Computer Systems Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.040721

ARTICLE

Security Monitoring and Management for the Network Services in the Orchestration of SDN-NFV Environment Using Machine Learning Techniques

Nasser Alshammari[1], Shumaila Shahzadi[2], Saad Awadh Alanazi[1,*], Shahid Naseem[3], Muhammad Anwar[3], Madallah Alruwaili[4], Muhammad Rizwan Abid[5], Omar Alruwaili[4], Ahmed Alsayat[1] and Fahad Ahmad[6,7]

[1]Department of Computer Science, College of Computer and Information Sciences, Jouf University, Sakaka, Al Jouf, 72341, Saudi Arabia

[2]Department of Computer Sciences, Kinnaird College for Women, Lahore, Punjab, 54700, Pakistan

[3]Department of Information Sciences, Division of Sciences and Technology, University of Education, Lahore, 54770, Pakistan

[4]Department of Computer Engineering and Networks, College of Computer and Information Sciences, Jouf University, Sakaka, 72341, Saudi Arabia

[5]Computer Science, Florida Polytechnic University, Lakeland, Florida, 33805, USA

[6]Delta3T, Lahore, Punjab, 54700, Pakistan

[7]Department of Basic Sciences, Common First Year, Jouf University, Sakaka, 72341, Saudi Arabia

*Corresponding Author: Saad Awadh Alanazi. Email: sanazi@ju.edu.sa

Received: 29 March 2023 Accepted: 22 May 2023 Published: 19 March 2024

ABSTRACT

Software Defined Network (SDN) and Network Function Virtualization (NFV) technology promote several benefits to network operators, including reduced maintenance costs, increased network operational performance, simplified network lifecycle, and policies management. Network vulnerabilities try to modify services provided by Network Function Virtualization MANagement and Orchestration (NFV MANO), and malicious attacks in different scenarios disrupt the NFV Orchestrator (NFVO) and Virtualized Infrastructure Manager (VIM) lifecycle management related to network services or individual Virtualized Network Function (VNF). This paper proposes an anomaly detection mechanism that monitors threats in NFV MANO and manages promptly and adaptively to implement and handle security functions in order to enhance the quality of experience for end users. An anomaly detector investigates these identified risks and provides secure network services. It enables virtual network security functions and identifies anomalies in Kubernetes (a cloud-based platform). For training and testing purpose of the proposed approach, an intrusion-containing dataset is used that hold multiple malicious activities like a Smurf, Neptune, Teardrop, Pod, Land, IPsweep, etc., categorized as Probing (Prob), Denial of Service (DoS), User to Root (U2R), and Remote to User (R2L) attacks. An anomaly detector is anticipated with the capabilities of a Machine Learning (ML) technique, making use of supervised learning techniques like Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), Naïve Bayes (NB), and Extreme Gradient Boosting (XGBoost). The proposed framework has been evaluated by deploying the identified ML algorithm on a Jupyter notebook in Kubeflow to simulate Kubernetes for validation purposes. RF classifier has shown better outcomes (99.90% accuracy) than other classifiers in detecting anomalies/intrusions in the containerized environment.

KEYWORDS

Software defined network; network function virtualization; network function virtualization management and orchestration; virtual infrastructure manager; virtual network function; Kubernetes; Kubectl; artificial intelligence; machine learning

1 Introduction

The network and infrastructure of Network Function Virtualization (NFV) have become more varied and complex, technology upgrades have taken much longer, and the demand for human experts for comprehensive network configuration has increased network maintenance costs [1]. The European Telecommunications Standards Institute (ETSI) NFV working group provides representative systems and information for improvised NFV. NFV refers to the restructuring of the telecom sector in which network operations formerly hosted on dedicated physical equipment are now virtualized and operate on pools of commodity servers.

1.1 Network Attacks

The anomalous attack of the given data set is categorized into four classifications.

i. DoS attacks accommodate authorized requests or prevent users from accessing a network. DoS attacks include Back, Smurf, Neptune, Teardrop, Pod, and Land in the provided dataset.
ii. When the attacker accesses a user account on the server via password or using other resources, then exploits a backdoor to obtain root access to the device, are U2R attacks. In the dataset, U2R attacks include Buffer Overflow, Rootkit, Load Module, and Perl.
iii. R2L attack happens when an unauthorized user can transfer information to a computer on the network to achieve local access as a resident of that network. For instance, Warezclient, Warezmaster, Guess Password, Internet Message Access Protocol (IMAP), FTP Writes, Multi-Hop, Phf, and Spy are R2L attacks in the given dataset.
iv. A probe attack is an operation to collect information or data to enforce security controls. Probe attacks in the provided dataset include Satan, IPsweep, Portsweep, and Nmap

1.2 Software Defined Network

Software Defined Network (SDN) primary function is to manage the system, exercise control over it, and endow it with the capacity to generate basic programs and separate system administration and applications. Using SDN programs is genuinely programmable and enables the progressive arrangement of a comprehensive traffic stream to deal with evolving issues. SDN is a controller that integrates the framework overseeing rationally and decouples the data plane from the control plane [2].

A vital part of the control plane is the controller, which has a comprehensive network view and can facilitate the flexible and adaptable organization of streams. OpenFlow is the standard for communication between data and control planes [3]. The scheme represents a channel between these two domains, allowing switch stream table entries to be added or removed. Moreover, programmable frameworks can eliminate impediments to new and accelerating advancements in computer networks. SDN was created by experts who had problems upgrading or modifying the software on network hardware equipment when they attempted to do something different. As shown in Fig. 1, SDN can be separated into three primary planes, the application, control, and data planes, to create the SDN architecture's utility.

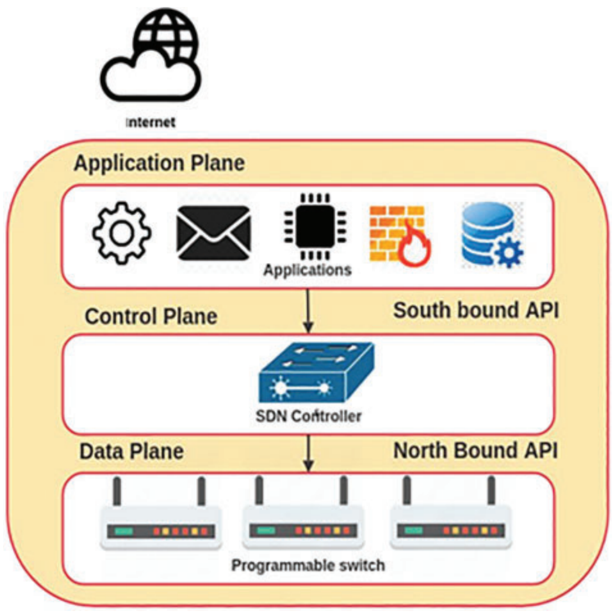


Figure 1: Software defined network architecture

SDN has been created to isolate the network control plane from the forwarding plane, which can reduce operating expenses and the time required for implementing new services compared to traditional networks. Despite its benefits, this technology has inherent risks and vulnerabilities. SDN architecture involves the development of robust real-time Intrusion Detection Systems (IDS) for identifying illicit behavior [4].

1.3 Network Function Virtualization

NFV is a network environment that runs virtual machines (VMs). Since SDN provides network control plane flexibility and innovation capabilities, the virtualization of network functions supported by NFV provides network operators flexibility in service delivery. Customers' demands can be individually supplied and dynamically changed by combining Virtual Network Function (VNF), composing SFC [5,6]. NFV consists of an Operational Support System/Business Support System (OSS/BSS), VNF, Network Function Virtualization Infrastructure (NFVI), and Network Function Virtualization MANagement and Orchestration (NFV MANO) modules. NFVI comprises all the hardware and software resources deploying VNFs [7]. The hardware resources consist of computation, storage, and network elements, a virtualization layer that effectively provides VNFs with processing, storage, and networking capabilities are presented in Fig. 2. NFV MANO is responsible for network policies, configuration, modification, service orchestration, and resource orchestration. It is further divided into three modules which are discussed below. Resource and service orchestration increases the workload of the environment, which also enhances the risk of multiple intrusions in the NFV MANO environment.

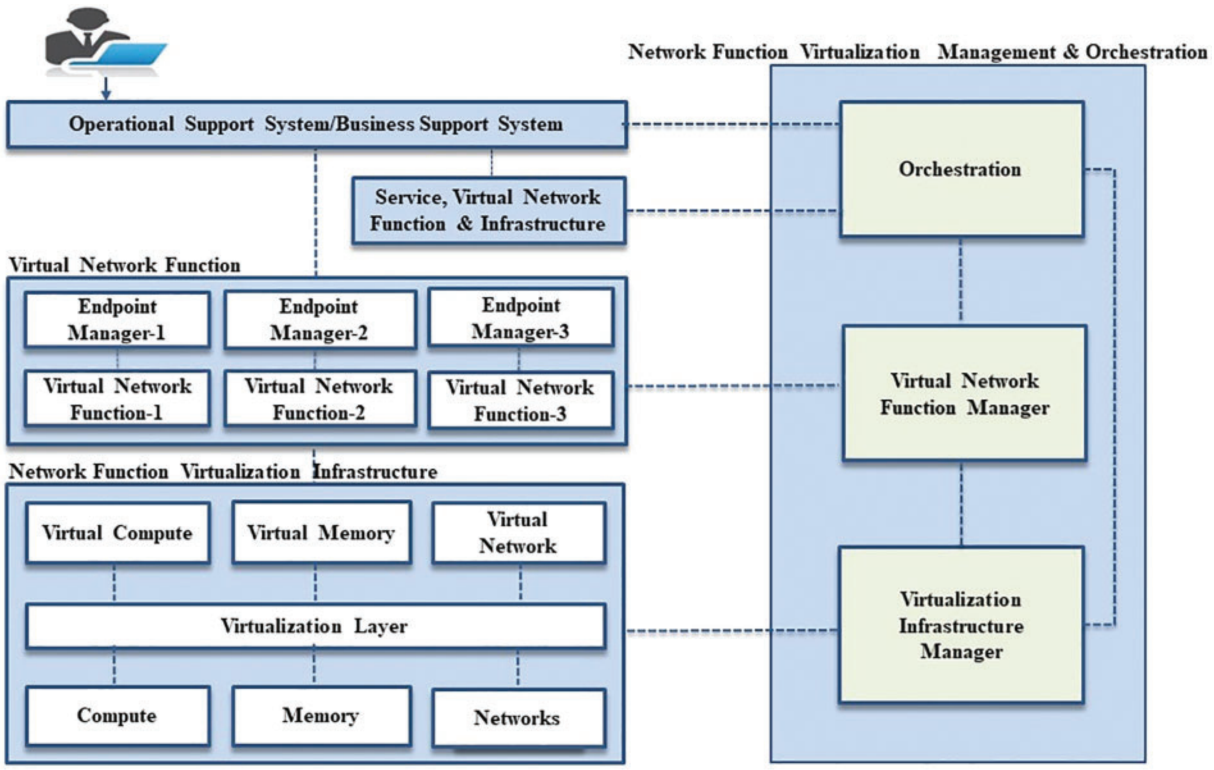

Network Function Virtualization Management & Orchestration
Operational Support System/Business Support System
Service, Virtual Network Function & Infrastructure
Orchestration
Virtual Network Function
Endpoint Manager-1
Endpoint Manager-2
Endpoint Manager-3
Virtual Network Function-1
Virtual Network Function-2
Virtual Network Function-3
Virtual Network Function Manager
Network Function Virtualization Infrastructure
Virtual Compute
Virtual Memory
Virtual Network
Virtualization Layer
Virtualization Infrastructure Manager
Compute
Memory
Networks

Figure 2: Network function virtualization

NFV and SDN have appeared as successful approaches in recent years. Specifically, NFV and SDN are two closely related technologies that operate on various layers of the NFV architectural system using the network abstraction paradigm. NFV aims to virtualize all network capabilities from the hardware it resides in, enabling the network to expand without adding more equipment. SDN makes the configuration, programming, and operation of networks simpler. In other words, SDN is highly complementary to NFV, which provides complete network control capability and handles traffic steering on the data plane as part of Service Function Chaining (SFC) development. SDN makes it easy to construct new networking abstractions, simplifying network management and enabling the network's transformation [8].

1.3.1 Network Function Virtualization Orchestration

It is responsible for the docking and lifecycle of new network networks (e.g., instantiation, scaling in and out, performance measurement, and termination). The NFV orchestrator also handles global development control and permission for resource requests in the NFV.

1.3.2 Virtual Network Functions Manager

This layer controls the lifecycle of VNFs management from instantiating, upgrading, scaling, and terminating to performing other functions. It also conducts coordination and case monitoring with other elements of the NFVI. This layer includes network functionality implementations, e.g., firewall, load balancing, and Intrusion Detection System (IDS).

1.3.3 Virtual Infrastructure Manager

The Virtualized Infrastructure Manager (VIM) functionality involves monitoring and maintaining VNFs' interaction with NFVI. It effectively conducts resource management, which entails managing and allocating VNFs of NFVI services such as computation, storage, and network resources.

1.4 Kubernetes

Kubernetes is a container orchestration framework that automates applications' configuration, scale, and management. It is part of the VIM and the Virtual Network Function Manager (VNFM) in ETSI's NFV architecture context. However, it extends beyond the application life cycle management [9]. Kubernetes is an open-source framework that provides a platform for NFV [10]. It contains an enhancement called the Operator Module, which illustrates some application-specific systems integration framework represented in Fig. 3 [11].

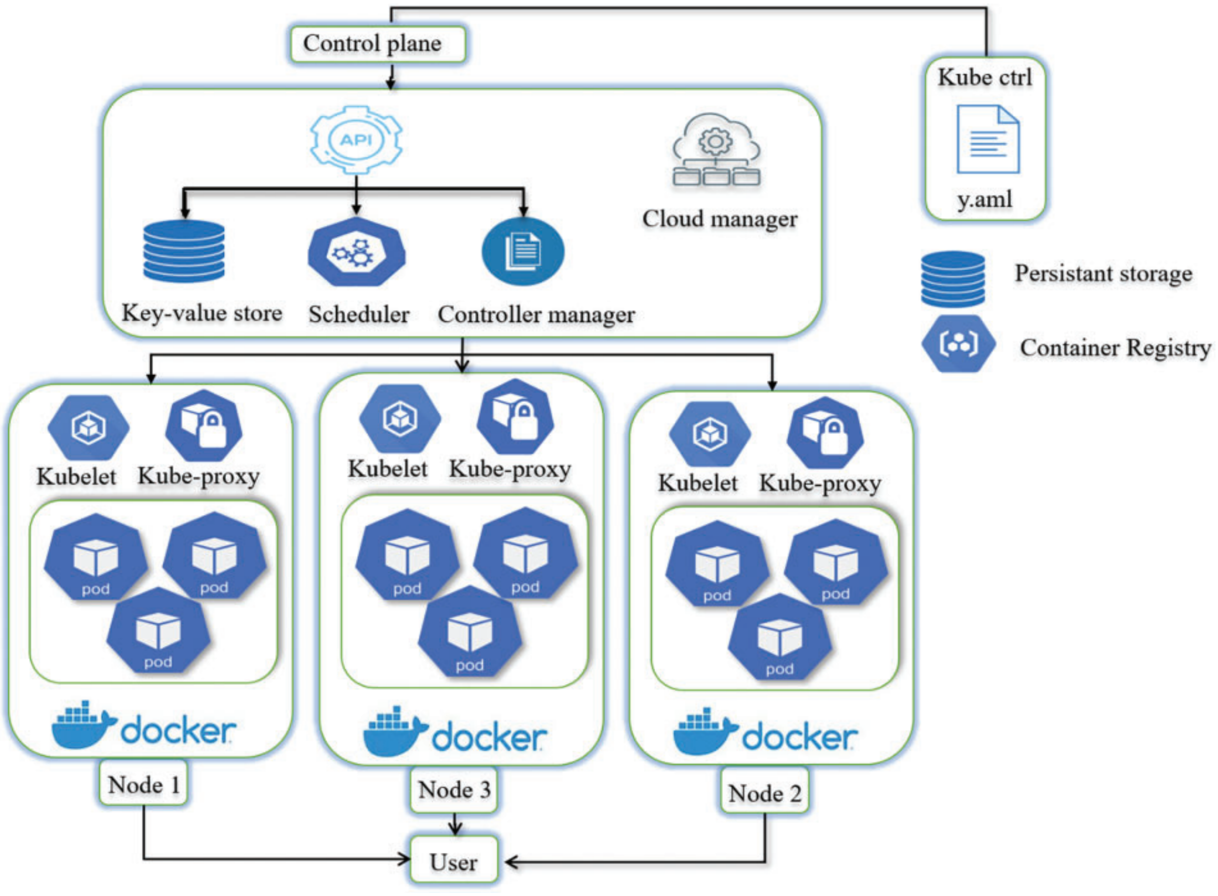


Figure 3: Kubernetes architecture

There are several components of Kubernetes. Kubernetes multiple devices meet the management and orchestration standards to achieve efficiency restrictions and specifications. In particular, system containers can assist in packaging all system libraries, directories, and programs needed to serve the target environment. Containers help programmers to execute and handle their products and services effectively.

Containers help programmers to execute and handle their products and services effectively. There are several components of Kubernetes.

1.4.1 Control Plane

The control plane regulates the worker nodes and pods in Kubernetes. It is a group of processes that manage Kubernetes nodes. It is divided into five sections: the Kube-API-Server, the Key-Value-Store, the Kube-Scheduler, the Kube-Controller-Manager, and the Cloud-Controller-Manager. The network node these modules operate is called the 'Master Node.' These components can operate on a single node or multiple nodes in development. Still, they are suggested to run on multiple nodes to provide high efficiency and load balancing.

1.4.2 Kubernetes Worker Node

The computers or physical servers that run required applications are known as worker nodes in a cluster. The Kubernetes master manages each node, and numerous nodes are related to the master node. Multiple pods are running on the node, and multiple containers are running between pods.

1.4.3 Kubectl

The Kubectl is a command-line interface and mechanism for connecting with the Application Programming Interface (API) service and sending commands to a master node.

1.4.4 Persistent Storage

Kubernetes handles the containers running an application and manages the application's data connected to a cluster [12]. Kubernetes helps users order computing capacity without understanding the underlying storage system. Persistent amounts cannot survive a pod but are unique to a cluster.

1.4.5 Container Registry

A container repository holds the container images on which Kubernetes is based, setting up a registry or a third party.

1.5 Anomaly Detection

Massive research has been carried out regarding the privacy and security threats posed by the use of communication technology in various domains, including the household, health, utilities, business operations and information, and supply chain, as a result of emerging technologies and digitalization. While people rely significantly on digital devices for their day-to-day activities and businesses, cybersecurity garners widespread interest [13,14]. Anomalies are beyond-of-the-ordinary data patterns that deviate from the norm. Anomaly detection is the identification of data deviations or unpredictability. Doing so at the beginning can save precious resources and time during data processing and decision-making after identifying anomalous data [15].

1.6 Machine Learning Approach

Machine Learning (ML) algorithm deployed in the network environment is Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), Naïve Bayes (NB), and Extreme Gradient Boosting (XGBoost) Classifier, prioritizing the performance of the network lifecycle. These ML algorithms output based on a feature selection classifier for input space, which has been used as a classifier [16–20]. Using univariate and correlation, the idea is to classify the prioritized content against the least prioritized content. A tool known as an IDS is capable of detecting potential cyber-attacks. The anomaly detection mechanism controls the network life cycle to see malicious activity.

There are two kinds of network anomaly in the environment, i.e., signature-based detection systems and anomaly-based detection systems. Signature-based intrusion detection systems track possible threats by looking for suspicious patterns, such as network byte sequences or malware-recognized malicious instruction sequences [21,22]. Although signature-based intrusion detection systems can detect existing attacks quickly, they cannot detect new attacks with no history. A system based on abnormalities is designed to detect unusual attacks. Because of their excellence in detecting unexplained threats, anomaly-based detection systems are still widely used [23–26].

1.7 Machine Learning Classifiers

In this section, the different ML algorithm domains are implemented for detection purposes. Anomaly detection is a popular application of ML techniques. NB, LR, SVM, RF, and XGBoost are ML algorithms used for training and implementation in the Kubernetes module to secure the NFV MANO environment [27]. LR and Linear SVM algorithms typically work for multiclass classification data, while RF and XGBoost work for separable spatial data. However, there is no basic rule for choosing an algorithm, and we can test them all to see which one fits better for the containerized environment.

1.7.1 Logistic Regression

LG is used to predict the categorical dependent variable from several independent variables. The net input function contains all the selected features used for classifier training and testing the dataset. Then the sigmoid function is calculated using these input features. The result of training and the predicted data feature is in discrete value. LR gives the outcomes probabilistic values, which lie between 0 and 1.

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

$$h\theta(x) = \frac{L}{1+e^{-k(x-x_{0)}}} \tag{2}$$

In the unit step module, the threshold value defines the probability of either 0 or 1. Values above the threshold value tend to be 1, while values below the threshold value tend to be 0. 0.50 is the threshold value. P represents the probabilities which means that if probability > 0.50 value is rounded off to 1, and if probability ≤ 0.50 value is rounded off to 1.

$$Threshold = f(p) = \begin{cases} 1, & 0.50 < p \\ 0, & 0.50 \geq p \end{cases} \tag{3}$$

If the value leads to 1, showing standard cases probabilities are more than abnormalities also, if it leads to below 0.5, the number of anomalous cases is significant, and the features dataset needs to be improved.

1.7.2 Naïve Bayes

The NB algorithm solves classification problems based on the Bayes theorem. It is mainly used in text classification with an extensive training dataset. It assumes that the frequency of each feature is independent of the frequency of other features [28]. To render classifications, x is used to estimate y to evaluate the data function. The ML classifier represents many attributes and classes by C_1, C_2, and C_K in the ML classifier. The probability that an object with an x_1, x_2, x_n characteristic vector belongs to a class.

$$P(y) = P(C_i|x_1, x_2, \ldots, x_n) = \frac{P(x_1, x_2, \ldots, x_n|C_i) .P(C_i)}{P(x_1, x_2, \ldots, x_n)} \, for\ 1 \leq i \leq k \tag{4}$$

P(y) is also known as the likelihood class, and P (xi|y) is the probability condition. After all, odds are measured, and the chance of anomalous and usual cases is now decided. The training data and characteristics match the NB classifier and calculate the final results, verified using cross-validation and performance metrics results.

1.7.3 Support Vector Machine

SVM is used for the identification of irregularities. It provides a certain proportion of predefined outliers to identify anomalies. They lie between the origin and the optimum hyperplane separation. SVM performance values are ordinary behavior (higher values are significant), and negative values are abnormal behavior (with lower values representing more significant abnormality). Sclera uses the Linear SVC to define the linear classifier as the point set x satisfactory in the SVM model. When the selected feature data is discarded, w is needed to identify new data. If the value of y_i = 1 means most cases are malicious, and if y_i = −1 means most cases are normal.

$$L = Optimization\ target - Summation\ of\ constraint\ with\ 0\ multiplier \tag{5}$$

$$L = \frac{1}{2}\,||w||^2 - \sum \alpha_i \left[y_i \left(w^T x + b\right) - 1\right] \tag{6}$$

SVM is trained with the parameter regularization L = 1.0, which is suited to handle high-dimensional data due to its high generalization and learning capabilities. The SVM fits the trained data with the predicted set and gives the outcomes calculated by the performance matrices.

1.7.4 Random Forest

RF is a nonlinear regression and classification methodology integrating many decision-making units, training each on bit different assumptions, and considering a characteristics sample. The RF has other parameters, including n estimators, max depth and min samples split, min samples leaf, and so on [29]. The best estimator is chosen for training and checking data sets using a Sklearn Random Search CV library. The important bootstrap sampling function based on a feature to draw a Decision Tree (DT) is calculated as. RF are expected finally by averaging tree simulations.

$$fi_i = \frac{\sum j : node\ j\ splits\ on\ features\ i\ ni_j}{\sum k \in all\ nodes\ ni_k} \tag{7}$$

where n_{ij} is the importance of j-node, w is a weighted number of samples that reach j-node, fi is the importance of feature j-node, c-node j is impurity, left is the child node from left split on node j, and right means child node from right split on node j. These can then be normalized to a value between 0 and 1 by dividing by the sum of all feature importance values.

$$normfi_i = \frac{fi_i}{\sum j \in all\ nodes\ fi_j} \tag{8}$$

1.7.5 XGBoost

The XGBoost includes a linear model solver as well as tree learning algorithms. It focuses on the model's performance and speed [30]. XGBoost feature selection classifier, XGBoost fit the training and testing data with the best estimator and selected a couple of features. The loss function trains the model to predict the best estimator. Root Mean Square Error (RMSE) is calculated for regression to predict the training of anomaly cases—binary grouping using log loss. For multi-classification, m log loss is used.

$$\text{Rooted Mean Squared Error for regression}: \mathrm{L} = \frac{1}{N}\sum\nolimits_{i=1}^{n}\left(y_i - y^{\wedge}{}_i\right)^2 \tag{9}$$

$$\text{LogLoss for binary classification}: \mathrm{L} = -\frac{1}{N}\sum\nolimits_{i=1}^{n}\left(y_i \log\left(p_i\right) + \left(1 - y_i\right)\log\left(1 - p_i\right)\right) \tag{10}$$

To comprehend other criteria, one must first grasp the underlying model. L is the label scale, l is the attribute of the chosen dataset, y_i is the actual attribute of the dataset, $y^{\wedge}{}_i$ is the label scale function, and p_i is the projected attribute of the dataset. XGBoost adds all of the trees built from the dataset's predictions and optimizes the effect.

1.8 Problem Statement

To meet the SDN-NFV system requirements, we must eliminate network security configuration errors that can lead to vulnerabilities and impact overall reliability, network stability, and maintenance costs. The existing frameworks related to NFV MANO lack the security that can provide holistic security management. Therefore, this research focuses on the security of NFV MANO's orchestration to improve integrity in the NFV environment. By improving network functions' security properties, NFV could offer security benefits with proper infrastructure locking.

1.9 Objectives

The research aims to supply the approaches that can strike the limitations exposed within NFV MANO network service environments' security. During this research, most characteristics are:

- Anomaly detection mechanisms secure information regarding network service, lifecycle, and performance in NFV MANO environments.
- To identify malicious activities and keep the availability, confidentiality, and integrity of NFV MANO environments.
- To enhance the end-user experience and Quality of Service (QoS) during the resource and service orchestration.

1.10 Contribution

The study's primary contribution is to spot anomalies immediately as they occur and to provide recommendations for each data point to mitigate the impact of any identified anomalies. The emphasis on real-time detection of orchestration inconsistencies is a novel aspect of this method. By detecting anomalies as soon as they occur, the system can minimize their effect on service efficiency or eliminate component failures from occurring. Regardless of the presence of anomalies, the provision of distinctive recommendations for each registered data point is another notable aspect of the approach. This level of granularity enables operators to address issues promptly and efficiently, thereby reducing the overall impact of system performance anomalies. The proposed method generally appears well-suited for monitoring complex systems and ensuring that anomalies are promptly detected and addressed.

The RF method aims to develop a robust and effective ML algorithm for identifying and classifying anomalies. The method is founded on constructing multiple DTs using a random subset of features and data samples, then averaging or voting on their predictions. RF addresses some of the shortcomings of DTs, which are susceptible to overfitting and instability when trained with complex data sets. RF can reduce overfitting and improve the model's generalization performance by constructing an ensemble of trees, each trained on a subset of the data. In addition, by arbitrarily selecting a subset of features for each tree, RF can reduce tree-to-tree correlation and increase their diversity. This can result in improved overall performance, as the ensemble can encapsulate various characteristics of the underlying data distribution. The RF technique is motivated by the wish to develop a flexible and robust ML algorithm capable of handling complex datasets and achieving high accuracy and generalization performance.

2 Literature Review

In this chapter, a literature study is carried out to highlight different researchers' attempts to optimize edge caching to improve efficiency and QoS in NFVO. Other recent works relating specifically to their application in various fields are also studied here, which have proven to be a great source of guidance in coming up with the proposed concept.

Multi-Layer Intrusion Detection and Prevention (ML-IDP) is an innovative strategy developed by Artificial Intelligence (AI) based security systems to identify intrusions in the SDN-NFV-enabled cloud of 5G networks. Using AI, the proposed method defends against security attacks [31]. The proposed ML-IDP method is evaluated with NS3.26 for various security threats, including Distributed Denial of Service (DDoS). The experiment results demonstrate that the ML-IDP efficiently recognizes and prevents attacks [32]. AI has a broader range of applications in the security management regime, which directs the investigation of cognitive abilities resembling humans to achieve more consistent and effective defense capabilities [33].

The Mouseworld, an SDN-NFV based safe analyzer, presented a novel exploratory system incorporating SDN and NFV to establish an ecosystem capable of mixing, transmitting, storing, and designating this traffic as usable for training and validation [34]. The use of ML algorithms to identify cybersecurity threats [35]. The results demonstrated security threat detection to validate the system's viability and practicability [36].

SDN raises particular security concerns, particularly when its controller is vulnerable to DDoS attacks. If DDoS attacks arise regarding an SDN server, the server's operation and contact capability will be overwhelmed. DDoS attacks in SDN can be detected using an ML-based model and feature selection techniques. In order to streamline these models, feature selection techniques with relatively shorter training times were chosen [37].

Access Control Management and Orchestration represent the security framework of Security Planning, Security Enforcement, and Secure Monitoring for NFV. Security Enforcement and Monitoring work together, mitigating the risks and modifying the Security Planning layer for better updates and deployments [1]. Open Baton toolkit implements a current NFV MANO and develops the server-client application where a virtual firewall handles all the controls and management. They had great results using the developed application, which can embed into a virtual infrastructure.

In this research, researchers measured traffic in an NFV environment that causes threats like Distributed Denial of Services and other cyber-attacks [5]. The attacks are identified by ML orchestration of VNF functions using a security orchestrator in the NFV. Using ML, the VNF-affected components are separated to track, collect, and process the traffic for further implementation. The limitation of testing is a potential network graph of VNF implementation in the cloud, but an information and control network may offer more scalability benefits through distributed analytics and data management [38,39].

This study proposes the Open Flow Discovery Protocol (OFDP) to capture network topology status to avoid injection and flood attacks. Spearman's ranking correlation is used to evaluate the network traffic between connections and calculate the time of each Link Layer Discovery Protocol (LLDP) frame to decide if there is topology discovery by a man-in-the-middle attack [40]. The verification details and timestamps are added to LLDP packets returned to the SDN controller. The results show that the proposed mechanism has a broader scope for detecting the topology through man-in-the-middle attacks than the Keyed-Hash Message Authentication Code (KHMAC) and SALL detection mechanisms [41].

Joint path allocations for energy efficiency, QoS-aware, and VNF service placement chaining enable traffic flow across the virtual network functions chain. This model solves VNF positioning, assignment, and flow routing optimization problems. Heuristic algorithms are proposed for the various optimization issues to find near-optimal solutions in reasonable times. Besides, the issue of utility feature chaining is resolved to minimize overhead power usage and network reconfiguration. The simulation results revealed that the schemes suggested distributing the network resource to close energy use to the perfect solution [42].

Most extant AI algorithms improve network performance by increasing network depth or breadth. Nonetheless, the large size of the network model presents two issues: a large-sized model will occupy an excessive amount of storage space, which is not favorable for deployment in practical applications, and a large-sized model will impose a significant computational burden. It is inappropriate for applications with limited computing capacity or stringent real-time requirements. When devising an AI-based model, the trade-off between model size and efficacy must be carefully considered [43]. Natural knowledge improves the comprehension of intelligent models, i.e., ML and Deep Learning (DL) models; consequently, there is an increasing tendency to leverage off-site knowledge bases to enhance data-driven models [44]. In DL-based methods, researchers employed Deep Neural Networks (DNN) to learn semantic information and then used the network's prediction outcomes to fill in absent parametric values [45].

Internet of Things (IoT) systems are susceptible to various security hazards, including DoS, network intrusion, and data storage [46,47]. An innovative security architecture based on ML has been put forward, exploiting SDN and NFV to mitigate various threats [48]. The planned intelligent platform includes a monitoring agent and a response agent which distinguishes network traffic patterns using ML models in IoT. The detection rate of anomalies was encouraging [49,50].

Detecting DDoS attacks in SDN by feature selection methods and ML models is used to examine attacks in the network and measure each classifier's performance rate [51]. The detector used multiple classification models like SVM, NB, KNN, and Artificial Neuron Networks (ANN). The accuracy performance of the KNN classifier is 98.3%, which is significantly higher than the accuracy rate of other classifiers in DDoS attack detection. The increased success rate of detecting DDoS attacks in SDN improves processing time [52].

Massive data center abnormality identification in data centers generates vast data, makes the machine operate in challenging big data environments, and can produce dynamic reactions in real-time. An anomaly detection data processing engine is proposed. Analytic systems are used that include ML algorithms that can interpret inconsistencies in the measurements and log streams captured in real-time from operating data centers. The analysis system used by Docker Innovations allows the computing machine to be completely modular and ensure maximum availability in massive big data environments.

ML-driven scaling and placement of VNF at the network edge measured network traffic at VNF instances. It represented a neural network model that auto-scaled itself by analyzing processed network traffic's function in the amount of VNF instances. The traffic tracks model achieves 97% prediction in the commercial mobile network [53,54].

In this study, NFV is used by several Post Office Protocols (POPs) and heterogeneous infrastructures. Equal and fair surveillance is vital to NFV's market success. Still, reliable and accurate monitoring over many instances of VNF is a challenge to diagnose any failures or deviations to cause corrective measures immediately. Two methods based on statistics for dynamically identifying deviations in NFV networks without predetermined thresholds are analyzed using the open-source NFV monitoring framework. The critical aspect of the process performance is correctly selecting the parameters to be used [55].

NFV-based Multi-Service chaining performs dynamical service feature chaining for foreground traffic to maximize both the end-to-end efficiency of the SFCs and network resources' overall usage [56]. For this purpose, both hybrid SDN/Internet Protocol (IP) networks and upcoming, mature IP environments are ready to use this scheme. The Deep Reinforced Learning (DRL) solution can achieve close to an optimal solution with optimal network efficiency. Until applicable to all other different traffic matrices, DRL has to be very lightweight training only. Based on the network traffic, the results have shown that proposed network success in complex traffic environments can be very easily achieved effectively [57,58].

3 Materials and Methods

In this section, the proposed methodology is outlined in detail. The presented framework is containerized environment where all network services of VIM and VNFM work adequately. The complete scenario is shown below in Fig. 4 and explained in the subsequent sections.

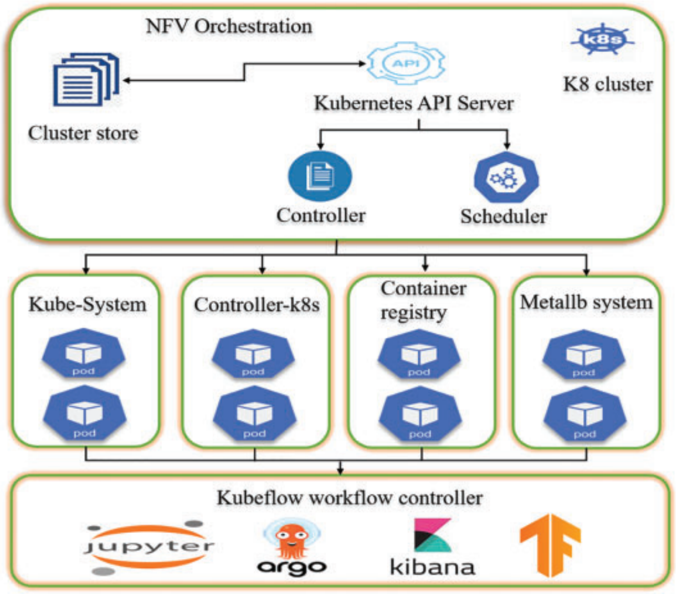


Figure 4: Proposed anomaly detection framework

3.1 Kubernetes Cluster

A Kubernetes cluster is installed in a public cloud using Juju, bootstrapped by micrk8s using the k8scluster node. Attackers with cloud credentials can gain access to the cluster's management layer. Anomaly events cause chaos in the cluster's data and infrastructure. Deployments, installations, storage, and compute capital can be deleted and hacked by anomaly attacks.

3.2 Cluster Store

The cluster store contains all the files, pod, and container data. This database also performs API, tracking, pod scheduling, and pod and Kubectl networking. The microk8s.kubectl create storage command is used to create the store. Any node collapse or process death changes the values in the inventory. The Kubernetes cluster stores data and modify it based on processes. It employs a key-value storage store to store the Kubernetes cluster's configuration, the current, and the desired state.

3.3 Kubernetes Application Programming Interface Server

The Kube-API server command allows connecting to the Kubernetes API server. The API server manages the cluster node setup, updates, and cluster API edition. The cluster node in Kubernetes contains a .yaml file that contains API Server. Attackers that gain access to a private registry use it to plant their vulnerable images, and the recipient can retrieve the latter. Because of the section's anomalous cases, each pod uses a different flag to constantly customize the anomaly cases and improve the end-user network life cycle.

3.4 Pod Controller

In this process, to further illustrate the model, the *microk8s.kubectl get pods-n namespaces* command is used in this module to access the k8s cluster container and pods names. The Kube-system, controller-k8s, container-registry, metallb-system, and Kubeflow container are used in the orchestration environment to configure network services with the labeled namespace and deployment. The kube config file, used by Kubectl, provides information about Kubernetes clusters, such as location and credentials. If attackers have access to this file, they can access the clusters. At that point, ML algorithms are used to detect malicious activity.

3.5 Scheduler

The Kubernetes scheduler is a bulk function that contains a set of policies, hypervisor, and workload-specific features, and it has a significant impact on availability, performance, and flexibility. The scheduler provides service specifications, policy constraints, affiliation, anti-affinity criteria, data translation, inter-workload interference, and deadlines. Workload-specific specifications are made available through the API if required.

3.6 Kubeflow Workflow Controller

It is one of the most critical modules in the assumed scenario. Kubeflow POD Controller is used for ML services such as Webhook-Bootstrap, Webhook-Deployment, Argo, Jupyter, Katib, KFServing-Controller, Kubeflow Pipeline, Meta-Info, Meta-Controller, ML-Pipeline, TensorFlow Job, and others. Jupyter notebook is used to track anomalies in Kubernetes and stable network services. In the Jupyter notebook asset, the dataset is first incorporated, then a function is selected using the correlation selector, and ML algorithms are applied to these selected features to predict the results using the evaluation metric shown in Fig. 5. It demonstrates how to serve a scikit-learn based classifier model on a Kubernetes cluster. The deployment steps are also applicable for models trained with ML. The traffic will be blocked if network traffic is flooded, and dismiss notifications will be sent.

Algorithm: Anomalous traffic identification in SDN-NFV environment

```
1.  Start
2.      Input: Packet reception from the SDN-NFV network
3.          Wdata: Whole data set with chosen features
4.              Ldata: Label data set with assigned categories
5.                  For j = 0 to M
6.                      do
7.                          Ldata: Test-Train-Splitting
8.                          ML_Classifier: Executed
9.                              Checking:
10.                                 If
11.                                     Non anomalous caes
12.                                 Then
13.                                     Ldata: Accepted
14.                                 Else
15.                                     Ldata: Blockaded
16.                                 End
17.                         Return: Performance rate
18.                         Cross_Validation: Checking performance rate
19.                     End
20.                 End
21.         Output: Frequency computed for chosen data features
22. End
```

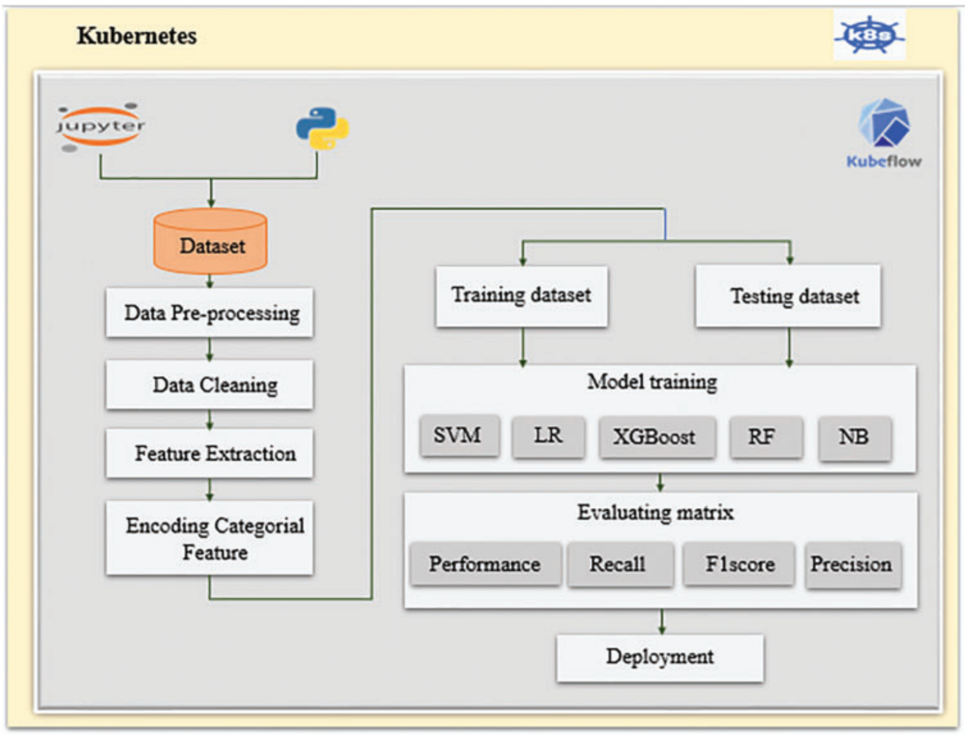


Figure 5: Framework of Kubeflow for anomaly detection

4 Validation and Results

For the identification of malicious activities that disturbed the network services of containerized environment implementation, this research follows these phases.

4.1 Overview of Dataset

The dataset consists of different anomalous cases and develops the intrusion in the network environment. There are 494021 data points and 42 attributes in the dataset. Since the dataset includes null values, they must be replaced with digits [59–62]. There are 37 numerical characteristics and 5 categorical characteristics. The term "function" is used as a class name. Regular activity is labeled 0, while malicious activity is labeled 1.

The network instances are classified into 23 classes containing 22 anomalous and 1 normal category, as seen in Table 1. There are multiple malicious activities like Smurf, Neptune, Teardrop, Pod, Land, IPsweep, Portsweep, Satan, Nmap, Warezclient, Warezmaster, Back, Guess Password, IMAP, FTPWrites, Multi-hop, Phf, Spy, Buffer Overflow, Rootkit, Loadmodule, Perl and their total number of frequencies are calculated in Table 2.

Table 1: Classification of anomalies and their frequencies

Category	Attack type	No. of records
Normal	Normal	97278
DoS	Smurf	280790
	Neptune	107201
	Land	21
	Back	2203
	Teardrop	979
	Pod	264
Probe	IPsweep	1247
	Portsweep	1040
	Nmap	231
	Satan	1589
R2L	Warez client	1020
	Guess password	53
	Spy	2
	Warez master	20
	Imap	12
	Phf	4
	Ftp writes	8
	Multi-hop	7
U2R	Buffer overflow	30
	Perl	3
	Rootkit	10
	Load module	9

Table 2: Records of anomaly attack

Normal/attacks	No. of records
Attack	396743
Normal	97278

4.2 Simulation Platform

Kubernetes is an open-source tool for enabling AI applications and ML network services. The Kubeflow tool is used to allow Kubernetes ML services. Kubeflow offers a Jupyter Notebook environment, which includes a Python module. Despite these points of concern, it deals with various libraries, suggesting that AI implementations might be feasible. Python was selected for this situation due to its wide range of interests. Sklearn (Scikit-learn) is an AI library that works with Python, and Sklearn offers the user a broad scope of choice with its various AI applications [11,63–65].

After loading the dataset, it explained the functionality of the data feature described in Table 3. Pandas is a wonderful Python-based data exploration library. Matplotlib is a library used to create graphs for data analysis, and NumPy is a Python library that allows you to perform scientific and coherent tasks quickly and efficiently.

Table 3: Dataset feature

Feature no.	Feature name	Description	Type
0	DURATION	Connection length.	Int64
1	PROTOCOL_TYPE	Protocol type.	Object
2	SERVICE	Service type, e.g., http, telnet, etc.	Object
3	FLAG	Connection's state at the time.	Object
4	SRC_BYTES	Data bytes are sent by the source IP address.	Int64
5	DST_BYTES	Data bytes are sent by the destination IP address.	Int64
6	LABEL	Attack indication whether it is a normal case., an attack case, or an unknown case.	Object
7	COUNT	The number of contacts with the same source and destination IP addresses as the current connection.	Int64
8	SAME_SRV _RATE	The percentage of connections to the same service.	Float64
9	SERROR_RATE	The percentage of connections that have "SYN" errors.	Float64
10	SRV_SERROR _RATE	Percentage of connections that have "SYN" errors in service feature.	Float64
11	DST_HOST _COUNT	The number of connections whose destination IP address is the same as the current connection	Int64
12	DST_HOST_SRV _COUNT	The number of connections with service style is the same as the current one.	Int64

(Continued)

Table 3 (continued)

Feature no.	Feature name	Description	Type
13	DST_HOST_SAME _SRC_PORT_RATE	The destination host count function shows the percentage of connections whose source port is the same as the current connections.	Float64
14	DST_HOST _SERROR_RATE	The destination host service's count function calculates the percentage of connections with "SYN" errors.	Float64
15	DST_HOST_SRV _SERROR_RATE	The destination host service's count function calculates the percentage of connections with "SYN" errors.	Float64

4.2.1 *Frequency of Selected Dataset Features*

This section represents the frequency of the dataset containing the features to find occurrences rate. Below graphs illustrate the flag data feature, service, and protocol category bar plots. On the y-axis, it displays the count. It also shows the percentage of each segment in the overall results. Fig. 6 depicts the SF Flag, S_0 Flag, and REJ Flag contribution rate higher than the others for detecting an anomaly. The SF Flag appears most frequently in the results. Fig. 7 depicts the ICMP protocol, and TCP protocol results are higher than others. The most popular protocol in data is the ICMP protocol. Fig. 8 illustrates the three most general service categories: ECRI, private, and HTTP.

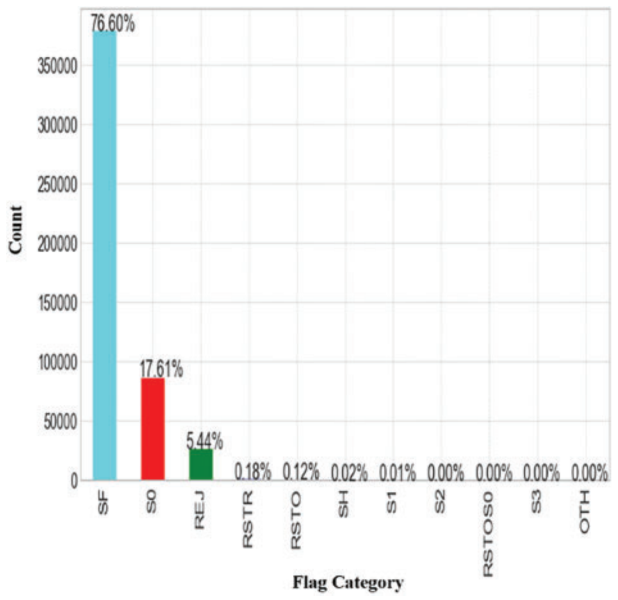


Figure 6: Frequency of flag-type data feature

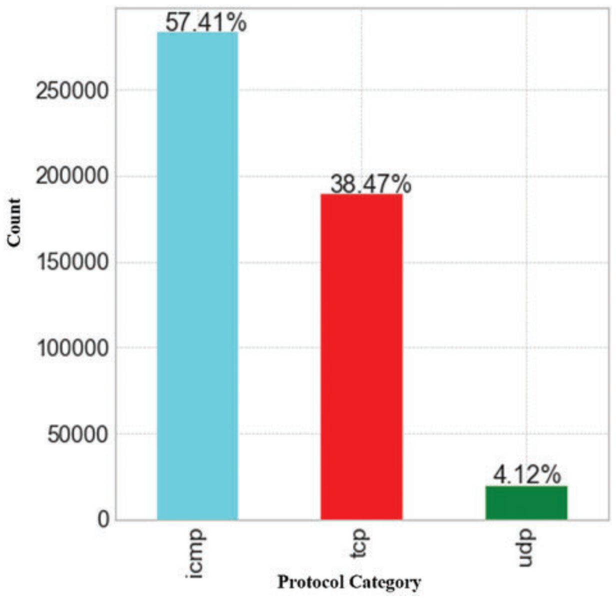


Figure 7: Frequency of protocol-type data feature

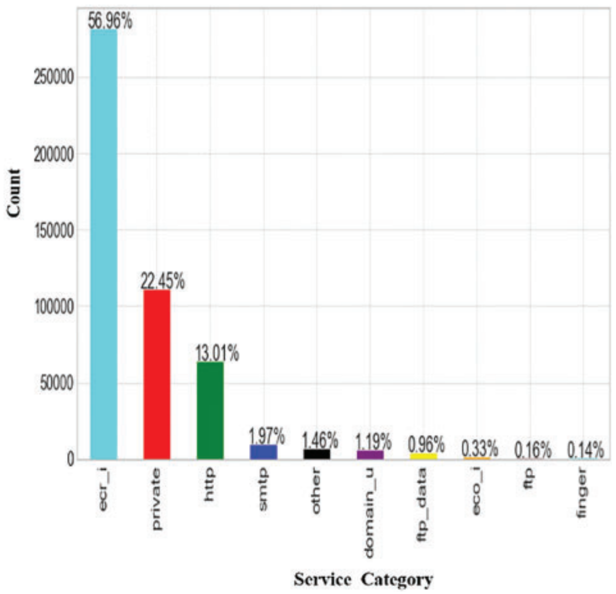


Figure 8: Frequency of service-type data feature

4.2.2 Data Encoding Categorical Features

The data has been translated into train and test data. It is accomplished so that the learning criteria can be extended to test data to train data. The encoding method is used to encrypt such categorical variables. An estimator is trained to build a transformer that transforms the object and float data type into the integer. A transformation specification is an estimator in which data preparation and ML model training transformations are done. The Python Fit command transforms the ML algorithm into a vector matrix that requires the X train and predicted data set as input.

4.2.3 Training and Testing Dataset

During this phase, the data set is divided into training and testing data sets, and multiple models are trained using different classification algorithms such as XGBoost, SVM, NB, RF, and LR classifiers. An ML algorithm's input and output variables must be integer quantities of the same scale. Data conversion entails converting categorical and string data to numerical data simultaneously. Data is split into train and test sets to avoid loss until decoding. Training data is often broken into cross-validation data. 70% of the data is used for training, and 30% for research.

4.2.4 Confusion Matrix of Training Dataset

The confusion matrix is a heat map certainly makes more sense than representing it as an array. The Scikit-learn library function is used for plotting an uncertainty matrix. To classify improperly labeled data points, an uncertainty matrix is plotted. The uncertainty matrix demonstrates the model's ability to forecast or distinguish the groups correctly. In this case, the numbers reflect the sum of TP, FP, TN, and FN. The suggested ML algorithms, such as NB, LR, SVM, RF, and XGBoost, have confusion matrices demonstrated through Figs. 9–13.

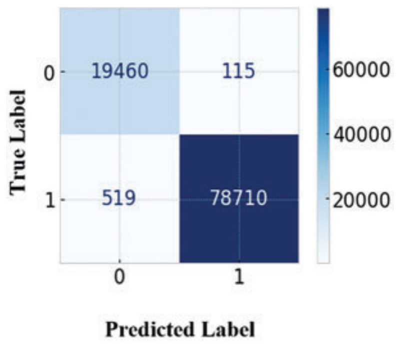


Figure 9: Confusion matrix for Logistic Regression

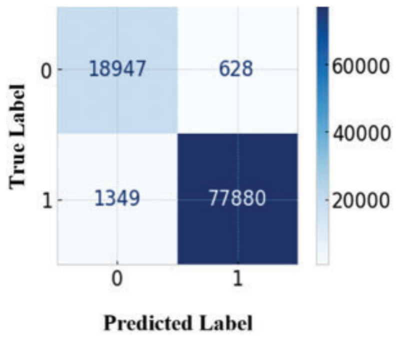


Figure 10: Confusion matrix for Naïve Bayes

Figs. 9 and 10 respectively show that in the training area, the LR model correctly identifies 19460 normal cases while labeling only 115 normal cases as anomalous. 519 anomaly cases were correctly identified in the training sample, while 78710 were misclassified. The NB model accurately detects

18947 normal cases, with only 628 normal cases classified as anomalous. 1349 anomaly cases were correctly identified in the training sample, while 77800 were misclassified.

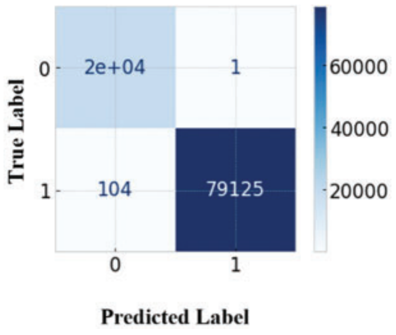


Figure 11: Confusion matrix for Random Forest

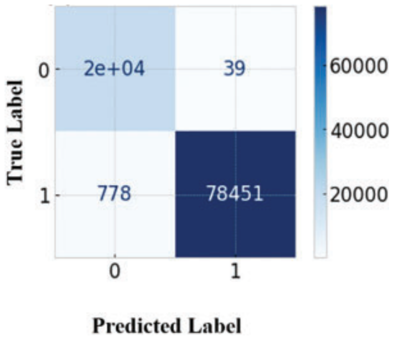


Figure 12: Confusion matrix for Support Vector Machine

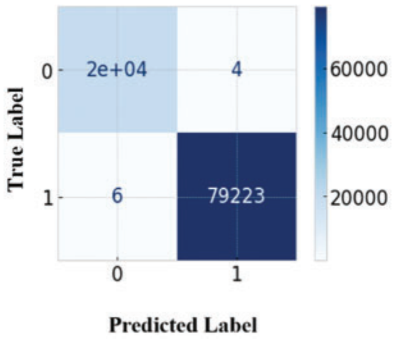


Figure 13: Confusion matrix for eXtreme Gradient Boosting

Figs. 11–13 respectively show that the RF model correctly identifies 20000 normal cases on the training sample, with only 1 normal case labeled as anomalous. 104 anomaly cases were correctly identified in the training sample, while 79125 were misclassified. The SVM model accurately detects 20000 normal cases, with only 39 normal cases labeled as anomalous. 778 anomaly cases were correctly identified in the training package, while 78451 were misclassified. The XGBoost model accurately detects 20000 normal cases, with only 4 normal cases labeled as anomalous. 6 anomaly cases were correctly identified in the training sample, while 79223 were misclassified.

4.2.5 Normalization of Training Dataset

Normalization is the method of scaling floating-point data to values between 0 and 1. Many of the training algorithms necessitate the normalization of input feature data. Figs. 14–18 display the normalized metrics of the possible ML algorithms such as NB, LR, SVM, RF, and XGBoost.

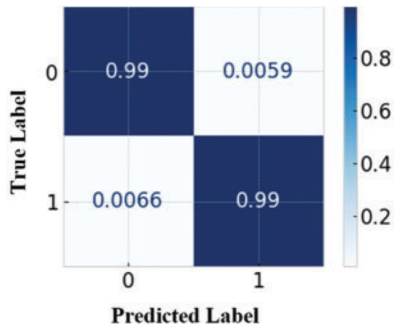


Figure 14: Confusion matrix for Logistic Regression

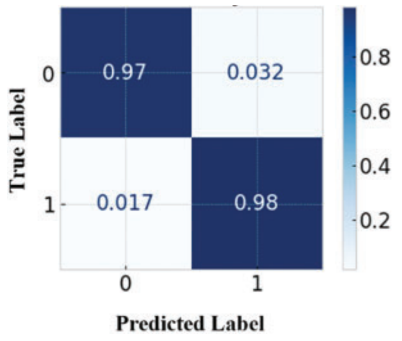


Figure 15: Confusion matrix for Naïve Bayes

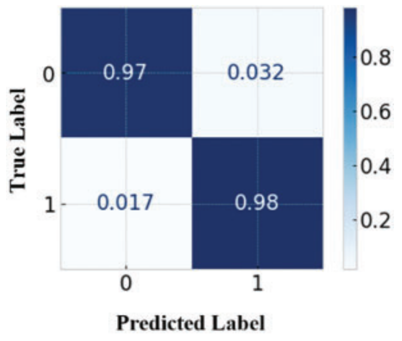


Figure 16: Confusion matrix for Random Forest

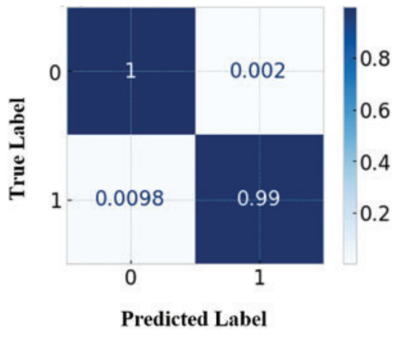


Figure 17: Confusion matrix for Support Vector Machine

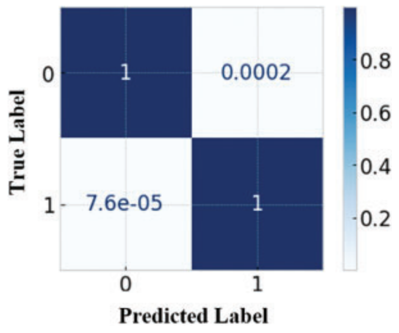


Figure 18: Confusion matrix for eXtreme Gradient Boosting

4.2.6 Correlation Matrix

Correlation is a method to calculate each data feature's dependencies to another data feature, which helps detect which attribute is better for classification. Suppose the dependencies of two data features are the same. In that case, that feature will not contribute to the model, so the data function must be discarded in favor of one strongly correlated variable [66]. There are numerous methods for measuring correlation; the most common is Pearson's correlation coefficient, with maximum values indicating a very high correlation [67]. Some of the correlation method's features are illustrated in Figs. 19 and 20.

4.2.7 Evaluation Metrics

The output of ML algorithms such as RF, LR, SVM, XGBoost, and NB for detecting anomaly material is presented in this section.

- True Positive Rate (TPR) demonstrates the percentage of scenarios where the real truth value is True, and the classifier's prediction is still true.

$$TPR = \frac{TP}{TP + FN} \tag{11}$$

- False Positive Rate (FPR) indicates the number of anomalous predictions generated.

$$FPR = \frac{FP}{TP + FP} \tag{12}$$

- False Negative Rate (FNR) can be viewed as an anomalous assumption predicted as a negative prediction.

$$FNR = \frac{FN}{TP + FN} \tag{13}$$

- True Negative Rate (TNR) refers to anomalous, accurately expected cases that are negative.

$$TNR = \frac{TN}{FP + TN} \tag{14}$$

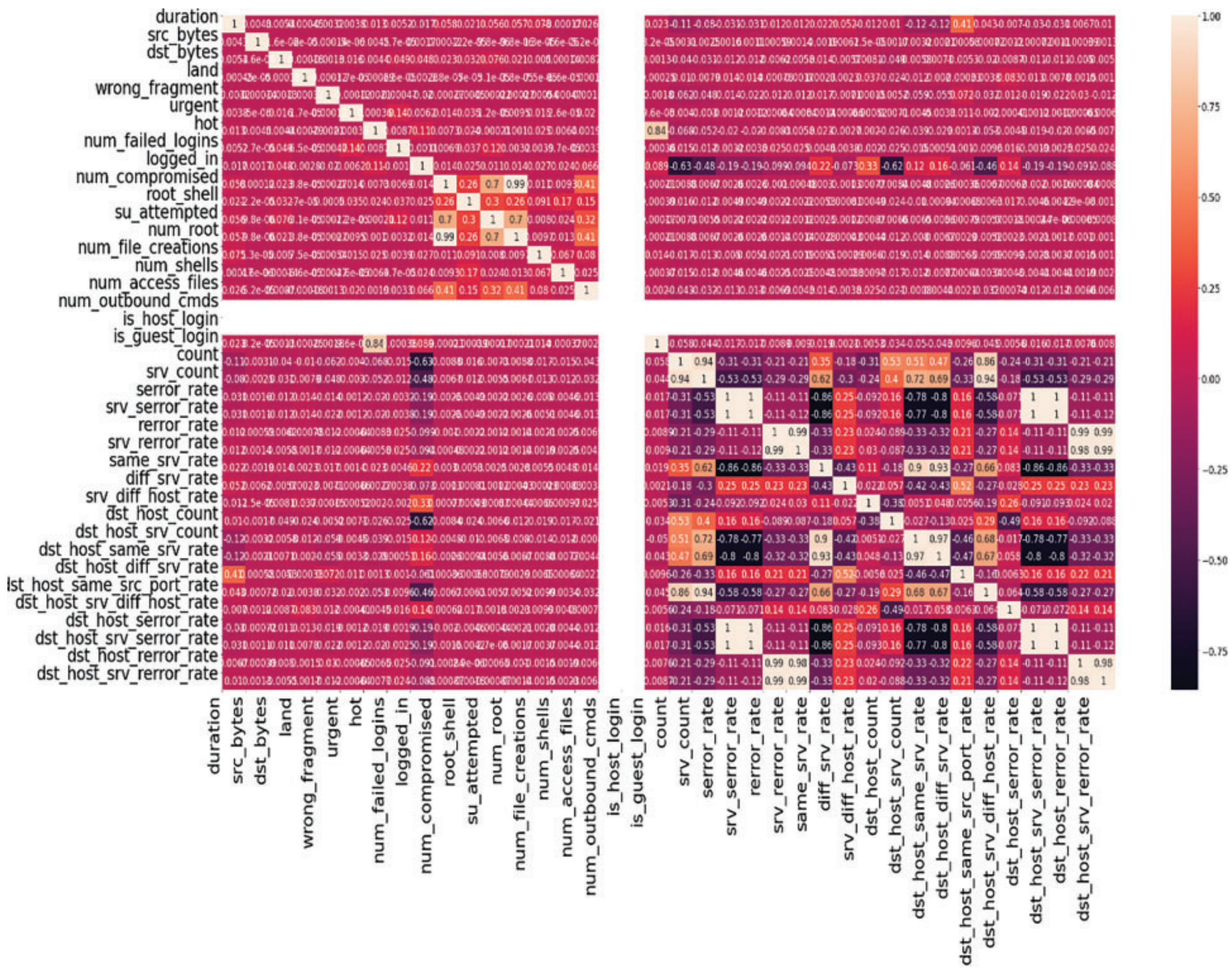
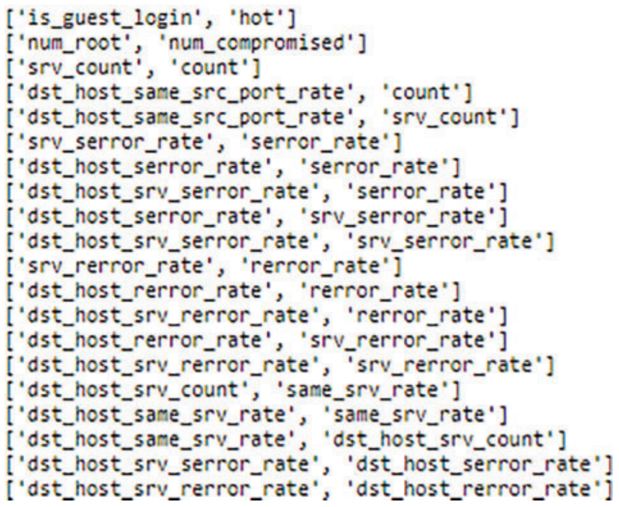


Figure 19: Correlation matrix for feature selection



```
['is_guest_login', 'hot']
['num_root', 'num_compromised']
['srv_count', 'count']
['dst_host_same_src_port_rate', 'count']
['dst_host_same_src_port_rate', 'srv_count']
['srv_serror_rate', 'serror_rate']
['dst_host_serror_rate', 'serror_rate']
['dst_host_srv_serror_rate', 'serror_rate']
['dst_host_serror_rate', 'srv_serror_rate']
['dst_host_srv_serror_rate', 'srv_serror_rate']
['srv_rerror_rate', 'rerror_rate']
['dst_host_rerror_rate', 'rerror_rate']
['dst_host_srv_rerror_rate', 'rerror_rate']
['dst_host_rerror_rate', 'srv_rerror_rate']
['dst_host_srv_rerror_rate', 'srv_rerror_rate']
['dst_host_srv_count', 'same_srv_rate']
['dst_host_same_srv_rate', 'same_srv_rate']
['dst_host_same_srv_rate', 'dst_host_srv_count']
['dst_host_srv_serror_rate', 'dst_host_serror_rate']
['dst_host_srv_rerror_rate', 'dst_host_rerror_rate']
```

Figure 20: Correlation analysis of extracted data feature

This distribution is presented in Table 4 by visualizing the confusion matrix.

Table 4: Performance metric

		Predicted	
		Positive (0)	Negative (1)
True	Positive (0)	TP	FN
	Negative (1)	FP	TN

4.2.8 *Performance Measures of Classifiers*

To evaluate the execution of ML processes, various tests are used. Their suggested solutions have different characteristics and provide multiple results when it comes to detecting anomalies/attacks. In Table 6, a few exhibition metrics, such as accuracy, precision, recall, and F1 score, are measured to distinguish network services as normal or malicious.

Accuracy is defined as the rate of correctly classified anomaly/attack instances in a class, and it can be expressed as:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{15}$$

Precision is the ratio of correct anomaly predictions to the total predicted anomaly, or assumption anomalies made by the classifier are correct from a total of positive anomalies.

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

The recall is the ratio of correctly expected anomaly attacks to overall attack cases. The proportion of correctly expected cases concerning the overall number of cases.

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

The F1 score, the F scale, is calculated as the recall and precision harmonic mean. It is self-evident in a classifier model that if the dataset has a high precision, it demonstrates a low recall value and vice versa.

$$F - measure = \frac{2 * Recall * Precision}{Recall + Precision} \tag{18}$$

The outcomes of this performance can easily be calculated by using a confusion matrix. The accuracy, precision, recall, and F1 score predicted the anomalous cases in the containerized environment that still exist after each classifier's training and predicted better classifier performance.

5 Discussion

TPR and FPR values can be used to compare performance in various optimization settings. Higher TPR values indicate a more elevated positive rate, which means optimized performance, whereas higher FPR values indicate a higher negative rate, which is not optimal for performance within Table 5. RF displays the maximum TPR value, while SVM displays the lowest.

Table 5: Performance rate for TPR and FPR

Algorithms	TPR	FPR
Naïve Bayes	0.9712	0.0288
Logistic Regression	0.9878	0.0122
Support Vector Machine	0.9037	0.0963
Random Forest	0.9990	0.0010
XGBoost	0.9812	0.0188

ML algorithms' performance is calculated in the order of precision, recall, F1 score, and accuracy. The overall score (i.e., weighted average) is sufficient to achieve 0.99 for all measurement parameters. However, the accuracy of predicting each attack style varies greatly. Although the identification rate for unknown attacks is reasonable, as shown by an F1 score of 0.99. The attack and normal groups (1 and 0 labels) produced similar effects. Since weighted average results were high when massive class identification performs well. The above figures show the classifier's overall performance is very high. The accuracy and recall levels are 99%, indicating that the system has reached a functional point. The RF classifier has a precision of 99%, and four classifiers have a recall rate of up to 98%, as represented in Table 6. Since the TP is restricted to a relatively small value and a small number of network lifecycle attacks that do not belong to the attack group are categorized into a normal category, the denominator of precision's formula would quickly increase, achieving a high precision impossible. In other words, the system detects minimum abnormal cases.

Table 6: Performance results

Algorithms	Accuracy	Precision	Recall	F1 score	Time (s)
Naïve Bayes	0.9712	0.9623	0.9845	0.9956	892
Logistic Regression	0.9878	0.9945	0.9912	0.9945	7368
Support Vector Machine	0.9037	0.9867	0.9913	0.9923	356
Random Forest	0.9990	1.0000	1.0000	1.0000	2523
XGBoost	0.9812	1.0000	1.0000	1.0000	1566

In the aforementioned scenario, RF models are appropriate because they have the following advantages over other well-known ML methods: low training time complexity O(nlog(n)) and fast prediction; resilience to deal with unbalanced datasets, an embedded feature selection method, and intrinsic metrics to rank features by importance; and native support for categorical and continuous features. When the RF models are subjected to comparative evaluations in the domain of IDS, these benefits are objectively highlighted. The RF model is a collection of DT that can be utilized for classification or regression. In the classification case, the prediction is based on the majority consensus of the DT-predicted values, whereas in the regression case, the result is the mean of the trees' results. In the training phase, a training set is created for each tree based on the samples in the original training set, and to create each tree split, random features are selected and evaluated to determine which one should produce the split. This randomness produces distinct trees, which, when combined, typically accomplish superior prediction performance.

We presented an approach based on RF: We trained the RF with identified datasets containing known anomalous cases in order to assist in defining the best descriptors for scoring/classification by providing the most pertinent information during the classification phase. This selection significantly decreases computational complexity and time, allowing the computational effort to be focused on the recommended candidates, thereby accelerating research in security monitoring and management for network services orchestrated by SDN-NFV. After the automatic selection of these anomalous cases, RF is used as a classifier to evaluate the selection quality and provide a prediction of anomalous behavior. Accurate feature selection has the potential to enhance system performance, processing speed, and prediction precision.

In this subsection, we evaluate the time complexity of the identified models. Table 6 compares the computational cost of the specified dataset's various approaches. RF's complexity time is not the finest. As the below-given table demonstrates, however, the RF approach obtains the highest prediction accuracy compared to other methods. In addition, the RF method achieved the highest success rate of 99.9%.

Table 7 shows the performance comparison of deployed models with the existing state-of-the-art studies using different datasets and anomaly detection techniques.

Table 7: Performance comparison of the deployed and existing classification models

Comparative Researches	Used dataset	Algorithm	Accuracy
Kasongo et al. [68]	UNSW-NB 15	Random Forest	74.87%
		Extreme Gradient Boosting	71.43%
Jing et al. [69]	UNSW-NB 15	Support Vector Machine	85.99% (binary classification) 75.77% (multiclass classification)
Belgrana et al. [70]	NSL-KDD	Radial Basis Function (RBF)	94.28%,
		Condensed Nearest Neighbors (CNN)	95.54%
Niu et al. [71]	Mixed Dataset	Improved Adaptive Random Forests (IARF)	99.68%
Deployed	**Mixed Dataset**	**Random Forest**	**99.90%**

6 Conclusion

In conclusion, managing a rapidly transforming network in an SDN, NFV environment is difficult. Nevertheless, employing a method of anomaly detection based on ML can significantly improve the network control automation process. The method detailed in this study provides an efficient method for identifying irregular orchestration behaviors based on service and network status, beginning with gathering data and selection and concluding with model training and validation. It was determined that RF approaches are superior to other classifiers for detecting various system anomalies in the containerized environment. The strategy's objective is to concentrate on operation monitoring, with the anomaly detector detecting any irregularities as soon as they occur and providing tailored recommendations for each registered data point, regardless of whether an anomaly occurred. Failure to detect anomalies promptly can have a negative impact on service efficiency or result in component failures, with little opportunity to mitigate the anomaly. Consequently, the proposed method can potentially improve network administration and service efficiency in an SDN, NFV environment, which has several advantages, such as reduced maintenance costs, improved network operational performance, a simplified network lifecycle, and policies management.

The proposed research presumed that ML techniques are used to analyze network data, spot potential security threats or anomalies associated with instituting security monitoring and management in an SDN-NFV environment, and present remedies to these challenges.

As the utilization of ML techniques for security monitoring and administration in SDN-NFV contexts is a relatively new discipline, several limitations and challenges must be addressed. Among the limitations based on security monitoring and administration for the network services in the orchestration of SDN-NFV environment using ML techniques are: (i) To successfully train models, enormous amounts of high-quality data are necessary. In the case of security monitoring and administration for SDN-NFV environments, however, there may not be sufficient data to train ML models reliably. It can result in erroneous forecasts. (ii) The proposed study assumes the security monitoring and management system deployment in the SDN-NFV environment.

In this context, we identify issues that should be addressed in future research: (i) The investigation of more sophisticated methods of ML that are more resistant to cybercriminal evasion methods; and (ii) The integration of blockchain technology into the system to provide a more reliable and decentralized approach to security monitoring and management.

Acknowledgement: Thanks to our institutes and colleagues who supported us morally.

Funding Statement: This work was funded by the Deanship of Scientific Research at Jouf University under Grant Number (DSR2022-RG-0102).

Author Contributions: All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Availability of Data and Materials: The dataset and material used in this study are publicly available.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] T. Q. Thanh, S. Covaci, M. Corici and T. Magedanz, "Access control management and orchestration in NFV environment," in *2017 IFIP Networking Conf. (IFIP Networking) and Workshops*, Stockholm, Sweden, IEEE, pp. 1–2, 2017.

[2] T. V. Phan and T. Bauschert, "DeepAir: Deep reinforcement learning for adaptive intrusion response in software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2207–2218, 2022.

[3] X. Shi, Y. Li, H. Xie, T. Yang, L. Zhang *et al.,* "An openflow-based load balancing strategy in SDN," *Computers, Materials & Continua*, vol. 62, no. 1, pp. 385–398, 2020.

[4] A. O. Alzahrani and M. J. Alenazi, "ML-IDSDN: Machine learning based intrusion detection system for software-defined network," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 1, pp. e7438, 2023.

[5] A. Kalliola, S. Lal, K. Ahola, I. Oliver, Y. Miche *et al.,* "Testbed for security orchestration in a network function virtualization environment," in *2017 IEEE Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Berlin, Germany, IEEE, pp. 1–4, 2017.

[6] J. Liu, G. Shou, Q. Wang, Y. Liu, Y. Hu *et al.,* "Load-balanced service function chaining in edge computing over FiWi access networks for Internet of Things," arXiv preprint arXiv:2006.08134, 2020.

[7] V. Eramo, A. Tosti and E. Miucci, "Server resource dimensioning and routing of service function chain in NFV network architectures," *Journal of Electrical and Computer Engineering*, vol. 2016, pp. 1–12, 2016.

[8] M. Pattaranantakul, "Moving towards software-defined security in the era of NFV and SDN," Université Paris-Saclay, France, 2019.

[9] V. Medel, R. Tolosana-Calasanz, J. Á. Bañares, U. Arronategui and O. F. Rana, "Characterising resource management performance in Kubernetes," *Computers & Electrical Engineering*, vol. 68, pp. 286–297, 2018.

[10] H. N. Vithlani, D. Marcel, B. Melville, M. Prüm, O. H. Y. Lam *et al.,* "Applicability of remote sensing workflow in Kubernetes-managed on-premise cluster environment," *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 30, no. 1, pp. 38, 2019.

[11] D. Vohra, *Kubernetes Management Design Patterns: With Docker, CoreOS Linux, and other Platforms*. New York City, USA: Springer, 2017.

[12] R. Pérez, M. Rivera, Y. Salgueiro, C. R. Baier and P. Wheeler, "Moving microgrid hierarchical control to an SDN based Kubernetes cluster: A framework for reliable and flexible energy distribution," *Sensors*, vol. 23, no. 7, pp. 3395, 2023.

[13] Y. Jacky and N. A. Sulaiman, "The use of data analytics in external auditing: A content analysis approach," *Asian Review of Accounting*, vol. 30, no. 1, pp. 31–58, 2022.

[14] Z. Muhammad, F. Amjad, Z. Iqbal, A. R. Javed and T. R. Gadekallu, "Circumventing Google Play vetting policies: A stealthy cyberattack that uses incremental updates to breach privacy," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, pp. 1–10, 2023.

[15] F. Shahzad, A. Mannan, A. R. Javed, A. S. Almadhor, T. Baker *et al.,* "Cloud-based multiclass anomaly detection and categorization using ensemble learning," *Journal of Cloud Computing*, vol. 11, no. 1, pp. 1–12, 2022.

[16] C. M. Mathas, O. E. Segou, G. Xylouris, D. Christinakis, M. A. Kourtis *et al.,* "Evaluation of Apache spot's machine learning capabilities in an SDN/NFV enabled environment," in *Proc. of the 13th Int. Conf. on Availability, Reliability and Security*, Hamburg, Germany, pp. 1–10, 2018.

[17] T. Qasim, M. H. Durad, A. Khan, F. Nazir and T. Qasim, "Detection of signaling system 7 attack in network function virtualization using machine learning," in *2018 15th Int. Bhurban Conf. on Applied Sciences and Technology (IBCAST)*, Islamabad, Pakistan, IEEE, pp. 484–488, 2018.

[18] T. Subramanya and R. Riggio, "Machine learning-driven scaling and placement of virtual network functions at the network edges," in *2019 IEEE Conf. on Network Softwarization (NetSoft)*, Paris, France, IEEE, pp. 414–422, 2019.

[19] Z. Munawar, F. Ahmad, S. A. Alanazi, K. S. Nisar, M. Khalid *et al.,* "Predicting the prevalence of lung cancer using feature transformation techniques," *Egyptian Informatics Journal*, vol. 23, no. 4, pp. 109–120, 2022.

[20] A. Khaliq, S. A. R. Kashif, F. Ahmad, M. Anwar, Q. Shaheen *et al.,* "Indirect vector control of linear induction motors using space vector pulse width modulation," *Computers, Materials & Continua*, vol. 74, no. 3, pp. 6263–6287, 2023.

[21] C. Sauvanaud, K. Lazri, M. Kaâniche and K. Kanoun, "Anomaly detection and root cause localization in virtual network functions," in *2016 IEEE 27th Int. Symp. on Software Reliability Engineering (ISSRE)*, Ottawa, ON, Canada, IEEE, pp. 196–206, 2016.

[22] S. Gautam, A. Henry, M. Zuhair, M. Rashid, A. R. Javed *et al.,* "A composite approach of intrusion detection systems: Hybrid RNN and correlation-based feature optimization," *Electronics*, vol. 11, no. 21, pp. 3529, 2022.

[23] M. Shabbir, F. Ahmad, A. Shabbir and S. A. Alanazi, "Cognitively managed multi-level authentication for security using fuzzy logic based quantum key distribution," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, pp. 1468–1485, 2022.

[24] W. A. Khan, F. Ahmad, S. A. Alanazi, T. Hasan, S. Naseem *et al.,* "Trust identification through cognitive correlates with emphasizing attention in cloud robotics," *Egyptian Informatics Journal*, vol. 23, pp. 259–269, 2022.

[25] T. Hasan, F. Ahmad, M. Rizwan, N. Alshammari, S. A. Alanazi *et al.,* "Edge caching in fog based sensor networks through deep learning associated quantum computing framework," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 6138434, 2022.

[26] S. Shahzadi, F. Ahmad, A. Basharat, M. Alruwaili, S. Alanazi *et al.,* "Machine learning empowered security management and quality of service provision in SDN-NFV environment," *Computers, Materials & Continua*, vol. 66, no. 3, pp. 2723–2749, 2021.

[27] A. R. Wani, Q. Rana, U. Saxena and N. Pandey, "Analysis and detection of DDoS attacks on cloud computing environment using machine learning techniques," in *2019 Amity Int. Conf. on Artificial Intelligence (AICAI)*, Dubai, United Arab Emirates, IEEE, pp. 870–875, 2019.

[28] C. Iwendi, S. Khan, J. H. Anajemba, M. Mittal, M. Alenezi *et al.,* "The use of ensemble models for multiple class and binary class classification for improving intrusion detection systems," *Sensors*, vol. 20, no. 9, pp. 2559, 2020.

[29] A. K. S. Ong, F. E. Zulvia and Y. T. Prasetyo, "“The Big One” earthquake preparedness assessment among younger Filipinos using a random forest classifier and an artificial neural network," *Sustainability*, vol. 15, no. 1, pp. 679, 2023.

[30] H. Jiang, Z. He, G. Ye and H. J. I. A. Zhang, "Network intrusion detection based on PSO-XGBoost model," *IEEE Access*, vol. 8, pp. 58392–58401, 2020.

[31] I. H. Abdulqadder, S. Zhou, D. Zou, I. T. Aziz and S. M. A. Akber, "Multi-layered intrusion detection and prevention in the SDN/NFV enabled cloud of 5G networks using AI-based defense mechanisms," *Computer Networks*, vol. 179, pp. 107364, 2020.

[32] C. Cath, S. Wachter, B. Mittelstadt, M. Taddeo and L. Floridi, "Artificial intelligence and the ‘good society’: The US, EU, and UK approach," *Science and Engineering Ethics*, vol. 24, no. 2, pp. 505–528, 2018.

[33] F. Ahmad and K. Ahmed, "Holographic interface management in the age of artificial intelligence," *International Journal of Computer Science and Network Security*, vol. 17, no. 3, pp. 77–86, 2017.

[34] A. Pastor, A. Mozo, D. R. Lopez, J. Folgueira and A. Kapodistria, "The mouseworld, a security traffic analysis lab based on NFV/SDN," in *Proc. of the 13th Int. Conf. on Availability, Reliability and Security*, Hamburg, Germany, pp. 1–6, 2018.

[35] Z. Khalid, M. Rizwan, A. Shabbir, M. Shabbir, F. Ahmad *et al.,* "Cloud server security using bio-cryptography," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 3, pp. 166–172, 2019.

[36] S. Lee, C. Yoon, C. Lee, S. Shin, V. Yegneswaran *et al.,* "DELTA: A security assessment framework for software-defined networks," in *NDSS*, San Diego, CA, USA, 2017.

[37] S. Shah and S. P. Bendale, "An intuitive study: Intrusion detection systems and anomalies, how AI can be used as a tool to enable the majority, in 5G era," in *2019 5th Int. Conf. on Computing, Communication, Control and Automation (ICCUBEA)*, Pune, India, IEEE, pp. 1–8, 2019.

[38] S. Shahzadi, B. Khaliq, M. Rizwan and F. Ahmad, "Security of cloud computing using adaptive neural fuzzy inference system," *Security and Communication Networks*, vol. 2020, pp. 1–15, 2020.

[39] A. Shabbir, M. Shabbir, M. Rizwan and F. Ahmad, "Ensuring the confidentiality of nuclear information at cloud using modular encryption standard," *Security and Communication Networks*, vol. 2019, pp. 1–16, 2019.

[40] L. D. Chou, C. C. Liu, M. S. Lai, K. C. Chiu, H. H. Tu *et al.,* "Behavior anomaly detection in SDN control plane: A case study of topology discovery attacks," *Wireless Communications and Mobile Computing*, vol. 2020, pp. 1–16, 2020.

[41] M. C. Chow and M. Ma, "A secure blockchain-based authentication and key agreement scheme for 3GPP 5G networks," *Sensors*, vol. 22, no. 12, pp. 4525, 2022.

[42] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar and B. Akbari, "Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 374–388, 2018.

[43] Y. Chen, R. Xia, K. Yang and K. Zou, "MFFN: Image super-resolution via multi-level features fusion network," *The Visual Computer*, pp. 1–16, 2023. doi: https://doi.org/10.1007/s00371-023-02795-0.

[44] Y. Chen, R. Xia, K. Zou and K. Yang, "FFTI: Image inpainting algorithm via features fusion and two-steps inpainting," *Journal of Visual Communication and Image Representation*, vol. 91, pp. 103776, 2023.

[45] Y. Chen, R. Xia, K. Zou and K. Yang, "RNON: Image inpainting via repair network and optimization network," *International Journal of Machine Learning and Cybernetics*, vol. 14, pp. 1–17, 2023.

[46] N. Dilawar, M. Rizwan, F. Ahmad and S. Akram, "Blockchain: Securing Internet of Medical Things (IoMT)," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 1, pp. 82–89 2019.

[47] M. Bagaa, T. Taleb, J. B. Bernabe and A. Skarmeta, "A machine learning security framework for IoT systems," *IEEE Access*, vol. 8, pp. 114066–114077, 2020.

[48] I. Farris, J. B. Bernabé, N. Toumi, D. Garcia-Carrillo, T. Taleb *et al.,* "Towards provisioning of SDN/NFV-based security enablers for integrated protection of IoT systems," in *2017 IEEE Conf. on Standards for Communications and Networking (CSCN)*, Helsinki, Finland, IEEE, pp. 169–174, 2017.

[49] D. Behnke, M. Müller, P. B. Bök, S. Schneider, M. Peuster *et al.,* "NFV-driven intrusion detection for smart manufacturing," in *2019 IEEE Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Dallas, TX, USA, IEEE, pp. 1–6, 2019.

[50] A. M. Zarca, J. B. Bernabe, A. Skarmeta and J. M. A. Calero, "Virtual IoT honeynets to mitigate cyberattacks in sdn/nfv-enabled IoT networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1262–1277, 2020.

[51] V. Sharma, V. Verma and A. Sharma, "Detection of DDoS attacks using machine learning in cloud computing," in *Int. Conf. on Advanced Informatics for Computing Research*, Shimla, India, Springer, pp. 260–273, 2019.

[52] G. Ilievski and P. J. R. Latkoski, "Efficiency of supervised machine learning algorithms in regular and encrypted VoIP classification within NFV environment," *Radioengineering*, vol. 29, no. 1, pp. 243–250. 2020.

[53] J. Hong, S. Park, J. H. Yoo and J. W. K. Hong, "A machine learning based SLA-aware VNF anomaly detection method in virtual networks," in *2020 Int. Conf. on Information and Communication Technology Convergence (ICTC)*, Jeju, Korea (South), IEEE, pp. 1051–1056, 2020.

[54] T. Hassan, W. A. Khan, F. Ahmad, M. Rizwan and R. Rehman, "Edge caching framework in fog based radio access networks through AI in quantum regime," in *Int. Conf. on Intelligent Technologies and Applications*, Bahawalpur, Pakistan, Springer, pp. 711–722, 2019.

[55] M. A. Kourtis, G. Xilouris, G. Gardikis and I. Koutras, "Statistical-based anomaly detection for NFV services," in *2016 IEEE Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Palo Alto, CA, USA, IEEE, pp. 161–166, 2016.

[56] D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta *et al.,* "Optimal virtual network function placement in multi-cloud service function chaining architecture," *Computer Communications*, vol. 102, pp. 1–16, 2017.

[57] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong *et al.,* "Deep anomaly detection for time-series data in Industrial IoT: A communication-efficient on-device federated learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6348–6358, 2020.

[58] S. M. Kasongo and Y. J. C. Sun, "A deep learning method with wrapper based feature extraction for wireless intrusion detection system," *Computers & Security*, vol. 92, pp. 101752, 2020.

[59] Z. Wang and A. M. Fey, "Deep learning with convolutional neural network for objective skill evaluation in robot-assisted surgery," *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 12, pp. 1959–1970, 2018.

[60] J. Aiken and S. Scott-Hayward, "Investigating adversarial attacks against network intrusion detection systems in sdns," in *2019 IEEE Conf. on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Dallas, TX, USA, IEEE, pp. 1–7, 2019.

[61] P. Hadem, D. K. Saikia and S. Moulik, "An SDN-based intrusion detection system using SVM with selective logging for IP traceback," *Computer Networks*, vol. 191, pp. 108015, 2021.

[62] D. Zhou, Z. Yan, Y. Fu and Z. Yao, "A survey on network data collection," *Journal of Network and Computer Applications*, vol. 116, pp. 9–23, 2018.

[63] A. P. Ferreira and R. Sinnott, "A performance evaluation of containers running on managed Kubernetes services," in *2019 IEEE Int. Conf. on Cloud Computing Technology and Science (CloudCom)*, Sydney, NSW, Australia, IEEE, pp. 199–208, 2019.

[64] J. Shah and D. Dubaria, "Building modern clouds: Using docker, kubernetes & google cloud platform," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conf. (CCWC)*, Las Vegas, NV, USA, IEEE, pp. 0184–0189, 2019.

[65] L. Mercl and J. Pavlik, "Public cloud Kubernetes storage performance analysis," in *Int. Conf. on Computational Collective Intelligence*, Hendaye, France, Springer, pp. 649–660, 2019.

[66] X. Jing, Z. Yan and W. Pedrycz, "Security data collection and data analytics in the Internet," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 586–618, 2019.

[67] Y. Zhou, G. Cheng, S. Jiang and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Computer Networks*, vol. 174, pp. 107247, 2020.

[68] S. M. Kasongo and Y. Sun, “Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset,” *Journal of Big Data*, vol. 7, pp. 1–20, 2020.
[69] D. Jing and H. B. Chen, “SVM based network intrusion detection for the UNSW-NB15 dataset,” in *2019 IEEE 13th Int. Conf. on ASIC (ASICON)*, Chongqing, China, IEEE, pp. 1–4, 2019.
[70] F. Z. Belgrana, N. Benamrane, M. A. Hamaida, A. M. Chaabani and A. Taleb-Ahmed, “Network intrusion detection system using neural network and condensed nearest neighbors with selection of NSL-KDD influencing features,” in *2020 IEEE Int. Conf. on Internet of Things and Intelligence System (IoTaIS)*, Bali, Indonesia, IEEE, pp. 23–29, 2021.
[71] Z. Niu, J. Xue, D. Qu, Y. Wang, J. Zheng *et al.,* “A novel approach based on adaptive online analysis of encrypted traffic for identifying Malware in IIoT,” *Information Sciences*, vol. 601, pp. 162–174, 2022.