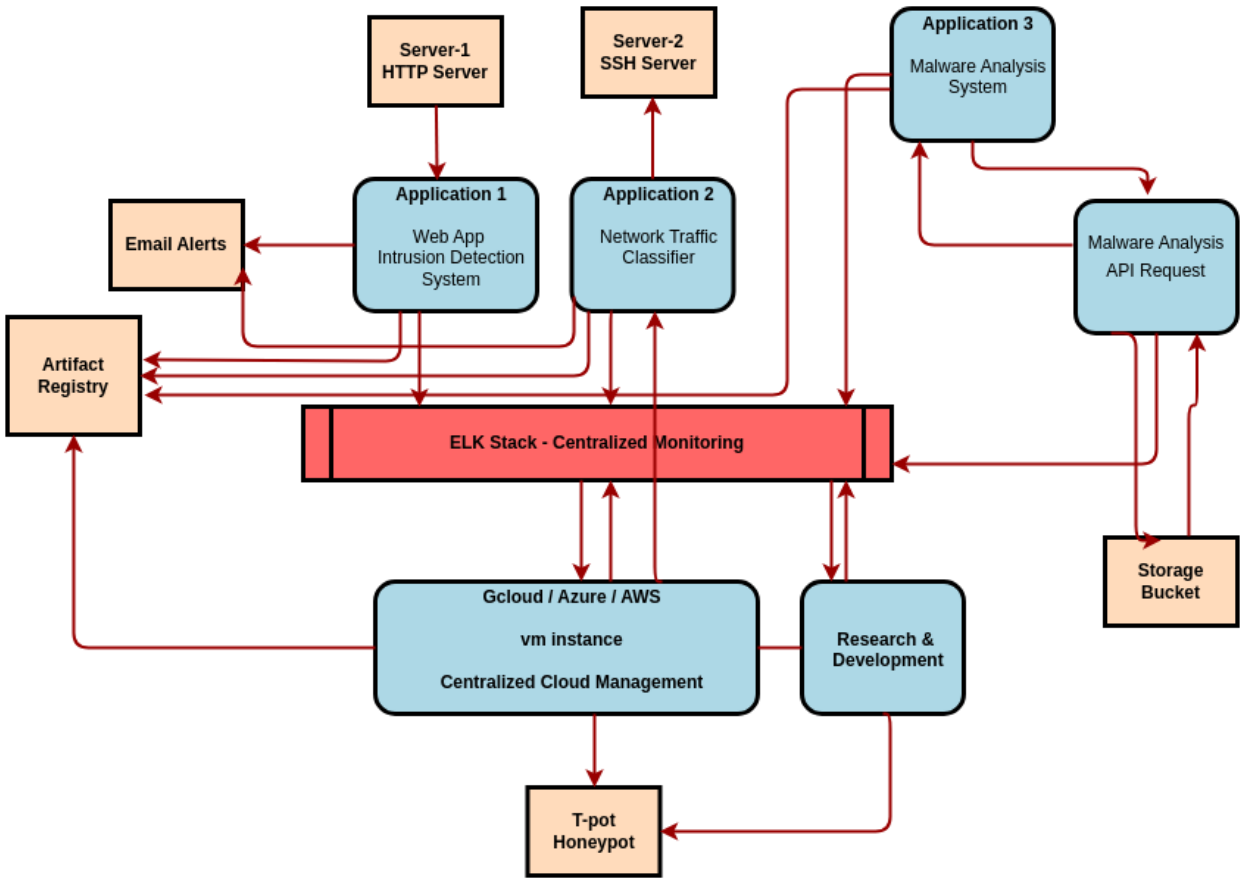

FIGURE 2. Proposed system data flow diagram.

TABLE 2. Performance comparison of algorithms for traffic classification.

Algorithm	NSL-KDD	NB15	CIC-IDS
Random Forest	90.92%	75.64%	99.82%
Logistic Regression	85.75%	63.16%	96.2%
Decision Tree	87.13%	73.95%	99.76%
KNN	85.34%	60.33%	99.22%
Naive Bayes	51.58%	41.98%	69.60%

NSL-KDD is a classic benchmark dataset with 41 features and 38 attack categories, capturing both connection-based and content-based attributes. It remains a valuable resource for detecting traditional intrusion types, such as DoS, R2L, U2R, and Probe attacks. CIC-IDS 2017 [11] represents modern real-world traffic with 80 characteristics that cover both flow-based and content-based characteristics. It includes contemporary attack types such as Brute Force, DDoS, and Infiltration, making it highly relevant for evaluating current threats. UNSW-NB15 [12] introduces 49 features that reflect a hybrid testbed environment, capturing complex attacks such as Fuzzing, Backdoors, and Shellcode. This data set allows the classifier to handle next-generation sophisticated cyber threats with the feature categories described in Table 3. By training the classifier on each dataset separately, we assess its performance and adaptability across a wide range of attack vectors and traffic environments, ensuring a comprehensive evaluation of its capabilities, as detailed in Algorithm 2.

Algorithm 1 Network Traffic Classification Using Random Forest

Require: Training data: Network traffic dataset $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$
where $\mathbf{x}_i$ is a feature vector and y_i is the attack type label.
Ensure: Traffic Classification model M
Pre-process data
Read data from CSV files.
Drop irrelevant features (e.g., flags, protocols, services).
Separate features ($\mathbf{X}$) and labels ($\mathbf{y}$).
Split data into training and testing sets: (X_{train}, y_{train}), (X_{test}, y_{test}).
Standardize features using StandardScaler.
Build Random Forest model M
Define a Random Forest classifier with a desired number of estimators (e.g., 100).
Set random state for reproducibility (e.g., 42).
Train model M on (X_{train}, y_{train}).
Evaluate model M on (X_{test}, y_{test}) using metrics (e.g., accuracy, classification report).
return Trained Traffic Classification model M

TABLE 3. NSL-KDD dataset feature categories with general descriptions.

Feature category	Features	Description
Basic Connection	duration, protocol_type, service, flag, src_bytes, dst_bytes, land	Basic network connection attributes.
Host-Based	wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, is_host_login, is_guest_login	Characteristics related to the behavior of the host system during the connection.
Traffic	count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate	Statistical features and connection rates between hosts and services.

Algorithm 2 Cloud-Based Malware Classification With LSTM

Require: Training data: executable file $(d_1, l_1), (d_2, l_2), \ldots, (d_N, l_N)$

where d_i is a file and l_i is its label. Number of classes: C

Maximum vocabulary size: V

Maximum sequence length: T

Ensure: Trained malware classification model M

Preprocess text data:

Tokenize files (d_i) into sequences of integers $(w_i^1, w_i^2, \ldots, w_i^{|d_i|})$.

Pad sequences to uniform length $(w_i^1, w_i^2, \ldots, w_i^T)$.

Encode labels: Convert text labels (l_i) to numerical labels (y_i).

Split data into training and testing sets: (X_{train}, y_{train}), (X_{test}, y_{test}).

Build LSTM model M:

Define a sequential model.

Add Embedding layer with vocabulary size V and embedding dimension E.

Add LSTM layer with hidden units H.

Add Dense layer with C output units and sigmoid activation for C classes.

Compile model M:

Set optimizer (e.g., Adam).

Set loss function (binary_crossentropy for binary classification, categorical_crossentropy for multi-class).

Set metrics (e.g., accuracy).

Train model M on (X_{train}, y_{train}) for a desired number of epochs.

Evaluate model M on (X_{test}, y_{test}) using metrics (e.g., accuracy).

return Trained malware classification model M

By effectively classifying incoming network traffic in real-time based on this rich feature set, our classifier can significantly enhance network security by enabling prompt identification and mitigation of potential cyberattacks. Unlike existing monolithic solutions, this novel approach to network traffic classification by containerizing key components into distinct engines: the Capture Engine, Feature Engine, and Model Classifier, offers modularity and flexibility, enabling users to independently scale and deploy each component according to their specific requirements. This approach streamlines feature extraction by automating feature engineering from PCAP files, facilitating seamless integration with diverse network environments. The network traffic classifier application encompasses a comprehensive and modular design workflow tailored to handle the complexities of real-time network traffic analysis within cloud environments. The logical design unfolds through three interconnected components, each contributing crucial functionalities to the overall system. Firstly, the packet capture engine serves as the foundational component, leveraging the Scapy library to capture network packets continuously. Operating within its designated container, this engine listens on specified interfaces, intercepts network traffic, and stores captured packets in PCAP files for subsequent analysis. Secondly, the packet analysis module, encapsulated within another container, reads the captured PCAP files, extracts connection-based statistics, and transforms them into structured datasets suitable for predictive modelling. Using the pre-trained machine learning model during the development phase, it predicts predefined attack labels. Finally, integrated into a separate container, the alerting mechanism monitors prediction outcomes and triggers alerts in real-time upon detecting anomalous network behaviour. These alerts serve as actionable insights for security analysts, enabling timely responses to potential threats. This containerised architecture ensures scalability, flexibility, and isolation, facilitating seamless deployment, management, and scalability of the network traffic classifier application within diverse computing environments.

3) WEB INTRUSION DETECTION SYSTEM

This research implements a web intrusion detection system leveraging real-time anomaly detection techniques. Anomaly detection, in the context of network and host security, identifies unusual activities that may signify an attacker's presence. However, traditional anomaly detection often struggles to define ''normal'' behaviour, leading to a high rate of false alarms. This research aims to address this challenge by employing AI for anomaly detection within the specific domain of web application security. The system focuses on extracting informative features from standard HTTP server logs.

Although these logs offer only a partial view of the overall network traffic, they still contain key features that can be extracted, as illustrated in Table 4. Building a dependable and comprehensive feature set is essential for the effectiveness of the anomaly detection process. By examining these characteristics, the system can detect anomalies that can signal various

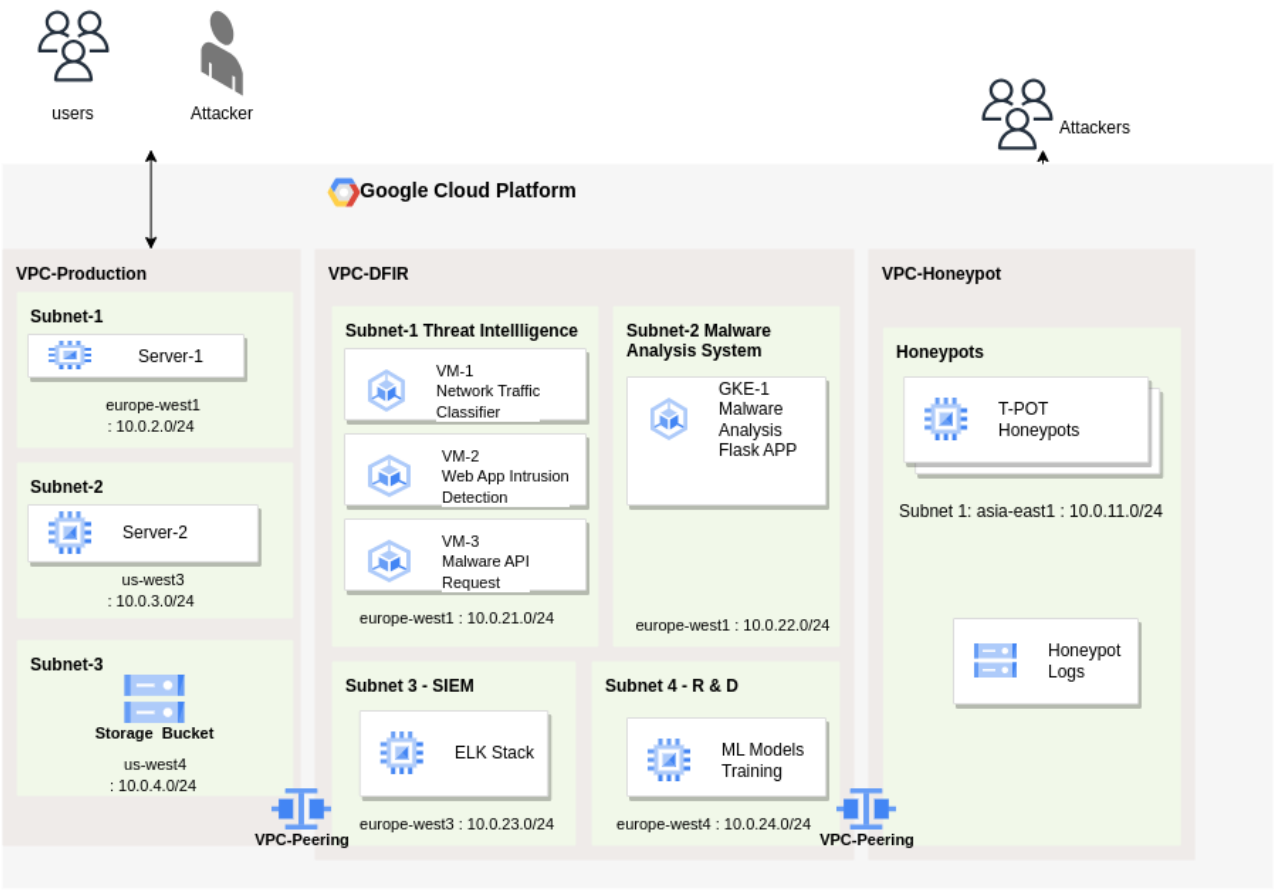

FIGURE 3. **Proposed system detailed architecture.**

types of attacks, including those identified in the OWASP Top Ten [46].

The key novelty of this design lies in its real-time deployment and distributed data collection. Each web server can deploy a lightweight agent responsible for collecting and forwarding logs promptly to the Web Intrusion Detection System (WIDS). This agent-based approach ensures efficient log collection and minimises the impact on individual web servers. The WIDS utilises a shared volume accessible by the virtual machine (VM) and the deployed container. This shared volume facilitates efficient storage and access to the collected logs for real-time analysis.

Anomaly detection identifies unusual activity that deviates from established patterns of normal behaviour. Common approaches include:

- Statistical Metrics: Analysing deviations from statistical properties like mean, standard deviation, or frequency distribution.
- Unsupervised Machine Learning: Employing algorithms that learn patterns from unlabelled data to identify outliers.
- Goodness-of-Fit Tests: Evaluating how well a data sample fits a pre-defined statistical model.
- Density-Based Methods: Identifying anomalies as data points located in low-density regions of the feature space.

This application leverages the Isolation Forest algorithm due to its effectiveness in high-dimensional datasets often encountered in real-world security applications. The isolation forest works by isolating potential anomalies by randomly partitioning the data. Instances that are easier to isolate are likely anomalies, while those requiring deeper partitioning are more likely to be normal data points. A critical aspect of anomaly detection is defining the threshold for flagging data points as anomalies.

In this web intrusion detection system, optimising the hyperparameters of the Isolation Forest algorithm was essential to achieve reliable anomaly detection using server log files. Initially, we set the sample size (*num_samples*) to 1,000, which defines the number of data points used to build each isolation tree. This value was later increased to 10,000 to evaluate scalability and enhance performance, significantly improving the model's ability to detect outliers in larger datasets. The random state parameter (*random_state*), responsible for reproducibility, was refined through multiple tests. After experimenting with various values, it was

TABLE 4. Features generated from HTTP server logs.

Feature Name	Description
1-IP-level access statistics	High frequency, periodicity, or volume by a single IP address or subnet is suspicious.
2-URL string aberrations	Self-referencing paths (/./) or backreferences (/../) are frequently used in path-traversal attacks.
3-Unusual referrer patterns	Page accesses with an abnormal referrer URL are often a signal of unwelcome access to an HTTP endpoint.
4-Sequence of accesses to endpoints	Out-of-order access to HTTP endpoints that do not correspond to the website's logical flow
5-User agent patterns	Alerts on never-before-seen user agent strings or extremely old clients which are likely spoofed.

optimised to 100, ensuring consistent model training without impacting overall performance. A novel approach was used to determine the optimal contamination ratio by simulating specific attack scenarios (e.g., path traversal attack) and analysing the model detection rate. This technique leverages domain knowledge about potential attack vectors to fine-tune the contamination threshold for real-world applications. We tested contamination levels of 0.001, 0.01, and 0.1. After extensive testing, a contamination ratio of 0.01 provided the best balance between anomaly detection sensitivity and minimising false positives, making it the most effective for this system. More detailed test results are provided in the Results section. The systematic optimisation of these hyperparameters led to a model that successfully balances computational efficiency and detection accuracy, tailored for real-time web intrusion detection in dynamic server environments.

Due to the inherent difficulty of acquiring perfectly representative datasets for all scenarios, continuous monitoring and adaptation are essential for robust security. This application prioritises simplicity and efficiency by using a single container. The container performs the following tasks:

- Reads HTTP access logs from web servers.
- Pre-processes the log data by converting it to a pandas data frame for manipulation in Python.
- Generates features mentioned in Table 4 for anomaly detection from the log data.
- Sends the extracted features to the trained Isolation Forest model for real-time anomaly detection.

To reduce false positives, the application triggers alerts only when the number of detected anomalies exceeds a predefined threshold. This approach is based on the assumption that real-world attacks often involve rapid bursts of activity, leading to a surge in detected anomalies. This design demonstrates a practical approach to deploying AI-powered anomaly detection for web intrusion detection in real time. By leveraging Isolation Forest and a domain-informed threshold determination method, the system aims to achieve efficient and accurate anomaly detection. The algorithm 3 illustrates the Web intrusion detection model using HTTP access logs.

Algorithm 3 Web Server Logs Anomaly Detection Using Isolation Forest

Require: Web server log file L
Ensure: Anomaly detection model M
Preprocess data:
  Read log file L into a DataFrame D.
  Remove missing values from D.
Extract features:
  IP-level statistics: ip_frequency, Unique_connections_count, ip_volume
  URL aberrations: url_aberrations
  Unusual referrer patterns: unusual_referrer
  User-Agent analysis: user_agent_analysis (categorical)
  Out-of-order access: out_of_order_access
  Standardize numerical features using StandardScaler.
Train Isolation Forest model:
  Create an IsolationForest model M with:
  Max samples: N (number of samples in D)
  Contamination: c (estimated outlier ratio)
  Random state: r (for reproducibility)
  Train M on features X in D.
Detect anomalies:
  Use M to predict anomaly scores y_pred for new data points.
  Mark data points with scores below a threshold as anomalies.
return Anomaly detection model M

TABLE 5. Malware dataset.

Dataset		Samples	
Type	**Purpose**	**Count**	**Percentage**
Benignware	Training	694	70%
	Testing	297	30%
Malware	Training	299	70%
	Testing	129	30%

4) MALWARE ANALYSIS SYSTEM

Malware analysis, which encompasses the investigation of malicious software's functionality, purpose, origin, and potential impact, traditionally requires extensive manual effort and expertise in software internals and reverse engineering. Our research introduces a novel application that streamlines this process, offering efficient and automated malware analysis capabilities.

The application development commenced with the implementation of the code to extract features from training binaries. Feature extraction involves identifying and collecting pertinent data from training binaries, which are then stored within a Python dictionary. Detailed information on the dataset used in the study is provided in Table 5. The dataset used in this investigation, sourced primarily from VirusTotal, presents both advantages and limitations that affect its applicability in real-world scenarios. Although VirusTotal offers a diverse collection of malware samples, it may be

biased toward more commonly reported threats, potentially neglecting emerging or niche malware types. Additionally, the temporal relevance of the dataset is crucial, as more samples may be needed to maintain the model's effectiveness against current threats. The ratio of benign to malicious samples can impact model performance and may result in a model that over-fits malicious samples, affecting its ability to generalise well to real-world scenarios.

Regarding generalisability, the effectiveness of the model can vary between different operating systems, network conditions, and user behaviours, highlighting the need for evaluation in diverse environments. Malware behaviour can vary significantly between different operating systems, application environments, and network configurations. For example, a model trained primarily on Windows malware might perform less effectively on macOS or Linux systems, where malware characteristics and user behaviour differ. To enhance practical applicability, ongoing efforts to diversify the training dataset and adapt to evolving threats are essential to maintain robust malware detection capabilities. Incorporating a detailed analysis of the dataset, addressing its limitations, and discussing the generalisability of the model across different environments enhances the robustness of the research.

Subsequently, a model was trained using the extracted string features from the samples. In addition to string features, the malware analysis system has integrated behavioural analysis to effectively counter the stealthy tactics employed by advanced persistent threats (APTs). This includes examining the features of portable executable (PE) headers, Import Address Table (IAT) characteristics, and the identification of callback servers. These elements play a critical role in recognising the subtle behaviours associated with sophisticated attacks. As part of future work, the malware analysis system can further enhance its detection capabilities by incorporating contextual threat intelligence about known APT groups, including their tactics, techniques, and procedures (TTP) and associated indicators of compromise (IoC). This integration will enable the system to make more informed decisions based on the behavioural patterns and attributes shared by various malware samples, such as embedded IP addresses, hostnames, strings of printable characters, and graphics.

By training the detection model on malware used by APT groups, it can evolve beyond a binary classifier, providing a more nuanced understanding of threats. This holistic approach ensures that the malware analysis component is better equipped to address the complexities of APTs, ultimately enhancing its real-world applicability and effectiveness in defending against sophisticated cyber threats.

Various algorithms were explored during the training phase, and the most suitable was selected based on evaluation metrics such as precision, precision, recall, and the F1 score. To facilitate comparison, Table 6 presents a summary of the performance of three different algorithms used in the study.

The models were trained using Google Colab with T4 GPU acceleration, optimising computational efficiency. In particular, the application adopts a combined model architecture to effectively mitigate false positives. The primary model

TABLE 6. Performance comparison of algorithms for malware analysis.

Algorithm	Accuracy	Precision	Recall	F1-Score
Random Forest	96.71%	94.44%	94.44%	94.44%
Support Vector Machine	91.54%	95.91%	74.60%	83.92%
Logistic Regression	94.60%	97.24%	84.12%	90.21%
Decision Tree	88.26%	87.25%	70.63%	78.07%

in this architecture is a random forest model complemented by a secondary Keras TensorFlow model. Both models were trained on a dataset obtained from VirusTotal.com, ensuring comprehensive coverage of malware samples. The integration of a deep learning model as the secondary component aims to reduce latency in real-time applications, enhancing system responsiveness, as summarised in Figure 4.

- Random Forest Model: This model provides a relatively fast initial classification with good accuracy.
- Keras TensorFlow Model: This deep learning model offers additional refinement, particularly for complex malware samples.

The deep learning model employs a Long Short-Term Memory (LSTM) network, a type of Recurrent Neural Network (RNN) well-suited for analysing sequential data like text as explained in Table 7. Table 6 summarises the architecture of the neural network used in the malware analysis system, detailing each layer's function, the number of neurons, and the activation functions applied. The model begins with an embedding layer that converts discrete words into dense vector representations (32 dimensions in this case). An LSTM layer with 32 hidden units then processes these sequences. Unlike the embedding layer, each hidden unit within the LSTM can be considered a ''neuron'' that learns and extracts features from the data as summarised in Table 7. Finally, a dense layer with a single output neuron and a sigmoid activation function performs binary classification. This single output neurone leverages the features learnt by the LSTM layer to predict the maliciousness of the analysed file. The Algorithm 2 illustrates the cloud-based malware classification using the LTSM network.

The system, as illustrated in Figure 3, follows a logical flow that begins when the user uploads a file through the application's user interface. Upon submission, the uploaded file undergoes initial processing, where its features are extracted and prepared for classification. Using a pre-trained machine learning model, the system predicts the probability of the file being malicious. If the probability exceeds a threshold of 0.7, indicative of a high likelihood of malicious content, the system immediately classifies the file as ''Malicious'' and generates a detailed analysis report in PDF format. In contrast, if the probability falls between 0.5 and 0.7, the system invokes a Keras deep learning model for further binary classification. Upon completion of the classification, the system produces a comprehensive analysis report, facilitating informed decision making regarding the security implications of the file. For files deemed benign on the basis of the classification results, the system provides a ''Benign''

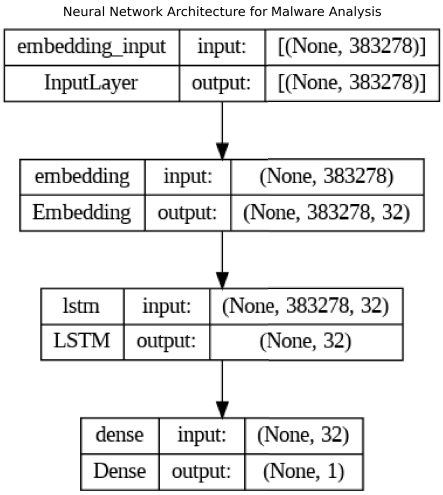

FIGURE 4. **Keras model parameters.**

classification and presents the analysis outcome to the user through a web interface. This approach combines the power of cloud computing, deep learning, and web technologies, offering a robust solution for real-time malware detection and analysis with enhanced user accessibility and efficiency.

The versatility of the application enables its utilisation in diverse scenarios. Firstly, it serves as a web service, allowing online users to upload executable files for malware analysis. The application quickly determines the maliciousness of the files and provides results to users. Secondly, it functions as an API, facilitating the scanning of storage buckets deployed in cloud environments. This multi-faceted approach ensures the broad applicability and practicality of the developed malware analysis solution, catering to varying user requirements and deployment environments.

B. SYSTEM IMPLEMENTATION

While prior research has explored the potential of AI/ML for cybersecurity, most efforts remain theoretical with limited real-world deployments. This research bridges the gap by implementing a trained AI-powered cybersecurity system in a real-time cloud environment. To ensure efficient and scalable deployment, the system leverages containerization technologies and is hosted on public cloud platforms such as Google Cloud and Microsoft Azure.

1) DEPLOYMENT STRATEGY

This research deployed three AI applications in a real-world cyber security environment to bridge the gap between theoretical frameworks and practical implementation.

- Web Intrusion Detection: A single container deployed on a virtual machine (VM) analyses the Web server access logs for anomalies. This simple deployment prioritises ease of maintenance for this initial application.
- Malware Analysis: A Flask application, deployed on Kubernetes Engine, classifies uploaded executable files (exe) as benign or malware. This containerised approach allows for easier scaling and updating of the model as needed.
- Network Traffic Classifier: This application, deployed using multiple Docker containers on a single Virtual machine, analyses network traffic captured as PCAP data for real-time attack detection. Containerisation again facilitates model updates and simplifies deployment.

The deployment strategy leverages containerisation (e.g., Docker) for several advantages:

- Scalability: Containers enable for easy scaling of computational resources to meet changing demands.
- Simplified Updates: Containerised models streamline updates, reducing downtime and improving maintenance efficiency.
- Portability: A single trained model can be deployed across diverse cloud environments by pulling the container image from a central registry like Docker Hub. This simplifies multi-cloud deployments.

In the implementation of the system, data transmission is handled with a focus on security and efficiency. Each web server is equipped with an agent that periodically transfers HTTP server logs to its designated system container using the Secure Copy Protocol (SCP). This ensures secure and encrypted transmission of logs, protecting sensitive data during transfer. For the network traffic classifier, packet mirroring is used. All network traffic, including packet payloads and headers destined for the application server, is mirrored to the virtual machine (VM) running the network traffic classifier. This approach ensures comprehensive monitoring of all traffic, allowing the system to detect anomalies, security breaches, and intrusions effectively. By mirroring all traffic, the network traffic classifier can perform thorough inspections across multiple packets, ensuring that no malicious activity is missed.

However, it is important to acknowledge the potential challenges associated with deployment. Although containerisation offers a robust deployment strategy, real-world implementation presents potential challenges. A critical factor is ensuring the throughput of the deployed model meets the demands of real-time operation. The hardware platform of the production system (CPU, GPU, memory) must provide sufficient computational resources to handle processing demands. Careful consideration of these resource requirements is essential to avoid bottlenecks and maintain the effectiveness of the AI system in a real-time cybersecurity environment.

The deployment of the system begins with the Network Traffic Classifier, initially deployed on a single VM instance to analyse captured PCAP data in real time. To ensure scalability and prevent potential bottlenecks as traffic volume increases, the system will implement horizontal scaling by using a load balancer. This approach will distribute the network traffic across multiple VM instances, ensuring efficient and balanced processing. Vertical scaling will be employed for the Web Intrusion Detection application, with each web instance being assigned its own dedicated VM. This ensures

TABLE 7. Neural network architecture.

Layer	Function	Neurons	Activation Function
Embedding	Converts words to dense vectors	max_words (adjustable)	N/A (vector representation)
LSTM	Captures long-range dependencies in sequences	32	Sigmoid, tanh, or ReLU (default: sigmoid)
Dense	Performs final binary classification	1	Sigmoid

TABLE 8. Application scalability.

Application	Scaling Method	Load Management
Network Traffic Classifier	Horizontal Scaling	Load balancer distributes network traffic across multiple VMs to prevent bottlenecks.
Web Intrusion Detection	Vertical Scaling	Individual VMs handle anomaly detection for specific web servers, resource allocation is based on web traffic.
Malware Analysis System	Horizontal Scaling:	Load balancer distributes file uploads across multiple containers to ensure smooth handling of a large number of requests

TABLE 9. Attack simulation results.

Attack Type	Attack Tool/Method	Model Used	Results & Comments
Brute Force Attack	Hydra (SSH/FTP)	Random Forest - (CICIDS2017)	Low FP, High accuracy
Denial of Service (DoS)	Nikto (SYN Flood)	Random Forest - (NSL-KDD)	Moderate Detection, few FPs
Fuzzing	wfuzz (Web App Fuzzer)	Random Forest - (UNSW-NB15)	High Detection with consistency
Port Scan	Nmap	Random Forest - (CICIDS2017)	Consistently detected
SQL Injection	sqlmap	Random Forest - (CICIDS2017)	Detected quickly, few FNs
Exploits	Metasploit Framework	Random Forest - (UNSW-NB15)	Lower Detection during attack
R2L FTP write	php web shell	Random Forest - (NSL-KDD)	Low detection, High FPs

focused resource allocation and simplified maintenance, particularly for handling the growing volume of web server access logs. Finally, the Malware Analysis application will utilise a load balancer to distribute file uploads across containers, with an additional node dedicated to API handling. Table 9 presents a summary of application scalability within the cloud environment. This setup optimises scalability and performance, ensuring seamless file classification and system responsiveness even under heavy workloads illustrated in Table 8.

2) CLOUD DEPLOYMENT COSTS AND OPTIMIZATION

Deploying the proposed AI-enabled cybersecurity system on platforms like Google Cloud and Azure involves costs related to compute resources, storage, and data transfer. In Google Cloud, using NVIDIA T4 GPUs costs approximately $0.35/hour, while TPUs range from $4.50 to $8.00/hour, with preemptible VM instances offering up to 70% cost savings. Similarly, Azure's NC T4 v3 series starts at $0.90/hour, with spot VMs providing budget-friendly alternatives. Storage solutions in Google Cloud start at $0.04/GB/month for persistent disks and $0.01/GB/month for archival storage, while Azure offers comparable options at $0.018/GB/month for standard disks and $0.005/GB/month for blob storage. Data transfer costs are competitive, with ingress being free and egress starting at $0.08/GB on Google Cloud and $0.05/GB on Azure. AI-specific services such as the Google AI Platform and Azure Machine Learning provide managed infrastructure and automation, enabling efficient scaling and cost control. For medium-scale deployments involving GPU training, real-time monitoring, and malware analysis, monthly costs are estimated at $500–$1,500, which can be reduced by up to 30% with strategies such as preemptible instances, low-cost storage tiers, and managed services. Table 11 provides a comprehensive overview of the hardware configurations, cloud resources, and software dependencies used in our implementation, including CPU/GPU specifications, memory requirements, and container resource allocations to support reproducibility.

By leveraging cloud scalability, containerization, and infrastructure automation, the proposed system ensures efficient deployment, seamless model updates, and high availability. Cloud-based machine learning services further enhance resource optimization, enabling real-time AI-driven cybersecurity solutions with minimal operational overhead.

V. TESTING AND RESULTS

A. NETWORK TRAFFIC CLASSIFIER

During development, various machine learning models were evaluated on the NSL-KDD, CIC-IDS 2017, and UNSW-NB15 datasets to identify the most effective approach for classifying network traffic patterns. The Random Forest algorithm emerged as the frontrunner across all three datasets, achieving accuracy rates of 90.92%, 99.82%, and 75.64%, respectively. This significantly outperformed other models, including Logistic Regression, Decision Tree, KNN, and Naive Bayes, as shown in Table 2. In optimising the Random Forest classifier for network traffic classification, specific hyperparameters were carefully tuned across all three models to improve detection performance.

We used Gini impurity as a criterion to measure the quality of the splits, ensuring that the model could effectively distinguish between classes. The minimum sample split was set to 2, allowing the trees to split nodes even with the smallest number of samples, thereby refining decision boundaries for improved accuracy. The maximum sample parameter was set

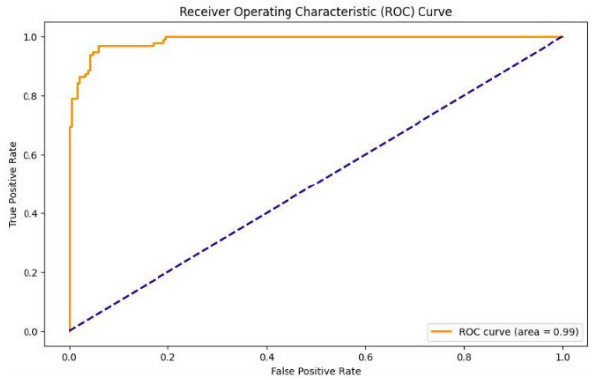

FIGURE 5. Keras model ROC curve.

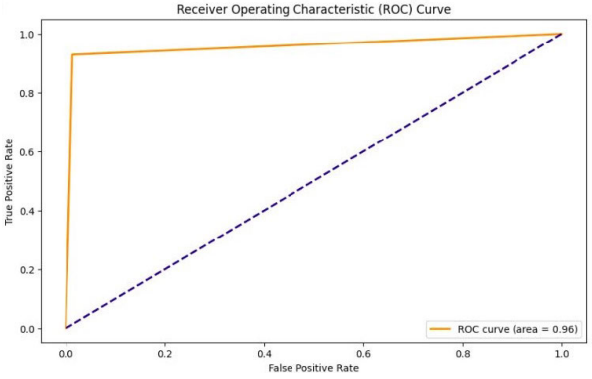

FIGURE 6. Random forest ROC curve.

to None, ensuring that the models were trained in the entire training set as illustrated in Figures 5 and 6. Additionally, a random state was specified for reproducibility, ensuring consistent results across multiple runs.

The NSL-KDD dataset, consisting of 125,973 training samples and 22,544 testing samples, was used with its predefined split to maintain consistency with previous studies. In the case of the CIC-IDS 2017 dataset, which contains approximately 2.8 million samples, the data was randomly divided into 80% for training (around 2.24 million samples) and 20% for testing (around 560,000 samples), ensuring a fair distribution of attack types and normal traffic in both sets. Similarly, the UNSW-NB15 dataset, with 175,341 training samples and 82,332 testing samples, was also used with its predefined split. The systematic optimisation of these hyperparameters improved the model's ability to accurately classify network traffic, making it robust in detecting network anomalies.

Following the deployment of the Network Traffic Classifier application in cloud environments using a modular, containerised approach, featuring separate containers for packet capturing and real-time classification of incoming traffic data, preprocessing and feature engineering were performed within a single container, streamlining the workflow before classification. The attack simulations were then performed on the application to evaluate its performance.

Table 9 summarises the results obtained from these attacks, detailing the methods used, the corresponding machine learning models, and their performance in detecting the attacks. The results highlight the models' performance against specific attacks relevant to the datasets on which they were trained. A comprehensive evaluation was conducted to assess the real-world capabilities of the system by simulating various cyber attacks on resources purposely made vulnerable and hosted within cloud environments. The testing was carried out using a Kali Linux environment, employing a variety of penetration testing tools to conduct attacks both remotely and within the cloud infrastructure. These attacks were executed at different stages of the cyber-kill chain to evaluate the effectiveness of trained models against specific threat vectors. Overall, the evaluation process effectively demonstrated the system's strengths and weaknesses in detecting various cyber threats, emphasising the importance of continuous refinement and training of the detection models to enhance their efficacy in real-world applications.

B. WEB INTRUSION DETECTION

Our research investigated the effectiveness of anomaly detection for web intrusion prevention. A key parameter, contamination, was explored to determine the model's sensitivity to anomalous traffic patterns. We evaluated different contamination levels (0.01 and 0.001) to assess the ability of the model to differentiate normal website traffic from potential attacks illustrated in Figures 7 and 8.

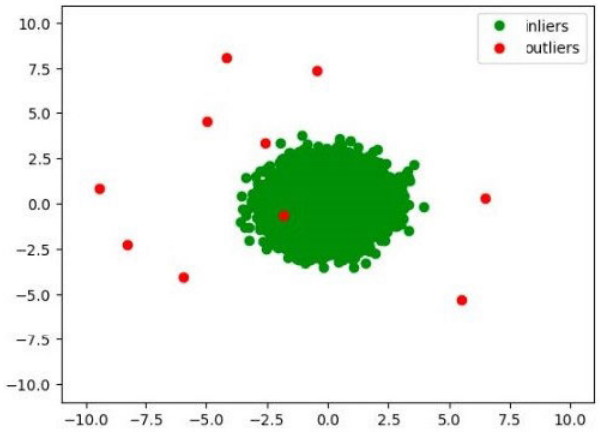

FIGURE 7. Contamination 0.01.

To determine the optimal contamination ratio for anomaly detection, we conducted an ablation study analyzing its impact on precision, recall, and false positive rates. As shown in Table 13, a strict threshold (0.001) minimized false positives (2.1%) but at the cost of recall, leading to missed attack detections (82.5%). Conversely, a more lenient setting (0.1) improved recall (97.5%) but significantly increased false positives (21.4%), generating excessive alerts. The intermediate contamination ratio of 0.01 provided the most balanced trade-off, achieving a recall of 91.3% while keeping false positives at a manageable 5.2%. Based on this evaluation,

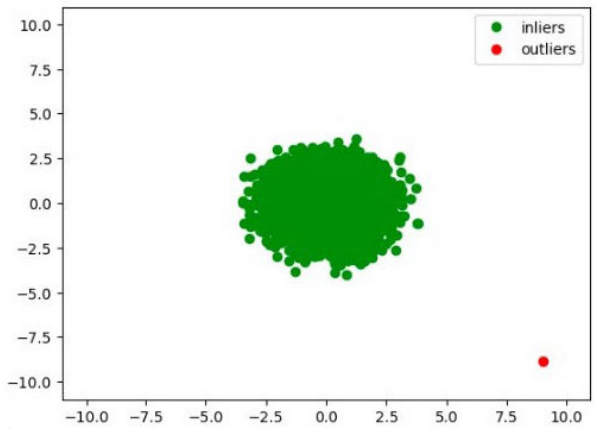

FIGURE 8. **Contamination 0.001.**

0.01 was selected as the optimal threshold for effective anomaly detection.

Following model training with features extracted from HTTP logs (as detailed in the methodology section), the system was integrated and deployed on the Azure cloud environment. A real-world simulation involved generating live traffic targeting a public website's IP address. The system effectively detected a simulated directory traversal attack launched using the Nikto tool on port 80. This detection triggered immediate logging and email notification, demonstrating the system's proficiency in real-time intrusion identification and response.

In addition to simulated attacks, the system underwent traffic load testing to evaluate its response under various conditions. A high volume of benign traffic was generated, during which the system maintained a high detection rate for regular activities but produced a few false positives, indicating areas where feature engineering could reduce noise in benign environments. These findings emphasise that fine-tuning the web intrusion detection model should be closely aligned with the specific application being hosted and the associated business requirements. By understanding the structure, functionality, and expected usage patterns of the application, we gain valuable information on the types of attacks that are more likely to occur. This understanding allows us to generate customised features that improve the model's ability to detect these specific threats, resulting in higher precision and relevance in identifying malicious behaviour.

C. MALWARE ANALYSIS SYSTEM

Following evaluation of performance metrics, Table 6 provides a summary of the algorithms tested in the malware analysis system along with their respective evaluation metrics, including accuracy, precision, recall, and F1 score. This table offers a comparative overview, highlighting the strengths and weaknesses of each algorithm in detecting malware accurately and effectively, aiding in the selection of the most suitable model for specific malware analysis needs.

Furthermore, Figures 5 and 6 illustrate the ROC curves for the Random Forest and Neural Network algorithms, respectively. These curves provide a visual representation of each model's true positive rate against the false positive rate, helping to assess the effectiveness of each algorithm in distinguishing between malicious and benign files. The ROC curves further validate the performance of these models, with a higher area under the curve (AUC) indicating stronger predictive capabilities.

The Random Forest algorithm exhibited an accuracy of 96%, while the Keras model achieved an even higher accuracy of 99%. Early testing revealed limitations in both models: Random Forest was computationally efficient but prone to false positives, whereas the Keras deep learning model demonstrated superior generalization but required significantly higher computational resources, leading to increased inference latency. To address these trade-offs, a hybrid approach was implemented, leveraging the strengths of both models. To validate its effectiveness, we conducted an ablation study comparing the contributions of Random Forest and Deep Learning individually. As shown in Table 12, while RF offers interpretability and low computational cost, it suffers from a higher false positive rate. Conversely, deep learning alone achieves better generalization but at the expense of increased resource consumption. The hybrid model balances these strengths, reducing false positives while maintaining high accuracy and optimizing inference time.

Additionally, the dual-model malware analysis was deployed as a Flask Application. Practical testing involved uploading an IRC Bot executable file to simulate real-world scenarios. The system accurately predicted the presence of malware and generated a detailed PDF report of static analysis. Furthermore, the system successfully identified benign applications, such as an executable Minesweeper game file.

Further testing included automating scans on storage buckets within the cloud, enabling the system to scan and identify any malicious files stored within these environments. These scans can be configured according to administrator preferences, providing flexibility according to organisational needs. The system API was tested by sending files to an endpoint and receiving a JSON response that details whether the file was classified as malicious or benign. This API integration allows the malware analysis system to be used in conjunction with other programmes, enhancing interoperability and scalability across different environments. The dual-model approach, coupled with successful practical testing, highlights the malware analysis system's ability to reliably identify and report on malicious entities while maintaining high accuracy in distinguishing benign software.

D. CONTROLLED TESTBED EVALUATION

To supplement the benchmark dataset evaluation and provide a practical assessment of the proposed system's real-world effectiveness, we conducted a series of controlled attack simulations within a cloud-based testbed environment. This testbed was designed to closely mimic production

environments by leveraging containerized microservices, network segmentation, and real-time network traffic monitoring.

The testbed infrastructure consisted of virtualized cloud instances hosting the AI-powered cybersecurity system, with designated environments for network traffic analysis, web intrusion detection, and malware classification. Attack simulations were executed using penetration testing tools such as *Kali Linux*, *Metasploit*, and *Hydra*, targeting vulnerabilities aligned with the **MITRE ATT&CK framework**. The scenarios included:

- **Credential-based attacks**: brute force via SSH/FTP.
- **Network-layer intrusions**: port scanning, DoS/DDoS simulations.
- **Application-layer threats**: SQL injection, web shell deployments.
- **Privilege escalation and persistence mechanisms**.

The system demonstrated strong detection capabilities under these adversarial conditions, effectively identifying malicious activities in real time. Notably, the network traffic classifier achieved a **91.82% detection rate** on attack traffic resembling CIC-IDS-2017 samples, while the Web Intrusion Detection System (WIDS) successfully flagged **89% of anomalous HTTP requests** with minimal false positives.

Although deploying the system in a full-scale production environment was beyond the scope of this study, the testbed-based validation provides substantial evidence of its practical applicability. The modular design and containerized architecture enable seamless integration into operational security environments, ensuring adaptability to evolving threat landscapes.

VI. DISCUSSION

The results obtained from the testing phase serve as crucial indicators of the overall effectiveness of the implemented system. Notably, the Random Forest algorithm used in both applications demonstrates impressive accuracy in both multi-class and binary classification tasks. Moreover, the accuracy achieved by the Keras deep learning model surpasses that of the Random Forest algorithm, reaching an impressive 99%. However, due to its high utilisation of compute resources during the prediction of new data points, the deep learning model is relegated to a reserve role in the dual model approach. Incorporating static analysis automation into the system streamlines processes and saves valuable time. The static analysis report, which includes the assembly code of the binary file, provides a comprehensive basis for further investigation. In summary, the malware analysis system, employing a dual-model approach, effectively integrates deep learning models and reduces false positives.

The network traffic classifier demonstrates significant potential in improving network security by accurately analysing real-time traffic and detecting a wide range of cyber threats. Using Random Forest as the main model provided strong results, especially with the CIC-IDS 2017 dataset, where it achieved 99.82% accuracy. This is important as CIC-IDS 2017 includes modern attack types such as DDoS and brute force, making the model highly relevant for current security challenges. The NSL-KDD and UNSW-NB15 datasets, with their focus on both traditional and next-generation threats, further validated the versatility of the classifier.

A key advantage of this system is its modular, containerised design, which allows each component, from traffic capture to classification, to operate independently. This flexibility ensures scalability and efficient deployment in diverse network environments. The use of containers simplified the workflow, making it easier to handle large volumes of traffic while maintaining high detection accuracy. Simulated attack testing using various penetration tools confirmed the effectiveness of the model, successfully identifying attacks such as DDoS and infiltration. However, like any machine learning-based system, its performance will depend on continuous updates and retraining as new threats emerge. In general, this research highlights the potential of AI-driven network traffic classification to provide real-time defence against evolving cyber threats.

The Web Intrusion Detection System (WIDS) effectively uses Isolation Forest for real-time anomaly detection in HTTP logs. Unique features were engineered based on HTTP log files, further augmenting the system's capabilities and contributing to its uniqueness. By optimising hyperparameters such as contamination levels, the system strikes a balance between detecting attacks and reducing false positives. The chosen contamination level of 0.01 showed the best results, offering sensitivity to threats without overwhelming false alarms. Real-time testing in a cloud environment proved the system's effectiveness in identifying attacks, such as a simulated directory traversal. The lightweight agent and cloud setup ensured efficient data collection and processing without straining the servers. Although the system performs well, defining 'normal' behaviour remains a challenge. Continuous monitoring and updates are essential to keep the model relevant in detecting new threats. This research demonstrates how AI-powered systems can enhance web security in dynamic environments.

To overcome bottlenecks and scale the system across large and complex cloud environments, the deployment techniques outlined offer a robust solution. For the Network Traffic Classifier, horizontal scaling with a load balancer is key. As traffic increases, additional virtual machine instances can be dynamically added to the pool, distributing the workload between multiple instances. For example, in a scenario where a large enterprise's network sees a sudden spike in traffic, the load balancer can ensure that no single VM is overwhelmed, preventing latency or data loss. This not only ensures efficient processing but also maintains real-time attack detection capabilities across distributed systems. In the case of Web Intrusion Detection, vertical scaling is more suitable due to the increasing complexity of web server logs. By allocating a dedicated VM to each web instance, resource allocation can be fine-tuned to meet the specific demands of each instance. For example, high-traffic websites can benefit from virtual machines with larger CPU and memory allocations, ensuring that the anomaly detection system continues to

TABLE 10. Summary of our AI-enabled system contribution to cyber incident response.

Security Problem	Application	Algorithm	Dataset	Contribution
Attack Detection	Network Traffic Classifier	Random Forest	**NSL-KDD**, **UNSW-NB15**, **CIC-IDS-2017**	**End-to-end Solution:** An end-to-end solution for network traffic classification encompassing feature extraction from PCAP and the classification model itself. **Automation:** Time and labour-efficient Automation of feature extraction from PCAP files by implementing a feature engineering container. **Real-time Analysis:** Real-time analysis allows an immediate detection and response to network threats.
Anomaly Detection	Web Intrusion Detection	Isolation Forest	Private Dataset	**Distributed Data Collection:** The deployment of lightweight agents on individual web servers for efficient log collection and forwarding to the central Web Intrusion Detection System (WIDS). This distributed data collection method minimises the impact on individual web servers while facilitating real-time analysis. Dynamic **Contamination Ratio Determination:** Methodology to optimise the model's sensitivity to anomalies by dynamically determining the contamination ratio for anomaly detection models by simulating attacks and observing the model's detection rate. **Threshold-Based Alarm Triggering:** Threshold-based approach to trigger alarms only when the number of detected anomalies exceeds the preset value, and this minimises unnecessary alerts (false positives) by prioritizing significant deviations from normal behaviour. **Feature Engineering from HTTP Logs:** Novel approach to feature engineering tailored to HTTP server logs, resulting in the creation of five features that capture relevant aspects of web server activity and enable the detection of breaches with high accuracy and efficiency.
Malware Detection	Malware Analysis System	Hybrid (RF & Keras)	VirusTotal	**Combined Model Architecture:** Hybride architecture consisting of a random forest model and a secondary Keras TensorFlow models mitigating false positives by leveraging the strengths of both models. **Utilization of LSTM Network for Deep Learning:** Long Short-Term Memory (LSTM) network, a type of Recurrent Neural Network (RNN) suitable for analysing sequential data like text, allows the model to effectively capture temporal dependencies in the data, enhancing its ability to detect subtle patterns indicative of malicious behaviour. **Real-Time Analysis with User Interface:** user-friendly web interface that allows users to upload executable files for real-time malware analysis. The process is streamed including file features analysis, maliciousness prediction, and detailed analysis report generation.

TABLE 11. System environment details.

Category	Specification
CPU	Intel Xeon Silver 4210 @ 2.20GHz (10 cores)
GPU	NVIDIA Tesla T4 (16GB VRAM)
RAM	64GB DDR4
Storage	1TB NVMe SSD
Operating System	Ubuntu 20.04 LTS
Cloud Resources	AWS EC2 'g4dn.xlarge' instance (4 vCPUs, 16GB RAM)
Containerization	Docker 24.0.5, Kubernetes 1.28
Deep Learning Framework	TensorFlow 2.10 / PyTorch 1.13
Other Libraries	Scikit-learn 1.2, NumPy 1.23, Pandas 1.5

function efficiently without delays or resource contention. For the Malware Analysis application, employing a load balancer to distribute file uploads across multiple containers can address the challenge of simultaneously handling large volumes of files. Additionally, dedicating a separate node for API handling isolates the file classification process from the user-facing API, improving system responsiveness and scalability. This separation of concerns allows the system to handle thousands of concurrent file uploads or API requests without performance degradation, ensuring that even in complex environments with heavy workloads, the system remains efficient and responsive. This modular architecture of containerised applications effectively addresses scalability and administration challenges, showcasing the versatility of container technology in deploying AI/ML applications. The deployment of various AI/ML applications in containers with different architectures underscores the flexibility and suitability of this deployment method.

While initial results are promising, continuous model updates and rigorous testing are imperative to ensure effectiveness against evolving cyber threats. Moreover, the scalability of the designed system allows the integration of new AI/ML models, expanding its capabilities in various cybersecurity areas. Deploying all models within a designated subnet enhances security and streamlines administration, with different cloud providers offering diverse solutions for AI/ML deployment. A summary of the proposed contribution of the AI-enabled system to the response to cyber incidents is given in Table 10.

VII. LIMITATIONS AND FUTURE WORK

The system discussed in this paper presents certain limitations that highlight areas for future improvement. One significant limitation is that while the system has been tested against a range of known attack types, it has yet to be rigorously evaluated under more comprehensive and

TABLE 12. Ablation study comparing individual and hybrid model performance.

Model Variant	Accuracy (%)	False Positive Rate (%)	Detection Time (ms)	Inference Time (ms)
Random Forest (RF only)	92.1	8.2	2.1	1.8
Deep Learning (DL only)	99.0	7.4	12.5	10.9
Hybrid (RF + DL)	99.0	5.6	5.8	4.3

TABLE 13. Ablation study on contamination ratio selection.

Contamination Ratio	Recall (%)	False Positive Rate (%)
0.001 (Strict)	82.5	2.1
0.01 (Balanced)	91.3	5.2
0.1 (Lenient)	97.5	21.4

diverse real-world threat scenarios. This includes handling the uncertainties associated with dynamic and evolving cyber threats, which may behave unpredictably compared to controlled environments. Additionally, when it comes to anomaly detection in web applications, distinguishing between what constitutes normal behaviour and what qualifies as an anomaly becomes increasingly complex as the volume of data grows. The detection of outliers, especially when processing large datasets, remains a challenge. Another area of concern is the occurrence of false positives, where the system incorrectly flags benign activities as malicious, which can lead to unnecessary alarms and impact the system's overall accuracy.

A qualitative comparison with commercial tools Splunk and CrowdStrike in real-world scenarios revealed that our system outperforms these two widely adopted industry tools. While these systems excel in established rule-based threat detection, our AI-enabled system introduces advanced real-time detection and scalability through containerized deployment. A rigorous, post-implementation evaluation against these tools, focusing on metrics such as accuracy, computational efficiency, and scalability, will be conducted as part of future work to validate our system's practical advantages.

Looking forward, there are several avenues for future work to address these limitations. First, a more comprehensive set of attack scenarios needs to be incorporated into testing to enhance the system's robustness against real-world cyber threats. Continuous training of models using honeypot data will be crucial to keeping the system adaptable to new and emerging attack patterns. To support this, the exploration of TPU-powered VM instances will enable more efficient training of deep learning models within the DFIR environment, overcoming current computational constraints. Moreover, automating the deployment process will enable seamless system deployment across any cloud environment, whether AWS, Azure, or GCP, enhancing the system's flexibility and responsiveness. The use of this system design with multiple virtual private networks (VPNs) and Docker containers provides full control over the design while also supporting a multi-cloud approach, which is a direction for future research.

In addition to the outlined future work, another promising direction is the exploration of how models can be tailored to meet specific organisational needs. For instance, organisations could benefit from models specifically trained for phishing detection, which could analyse incoming emails in real-time to identify potential phishing attempts based on known patterns and emerging tactics. Similarly, models could be developed to detect insider threats by analysing user behaviour across systems. This flexible and model-driven approach would allow organisations to select and customise AI/ML solutions to address their unique cybersecurity challenges.

Another distinctive feature to investigate is the ability to update models using honeypot data collected from the T-Pot platform deployed within the honeypot environment, as illustrated previously in Figure 3. Within this environment, a storage bucket is established to facilitate the transmission of essential log data from the honeypot. A virtual machine instance, located in the research and development subnet of the DFIR environment, gathers data and performs data processing and feature engineering. The processed data are subsequently labelled and uploaded to the training data bucket. Leveraging this training data, machine learning models are trained within the cloud environment using TPU-powered VMs and tested, which results in a new, up-to-date version of the model that is then pushed to the system. The type of training data extracted is tailored to the functionality of each model. This paper presents the concept of the honeypot subsystem, which warrants further investigation in future studies. By continuously exposing the system to potential attack vectors through multiple honeypot daemons and consistently collecting data to train the machine learning models, this approach ensures that the system remains at the forefront of threat detection and incident response. Moreover, it enhances the system's adaptability to the ever-evolving cybersecurity landscape.

Finally, expanding the use of deep learning across all applications within the system will further strengthen its ability to handle complex and diverse cybersecurity challenges. Since neural networks have shown tremendous results, as indicated in malware detection applications, it is highly suitable to introduce neural networks for the network traffic classifier and the web intrusion anomaly detector and also to adapt a dual model approach to reduce false positives. For a visualised perspective highlighting our system contribution, we elaborated Figure 9, which presents a comparative spider radar map in terms of novelty, completeness and innovation. The proposed system offers a comprehensive solution that surpasses existing approaches and systems in several key dimensions.

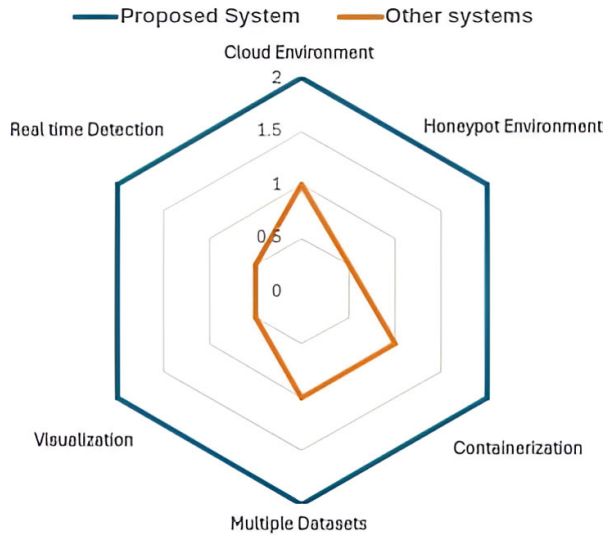

FIGURE 9. Performances comparison of the proposed system [48] with related works.

VIII. CONCLUSION

In this paper, we presented a comprehensive exploration of the practical use of Artificial Intelligence (AI) techniques in the context of cyber security, focusing on their integration into incident response systems within cloud environments. Through the development and deployment of a cyber threat defence system, including a network traffic classifier application, malware analysis application, and web intrusion detection system, we demonstrated the practical application of AI/ML in enhancing cyber security capabilities. Our research highlighted the potential of AI/ML to address emerging cyber threats and improve the efficiency and effectiveness of cyber incident investigations. By deploying the system on both Google Cloud and Microsoft Azure platforms, we showcased the scalability and versatility of AI-powered cyber security solutions in cloud environments. Through testing on multiple datasets such as NSL-KDD, CIC-IDS 2017, UNSW-NB15, and VirusTotal samples, Random Forest models consistently achieved high accuracy in detecting a range of cyber threats, while deep learning models offered additional precision despite their higher computational cost. The modular and containerised architecture of the system ensures efficient deployment and scalability across cloud environments, allowing real-time traffic analysis and threat detection.

Furthermore, the integration of a T-pot for continuous development of models and the ELK Stack for log gathering and visualisation, emphasising the importance of comprehensive data analysis and visualisation in cyber security operations. These tools, combined with AI/ML techniques, offer a holistic approach to cyber security that enables proactive threat detection and rapid incident response. In general, our findings underscore the critical role of AI/ML in modern cyber security and highlight the need for continued research and development in this field. As cyber threats continue to evolve, the use of advanced technologies such as AI and machine learning will be essential to stay ahead of adversaries and ensure the security of digital assets and infrastructure.

FOOTNOTES

ETHICAL APPROVAL

This research was deemed not requiring ethical approval.

AVAILABILITY OF DATA AND MATERIALS

The code generated for this investigation is detailed in [48] and made publicly available on GitHub **https://github.com/Ashfaaq98/Periscope-AI**.

COMPETING INTERESTS

The authors declare that they have no known competing interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGMENT

The APC and Open Access fees for this work are funded by the University of Liverpool.

REFERENCES

[1] (Apr. 19, 2024). *GOV.U.K., Official Statistics Security Breaches Survey*. Accessed: Mar. 4, 2024. [Online]. Available: https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2024/cyber-security-breaches-survey-2024

[2] IBM. (2024). *Cost of a Data Breach Report*. Accessed: Oct. 17, 2024. [Online]. Available: https://www.ibm.com/reports/data-breach

[3] J. N. Angelis, R. S. Murthy, T. Beaulieu, and J. C. Miller, "Better angry than afraid: The case of post data breach emotions on customer engagement," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 2593–2605, 2022, doi: 10.1109/TEM.2022.3189599.

[4] D. Cotroneo, A. Paudice, and A. Pecchia, "Empirical analysis and validation of security alerts filtering techniques," *IEEE Trans. Depend. Secure Comput.*, vol. 16, no. 5, pp. 856–870, Sep. 2019, doi: 10.1109/TDSC.2017.2714164.

[5] S. Rizvi, M. Scanlon, J. Mcgibney, and J. Sheppard, "Application of artificial intelligence to network forensics: Survey, challenges and future directions," *IEEE Access*, vol. 10, pp. 110362–110384, 2022, doi: 10.1109/ACCESS.2022.3214506.

[6] H. Xu et al., "Smart mobility in the cloud: Enabling real-time situational awareness and cyber-physical control through a digital twin for traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3145–3156, Mar. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10018121

[7] B. Żurkowski and K. Zieliński, "Root cause analysis for cloud-native applications," *IEEE Trans. Cloud Comput.*, vol. 12, no. 1, pp. 232–250, Jan. 2024, doi: 10.1109/tcc.2024.3358823.

[8] A. Pichan, M. Lazarescu, and S. T. Soh, "Cloud forensics: Technical challenges, solutions and comparative analysis," *Digit. Invest.*, vol. 13, pp. 38–57, Jun. 2015, doi: 10.1016/j.diin.2015.03.002.

[9] S. Simou, C. Kalloniatis, S. Gritzalis, and H. Mouratidis, "A survey on cloud forensics challenges and solutions," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 6285–6314, Dec. 2016, doi: 10.1002/sec.1688.

[10] (1999). *NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB*. [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html

[11] A. Boukhamla and J. C. Gaviro, "CICIDS2017 dataset: Performance improvements and validation as a robust intrusion detection system testbed," *Int. J. Inf. Comput. Secur.*, vol. 16, no. 1, p. 20, 2021, doi: 10.1504/ijics.2021.117392.

[12] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Canberra, ACT, Australia, Nov. 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[13] K. Cao, S. Hu, Y. Shi, A. W. Colombo, S. Karnouskos, and X. Li, "A survey on edge and edge-cloud computing assisted cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7806–7819, Nov. 2021, doi: 10.1109/TII.2021.3073066.

[14] P. Purnaye and V. Kulkarni, "A comprehensive study of cloud forensics," *Arch. Comput. Methods Eng.*, vol. 29, no. 1, pp. 33–46, Jan. 2022, doi: 10.1007/s11831-021-09575-w.

[15] I. F. De Arroyabe, C. F. A. Arranz, M. F. Arroyabe, and J. C. F. de Arroyabe, "Cybersecurity capabilities and cyber-attacks as drivers of investment in cybersecurity systems: A U.K. survey for 2018 and 2019," *Comput. Secur.*, vol. 124, Jan. 2023, Art. no. 102954, doi: 10.1016/j.cose.2022.102954.

[16] D. Dunsin, M. C. Ghanem, K. Ouazzane, and V. Vassilev, "Reinforcement learning for an efficient and effective malware investigation during cyber incident response," *High-Confidence Comput.*, vol. 2025, Jan. 2025, Art. no. 100299, doi: 10.1016/j.hcc.2025.100299.

[17] A. T. Syed, M. C. Ghanem, E. Benkhelifa, and F. I. Abro, "SPECTRE: A hybrid system for an adaptative and optimised cyber threats detection, response and investigation in volatile memory," 2025, *arXiv:2501.03898*.

[18] C. Stelly and V. Roussev, "SCARF: A container-based approach to cloud-scale digital forensic processing," *Digit. Invest.*, vol. 22, pp. S39–S47, Aug. 2017, doi: 10.1016/j.diin.2017.06.008.

[19] C. Stelly and V. Roussev, "Language-based integration of digital forensics & incident response," in *Proc. 14th Int. Conf. Availability, Rel. Secur.*, Jun. 2019, pp. 1–6, doi: 10.1145/3339252.3339278.

[20] G. Duan, H. Lv, H. Wang, G. Feng, and X. Li, "Practical cyber attack detection with continuous temporal graph in dynamic network system," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 4851–4864, 2024, doi: 10.1109/tifs.2024.3385321.

[21] Q. Zhang, C. Zhou, N. Xiong, Y. Qin, X. Li, and S. Huang, "Multimodel-based incident prediction and risk assessment in dynamic cybersecurity protection for industrial control systems," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 46, no. 10, pp. 1429–1444, Oct. 2016, doi: 10.1109/TSMC.2015.2503399.

[22] A. Nisioti, G. Loukas, A. Laszka, and E. Panaousis, "Data-driven decision support for optimizing cyber forensic investigations," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 2397–2412, 2021, doi: doi:10.1109/TIFS.2021.3054966.

[23] E. E.-D. Hemdan and D. H. Manjaiah, "An efficient digital forensic model for cybercrimes investigation in cloud computing," *Multimedia Tools Appl.*, vol. 80, no. 9, pp. 14255–14282, Apr. 2021, doi: 10.1007/s11042-020-10358-x.

[24] J. Dykstra and A. T. Sherman, "Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform," *Digit. Invest.*, vol. 10, pp. S87–S95, Aug. 2013, doi: 10.1016/j.diin.2013.06.010.

[25] A. Zewdie and T. Girma, "IoT security and the role of AI/ML to combat emerging cyber threats in cloud computing environment," *Issues Inf. Syst., Issues Inf. Syst.*, vol. 21, no. 4, 2020, doi: 10.48009/4_iis_2020_253-263.

[26] D. Dunsin, M. C. Ghanem, K. Ouazzane, and V. Vassilev, "A comprehensive analysis of the role of artificial intelligence and machine learning in modern digital forensics and incident response," *Forensic Sci. Int., Digit. Invest.*, vol. 48, Mar. 2024, Art. no. 301675, doi: 10.1016/j.fsidi.2023.301675.

[27] C. Lin, D. He, X. Huang, M. K. Khan, and K.-K.-R. Choo, "DCAP: A secure and efficient decentralized conditional anonymous payment system based on blockchain," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2440–2452, 2020, doi: 10.1109/TIFS.2020.2969565. [Online]. Available: http://doi

[28] A. M. Qadir and A. Varol, "The role of machine learning in digital forensics," in *Proc. 8th Int. Symp. Digit. Forensics Secur. (ISDFS)*, 2020, pp. 1–5, doi: 10.1109/ISDFS49300.2020.9116298.

[29] K. Hilmand, H. Sarmad, and M. Bakht, "A survey of machine learning applications in digital forensics," *Trends Comput. Sci. Inf. Technol.*, vol. 2021, pp. 20–24, Apr. 2021, doi: 10.17352/tcsit.000034.

[30] S. Nanda and R. A. Hansen, "Forensics as a service: Three-tier architecture for cloud based forensic analysis," in *Proc. 15th Int. Symp. Parallel Distrib. Comput. (ISPDC)*, Jun. 2016, pp. 178–183, doi: 10.1109/ISPDC.2016.31.

[31] D. Dunsin, M. C. Ghanem, and K. Ouazzane, "The use of artificial intelligence in digital forensics and incident response in a constrained environment," *Int. J. Inf. Commun. Eng.*, vol. 16, no. 8, pp. 280–285, 2022. [Online]. Available: https://repository.londonmet.ac.uk/7708/

[32] P. H. Rughani, "Artificial intelligence based digital forensics framework," *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 8, pp. 10–14, Oct. 2017, doi: 10.26483/ijarcs.v8i8.4571.

[33] M. E. Alex and R. Kishore, "Forensics framework for cloud computing," *Comput. Electr. Eng.*, vol. 60, pp. 193–205, May 2017, doi: 10.1016/j.compeleceng.2017.02.006.

[34] Ö. Aslan, M. Ozkan-Okay, and D. Gupta, "Intelligent behavior-based malware detection system on cloud computing environment," *IEEE Access*, vol. 9, pp. 83252–83271, 2021, doi: 10.1109/ACCESS.2021.3087316.

[35] Deutsche Telekom Security GmbH (2024). *T-Pot 24.04.0 (Version 24.04.0) [Computer Software]*. [Online]. Available: https://github.com/telekom-security/tpotce

[36] A. Bhardwaj and K. Kaushik, "Predictive analytics-based cybersecurity framework for cloud infrastructure," *Int. J. Cloud Appl. Comput.*, vol. 12, no. 1, pp. 1–20, 2022, doi: 10.4018/IJCAC.297106.

[37] X. Du et al., "SoK: Exploring the state of the art and the future potential of artificial intelligence in digital forensic investigation," in *Proc. 15th Int. Conf. Availability, Rel. Secur.*, New York, NY, USA, Aug. 2020, pp. 1–10, doi: 10.1145/3407023.3407068.

[38] A. Patrascu and V.-V. Patriciu, "Logging framework for cloud computing forensic environments," in *Proc. 10th Int. Conf. Commun. (COMM)*, May 2014, pp. 1–4, doi: 10.1109/ICCOMM.2014.6866662.

[39] Y. A. Balushi, H. Shaker, and B. Kumar, "The use of machine learning in digital forensics: Review paper," in *Proc. 1st Int. Conf. Innov. Inf. Technol. Bus.*, May 2023, pp. 96–113, doi: 10.2991/978-94-6463-110-4-9.

[40] H. Sedjelmaci, S. M. Senouci, and N. Ansari, "A hierarchical detection and response system to enhance security against lethal cyber-attacks in UAV networks," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 48, no. 9, pp. 1594–1606, Sep. 2017, doi: 10.1109/TSMC.2017.2681698x.

[41] P. Kendrick, A. Hussain, and N. Criado, "Multi-agent systems for dynamic forensic investigation," in *Intelligent Computing Theories and Application*, D. S. Huang, V. Bevilacqua, and P. Premaratne, Eds., Cham, Switzerland: Springer, Jan. 2016, pp. 796–807, doi: 10.1007/978-3-319-42291-6_79.

[42] I. Baptista, S. Shiaeles, and N. Kolokotronis, "A novel malware detection system based on machine learning and binary visualization," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2019, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8757060

[43] M. Ghanem, F. Dawoud, H. Gamal, E. Soliman, T. El-Batt, and H. Sharara, "FLoBC: A decentralized blockchain-based federated learning framework," in *Proc. 4th Int. Conf. Blockchain Comput. Appl. (BCCA)*, San Antonio, TX, USA, Sep. 2022, pp. 85–92, doi: 10.1109/BCCA55292.2022.9922258.

[44] M. R. Watson, N.-U.-H. Shirazi, A. K. Marnerides, A. Mauthe, and D. Hutchison, "Malware detection in cloud computing infrastructures," *IEEE Trans. Depend. Secure Comput.*, vol. 13, no. 2, pp. 192–205, Mar. 2016. [Online]. Available: https://10.1109/TDSC.2015.2457918

[45] J. Patman, D. Chemodanov, P. Calyam, K. Palaniappan, C. Sterle, and M. Boccia, "Predictive cyber foraging for visual cloud computing in large-scale IoT systems," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2380–2395, Dec. 2020, doi: 10.1109/TNSM.2020.3010497.

[46] OWASP Found. (2023). *OWASP Top 10–2023*. [Online]. Available: https://owasp.org/Top10/

[47] ENISA. (2022). *Threat Landscape*. [Online]. Available: https://enisa.europa.eu/topics/cyber-threats/threats-and-trends

[48] M. Ashfaaq M. Farzaan, M. Chahine Ghanem, A. El-Hajjar, and D. N. Ratnayake, "AI-enabled system for efficient and effective cyber incident detection and response in cloud environments," 2024, *arXiv:2404.05602*.

[49] L. Ben-Shimol et al., "Observability and incident response in managed serverless environments using ontology-based log monitoring," 2024, *arXiv:2405.07172*.

[50] S. Liu et al., "RCInvestigator: Towards better investigation of anomaly root causes in cloud computing systems," 2024, *arXiv:2405.15571*.

MOHAMMED ASHFAAQ M. FARZAAN received the bachelor's degree in information technology with a specialization in computer systems and network engineering and the M.Sc. degree (Hons.) in cyber security from the University of Westminster, U.K. He is currently an Research and Development Lead and an Engineer working on applied AI with Priam Cyber AI Ltd., U.K. He is a Multi-Certified Cyber Security Expert holding many Certification, such as ISC2 Certified in Cyber Security (CC) and Blue Team Junior Analyst (BTJA) and has collaborated with various professionals from the industry on different projects, including autonomous incident response, LLM-based threat hunting, and malware analysis. His research interests include multiple disciplines, including cybersecurity, digital forensics, generative AI, large language models, security operations center (SOC), and the IoT/cloud.

MOHAMED CHAHINE GHANEM received the Engineering degree in computer sciences, the M.Sc. degree (Hons.) in digital forensics, and the Ph.D. degree in cyber security from the City, University of London, and the PGDip. degree in security studies. He is currently an Associate Professor of cyber security and AI and the Director of the Cyber Security Research Centre, London Metropolitan University, and an Associate Professor of cyber security with the Cyber Security Institute, University of Liverpool. His research interests include applying AI to solve real-world cybersecurity and digital forensics problems and he has published numerous research articles in the world's top cybersecurity journals. He is a Senior Fellow of HEA. He is a Professional Member of British Computer Society achieved many certificates, such as GCFE, CISSP, ACE, XRY, and CPCI, with over 19 years of experience at the senior management level in the field of cybersecurity, digital forensics and incident investigation in law-enforcement and world-leader corporates. He delivered several keynote lectures at different prestigious venues. He is an Academic Editor of *ACM Digital Threats: Research and Practice* and a Senior Advisor at Kroll LLC.

AYMAN EL-HAJJAR is currently a Senior Lecturer and the Head of the Cyber Security Research Group, School of Computer Science and Engineering, University of Westminster. He also leads the Cyber Security and Forensics B.Sc. Program. While leading the university's Cyber Security Research Group, the group has successfully secured substantial funding. This funding supports their extensive research in the IoT security, data integrity, incident response, and cryptographic solutions. He has published numerous research papers and edited books in the field of cybersecurity. His publications cover many topics, including the IoT security, data integrity, cryptography, and smart city technologies. His research interests include the security of the Internet of Things (IoT) and smart cities. Within these domains, he and his Ph.D. students are exploring the use of various cryptographic models to secure communication of the IoT devices both in transit and at rest, specifically focusing on data confidentiality and integrity and enabling secure key distribution among devices. He is also interested in addressing the human factor in cybersecurity. He actively seeks solutions that lessen the burden on end-users, advocating for a paradigm shift toward more accessible and user-friendly security practices. He is also a member of U.K. Cyber Security Council Ethical Committee, ensuring the ethical practices of all the council's members are maintained to the highest standard. He is also a reviewer of top cyber security and cryptography journals.

DEEPTHI N. RATNAYAKE received the Ph.D. degree in probe request attack detection in wireless LANs using intelligent techniques. She is currently a Principal Lecturer of cyber security with the University of Hertfordshire, U.K., an active contributor to the Cybersecurity and Computing Systems Research Group, and the Program Leader of the Modular Master's Program in Computer Science. With over 30 years of experience spanning industry, defense, and academia, she brings significant expertise in cybersecurity, networking, and information systems management. Her previous role as the Head of a defense network has provided her with both technical depth and strategic insight that she applies to her current work. Her research interests include intrusion detection and prevention through intelligent techniques, security in cloud computing, software defined networks, and information security management and compliance. She is an Associate Editor of the *Information Security Journal: A Global Perspective* and serves as a Cyber Security Columnist and is an active, longstanding member of the Information Security Specialist Group of British Computer Society (BCS-ISSG). A passionate advocate for addressing real-world cybersecurity issues, she has presented extensively at academic and professional venues, authored high-impact journal articles, and contributed to both national and international conferences. Her work continues to advance state-of-the-art cybersecurity practices and solutions.