Performing Dynamic Malware Analysis in Software Defined Network using Machine Learning Techniques

Vasantharaj Karunakaran[1], L. Priya[2]

[1]Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, Tamilnadu, India, vasantharajk35@gmail.com
[2]AP, Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, Tamilnadu, India, priyal@hindustanuniv.ac.in

Methods for analyzing harmful or benign packets are crucial for enhancing security systems. Current security methods can only detect the packets to a limited extent. A malware analysis architecture combining the features of the Controllers that may use any Machine Learning techniques that can be developed thanks to the pliability of Software-Defined Networking (SDN). In this research, we present a Secure SDN Architecture along with Machine Learning techniques, namely a Support Vector Machine-Dynamic Malware Analyzer (SVM-DMA) and Random Forest-Dynamic Malware Analyzer (RF-DMA) Controlled by a POX Controller. We use two machine learning techniques, including Support Vector Machine (SVM) and Random Forest, to achieve improved accuracy using a Confusion Matrix. SVM achieves a 96% detection accuracy on the chosen NSL-KDD dataset sub-features. The PACKET_IN event at the controller is used to analyze packet and flow behaviour by periodically retrieving flow data from the Open Flow switches. In the random forest, we observed all 41 attributes of the NSL-KDD dataset, and the features were bootstrapped by splitting them into a decision tree. An accuracy of 96.5 has been achieved by using this method.

Keywords: dynamic malware, software defined network, SVM, RF, NSL.

1. Introduction

There are several reasons to undertake dynamic malware analysis[1], but the main one is comprehending and reducing the risks associated with the dangerous software. Security experts may find and identify new malware strains by dynamically studying the behaviour of the infection. This involves locating patterns or signatures that intrusion detection systems[2] or antivirus software may employ to detect malware. The information gathered from internet traffic monitoring can help network managers deploy resources more wisely and with more knowledge [3]. The difficulty of classification in a traditional network traffic system is in the coding; in a machine learning system, the difficulty is in the net suggestions that are gathered and the algorithm that is employed.

The essential element of machine learning algorithms is their ability to learn. The development of this talent comes from consistent practice and advancement. Each type of network traffic is limited to specific statistical features of the network, such as packet length, duration, and transmission inter-arrival time. With the help of the provided training data set, the computer is able to identify this uniqueness in the network pattern. The structure of the gathered internet data is determined by machine learning. Differentiating between email clients, file transfers, video streaming services, and online browsers can be difficult.

To control the rapid increase in network traffic and make effective use of network resources, a critical network management solution is required. Hardware for networking has to incorporate intelligence to make organization, optimization, maintenance, and administration easier. It is difficult to employ machine learning for device control due to the flexibility of the networking architecture. It is feasible to integrate intelligence into interface devices thanks to the SDN platform. Network resource organization, optimization, maintenance, and administration are hampered by the intricate structure of network information. One option is to give network devices intelligence. The control plane and the transmitted data are kept separate in SDN. The Open Flow protocol serves as the downstream interface between the controller and the devices. The controller may connect with network applications using the north bound interface [4]. The network dynamics of the topology[5] are recorded and sent to the controllers for decision-making and network device management. Data analytics and an adaptive control mechanism are offered to handle network dynamics and deliver appropriate control instructions. In the form of network status, such as connection failures or topological changes, the feedback mechanism provides network topology updates.

There are three different topologies for wired network scenarios: tree-like topology, straight topology with the same controls as the host, and single switch with 'n' hosts. A custom topology can be built based on the demands of the user. The development of infrastructure, such as GPUs, data processing frameworks like Spark and Hadoop, and machine learning libraries like scikit-learn[6] and TensorFlow[7], has opened up the possibility of incorporating intelligence into networking gear.

The system's core components are the controllers. The active network resource organization is made possible by this controller. It is possible to keep an eye on the devices because the controller features a global network view. The POX controller manages the data flow between the devices [8]. Flow rubrics are manipulated using the Open Flow. The devices linked to the controller and architecture interact with each other through the southbound interface. It is feasible to communicate with the northbound line. The POX controllers is utilized for topology management and distant monitoring[9]. The'mininet' tool[10], [11], [12] is used to construct the network topology. 'tcpdump' captures real-time (continuous) communication in the format of 'pcap' files. The 'netmate' flow generator is used to create the flow features. The labels are applied to the data that has been recorded. Different ML methods are used to make the prediction.

Dynamic Malware Analysis has seen a surge in interest in machine learning (ML) approaches. The SVM, a member of the supervised class of machine learning, is among the most widely used ML techniques[13], [14]. SVM is a classification method shown to help resolve various issues, including image processing, pattern identification, and cyber-security, notably in Dynamic Malware Analysis. An SVM is a type of neural network that resembles a perceptron and is best suited for binary pattern classification—more precisely, linearly separable patterns. This approach produces a classifier that can forecast whether a given network traffic will be normal or abnormal. To get the best learning and generalizing capacity, SVM looks for the optimal trade-off between model complexity and learning ability based on a small sample of data. Malware analysis may achieve greater detection accuracy with the use of the SVM approach, and as a result, this area of study is becoming increasingly important[15]. Support Vector Machines are beneficial in analyzing the packets in a Software-Defined Networking(SDN) environment[16], [17] to watch for malicious activities or policy infractions on the network. The Software-Defined Network - Dynamic Malware Analyzer (SVM-DMA) will notify the SDN controller whenever it detects suspicious activity or a policy violation. We approach NSL-KDD Dataset to perform the Dynamic Malware Analysis. The controller will then take one of two actions: if the analysis predicts the packet behaviour correctly during a PACKET_IN event, it logs the packet information; if it fails to analyze the behaviour of the packet that occurs during the analysis of flow statistics, it installs a rule in the affected switch to drop the packets of the abnormal flow. In the unlikely event that the analysis is unable to predict the behaviour correctly, the selective monitoring of SVM-DMA and flows during a PACKET_IN event allows for the execution of an IP traceback[18]. Furthermore, compared to recording every packet, selective IP packet logging will result in a considerable reduction in the amount of log records that must be kept up to date.

Performing only SVM-DMA to analyze the malware is insufficient as we use limited data attributes for our work. One more method will be exact, (i.e) Dynamic Malware Analyzer using Random Forest method known as RF-DMA. In this method, we utilize all 41 features of the dataset[19]. As it is a Random forest technique, splitting the dataset into a decision tree is a mandatory task. This task is accomplished by the Bootstrapping technique[20], which splits into a decision to predict the behaviour of the packets by voting. An inventive way to improve network security is to use Random Forest for dynamic malware Analysis in combination with Software-Defined Networking (SDN). From the network traffic data that SDN controllers have acquired, dynamic characteristics are extracted. Features like packet header data, flow statistics (including flow length, packet count, and byte count), and communication patterns (such source-destination IP pairings and port numbers), are incorporated in before applying Bootstrapping method.

We proposed a Support Vector Machine-Dynamic Malware Analyzer (SVM-DMA) and Random Forest-Dynamic Malware Analyzer (RF-DMA) to perform the Dynamic Malware Analysis. We utilized the features of POX Controllers installed along with the Mininet tool. Regarding the dataset, we approached NSL-KDD for both of the methods in which the former utilized 8 attributes of 41 and all the attributes are utilized in the later one.

Further the 8 attributes became successful and produced good results in SVM-DMA and by the concept of bootstrapping and by splitting into Decision tree the RF-DMA drafted good results.

The overall work deals with Section 2, the literature survey; Section 3 elaborates on the proposed methodologies, where Section 4 discusses the experimental outcomes, with performance measured in terms of Accuracy, Precision, Recall, and F-Score and finally Section 5 concludes with a discussion of the approaches' success and their potential application in the future.

2. Related Works

An important design choice that impacts a range of network problems, such as latency, resilience, energy efficiency, load balancing, and others, is where the controller(s) are placed inside the SDN control plane. In this study,[21] offered a comprehensive analysis of the controller placement problem (CPP) in SDN. Additionally, they discussed and emphasized the significance of the CPP in SDN. We introduce the conventional CPP formulation together with the corresponding system model. We also discuss several alternatives for CPP modelling and associatedindicators.

Because SDN is growing so rapidly, online malware developers are taking full advantage of this by creating new types of malwares and spreading them through several channels to affect millions of users. Malware poses a serious risk to computer security. Several research projects have been conducted to improve the efficacy of detection methods[22]. As demonstrated by [23], the malware's objective is to corrupt every file on the system and perform destructive actions. Malware is difficult to detect and defensive mechanisms usually fail as a result of signature-based security solution software. Using the Cuckoo Sandbox, sandboxing technology, as suggested by [24], detects malware's non-trusted code and analyses its behaviour.

Ransomware is regarded to have an unidentified victim. Ransomwares are malware programs that offer little opportunities for data recovery [25] and may be exceedingly time-consuming to deal with once they begin to propagate. According to [26], the controller connects exclusively to SDN switches using the OpenFlow protocol, adding the necessary entries to their flow tables upon detection of malicious behavior. Specifically, the controller monitors the network traffic that potentially infected devices produce and takes immediate action.

Additionally, Windows promotes malware analysis. The most pressing issue facing the research community at the moment is the detrimental danger posed by malware, which is always developing new categories. Various techniques have been used, but they can't find unknown malware. In order to do this, the suggested approach integrates dynamic malware analysis techniques with machine learning for Windows malware categorization and detection. It involves running the executable in a restricted environment with few uncovered resources for execution and post-execution analysis of the behaviour patterns statistics using the Cuckoo Sandbox Tool[27].

Static and dynamic analysis are the two basic categories into which malware analysis techniques fall. Although some code obfuscation strategies can reduce the effectiveness of static analysis, static analysis tools nevertheless retrieve data from the source code [28]. Dynamic analysis concentrates on behaviour data obtained from isolated program execution environments. Dynamic analysis is often more useful in this approach than static analysis since the disguised malware must exhibit genuine behaviour while operating.

[29]ProposedDMalNet, a dynamic malware analysis system based on API call graph learning and feature engineering. First, they employ several lightweight feature encoding approaches (e.g., Word2Vec, feature hashing, similarity encoding) to extract semantic features from API names and heterogeneous arguments, taking into account the peculiarities of different types of data. Second, they build an API call graph that represents the characteristics of the API name and arguments, and transformed the relationship between API calls into the structural data of the graph.

There has been a lot of interest in the field of Android malware detection in both academia and business. Studies on malware families in particular have helped with malware behaviour analysis and detection. However, identifying malware family traits and the features that can characterise a specific family have received less attention in previous research. In order to enable fingerprinting the malware families with these features, we are motivated to investigate the key features that can categorise and describe the behaviours of Android malware families. [30]Proposed twenty key features in three categories are used to build the fingerprints of ten malware families. Results of extensive SVM experiments show that the accuracy of the malware family classification ranges from 92% to 99%.

There is no perfect way to combat malware, which is a menace to contemporary computing that is only becoming more and more prevalent. The Internet's expansion may be attributed to the exponential rise in users of its different services, including social networks and cloud storage. Malware spreads quickly over the Internet and can occasionally result in a large-scale attack, such as a botnet attack. Antivirus software serves as a first line of defence against malware. Unfortunately, since many of them continue to use a signature-based strategy and malware writers are aware of this, this line of security is not very successful. They adapt their software accordingly, and as a result, malware infiltrates systems covertly. As a result, a behaviour-based or heuristic-based strategy is always required to combat various malware families. The most effective methods for combating malware are machine learning and reverse engineering, and several researchers are dedicated to solving various malware attack-related issues. Therefore, in order to eradicate malware from infected personal computers (PCs) or systems, [31] summarizes all of the methods currently in use for malware detection and analysis, including static and dynamic analysis.

Many strategies are employed by modern malware to avoid detection by static and dynamic analysis technologies. Existing dynamic analysis solutions either employ a higher privilege component that does the analysis or modify the malware that is already operating. While sophisticated spyware may quickly identify the former, the latter frequently results in a large performance overhead. [32]Put out a technique that analyses malware while the operating system is running. Additionally, a hypervisor hides the analytic component, making it invisible to the operating system and its programs. The system's efficiency study indicates that the resulting performance overhead is minimal.

The increasing interest in the use of software-defined networks (SDNs) for both wired and wireless applications has been greatly sparked by the increasing availability of mobile devices and applications, the advancements in virtualization technologies, and the development of cloud-based distributed data centres. Network security may be greatly enhanced by using standards-basedsoftware abstraction between the network control plane and the underlying data forwarding plane, which includes both virtual and physical devices. In order to protect

SDNs against malicious activity, a complete system that takes use of the inherent properties of SDNs and applies data mining to identify and categorize dangerous flows in the SDN data plane is presented in this study by [33], [34]. The system's design and workings are explained, with a focus on flow rule creation and flow categorization. In the SDN testbed environment, which replicates common SDN flows, the idea was validated. The trials verified that the system can be successfully deployed in SDNs to reduce dangers brought about by various malevolent intrusion activities. The outcomes demonstrate that, in comparison to alternative methods, our combination of data mining approaches offers superior malicious flow identification and categorization.

Information Classification is a typical work in Machine Learning [35], [36].Anticipate that a few information points will fit into one of two classes apiece, and our goal is to determine which class another information point will go into. Scientists are well-known for increased traffic characterisation in addition to grouping. Support Vector Machines (SVM) allow this model to address the overfitting problem. The scientists dissected established risk evaluation methodologies and investigated networks security risk assessment. In view of the main gamble minimization principle, SVM has shown to be a progressive kind of learning machine approach rather than prior gamble evaluation comes near.

SVM gives various benefits with regards to tending to design acknowledgment issues with short example sizes, nonlinearity, and high dimensionality SVM and twofold tree standards are made sense of inside and out and afterward used to organize security risk evaluation. They exhibited that the SVM strategy has higher Classification accuracy, better speculation execution, and less learning and testing time when contrasted with the ASVM(Advanced Support Vector Machine)[37] technique regarding arrangement accuracy, speculation execution, and learning and testing time, particularly for little examples.

There are other SVM studies that do not make use of the widely recognized standard datasets. For example, [38] used the SVM classifier in analyzing the malwares test on the Forest Cover Type dataset, and only 61.48% of the tests were successful. On the ADFA-LD dataset, [39] ran an intrusion detection test with SVM, and only obtained an accuracy of 76.2%. However, other efforts, such as[40] and [41], which employ the outdated KDD'99 dataset, which is less dependable than the NSL-KDD dataset[42], [43], nevertheless obtained high accuracy detection rates of 86.72% and 95.1%, respectively.

In order to determine the harmful intent of a program or piece of code, dynamic malware analysis entails watching how it behaves in a controlled setting. The characteristics are retrieved and then classified using a random forest method. Applying this method to a data set results in the clustering of the data into groups and subgroups, with a subset of the data being used as a training set. An arrangement like a tree that is produced by connecting data points to groups and subgroups is called a decision tree. After then, the computer uses a number of trees to form a forest. But since the variables are selected at random for every split in the tree, every tree is distinct. With the exception of the training set, the residual data is used to predict which forest tree generates the best classification of data points, and the output is the tree with the highest predictive power. Next, each program's kind is identified using a set of labels, where 1 represents malware and 0 represents benign files, as per [44]. With each node of the decision tree, the training set is divided into two subsets with different labels by reducing the uncertainty of the class labels.

3. Methodology of the Proposed System

A. SVM-DMA

In this article, we proposed a hybrid frame work that employs SVM along with the SDNin which the POX controller [45] features are incorporated. Fig 1 shows the block diagram of the 8-attributed featured SVM algorithm.

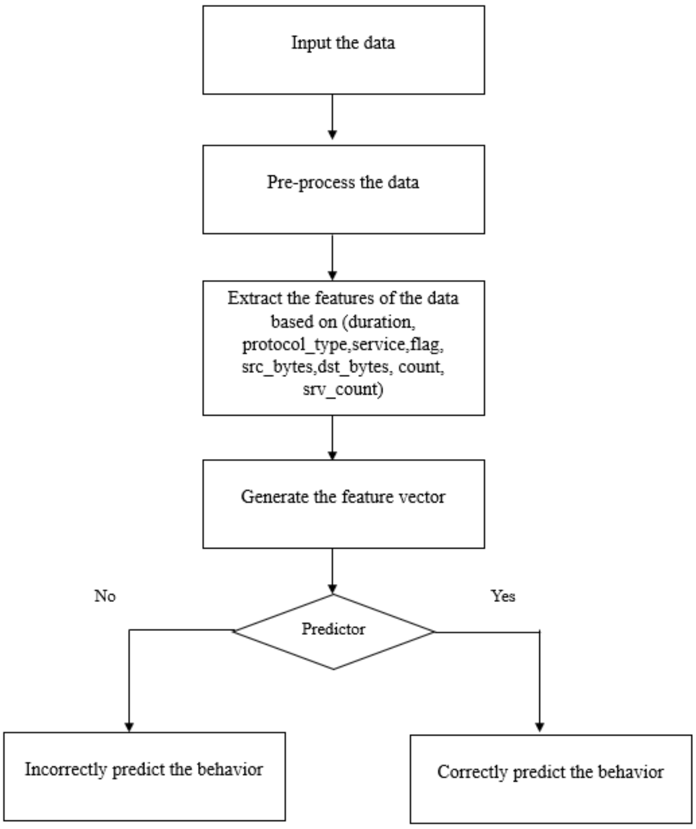
Input the data
Pre-process the data
Extract the features of the data based on (duration, protocol_type,service,flag, src_bytes,dst_bytes, count, srv_count)
Generate the feature vector
No
Yes
Predictor
Incorrectly predict the behavior
Correctly predict the behavior

Fig. 1. Workflow of SVM-DMA

In this article, we present a hybrid framework that employs SVM and SDN along with the POX Controller equipped to outperform the Malware Analysis as shown in Fig 1. Then we present SVM-DMA algorithm which are the prototype of our proposed model and later we develop an SVM-DMA architecture which incorporates the block diagram of Fig 1.

Table 1. Pseudo code of SVM-DMA(Proposed Method)

```
Install POX Controller
if ofp_in then
    packet_features = extract_8features(ofp_in)
    result = SVM_DMA(packet_features)
if result == 1 then
log_packet(ingress_ip,src_ip,dst_ip,protocol,eth_type,src_port,dst_port,t_stamp)
endif
elseifofp_flow_stats_reply then
    packet_features = extract_8features(ofp_stats_reply)
    result = SVM_DMA(packet_features)
else
    take_action(drop the packet)
endif
endif
```

SVM is used as the supervised machine learning technique and executed as an application at the SDN controller to accomplish the DMA. Fig. 1 depicts the entire suggested system architecture and includes eight components, such as an IP traceback mechanism and a logging scheme, to demonstrate the whole SVM methods. The architecture, which is shown in Fig. 2, is made up of three primary components: an IP traceback module, a flow collector, and logging of suspicious packets or flows.

Flow Collector: This module gathers all the flow information, including source IP, destination IP, protocol, source port, destination port, etc. It is activated by two events: first, the arrival of an OFPT_PACKET_IN message[2]; and second, a timer function. After that, DMA is used to further analyze these combined properties.

Logging of suspicious packets/flows: The anomaly detection module receives a selection of a packet's characteristics at the PACKET_IN event. The SDN controller logs the packet's chosen headers along with other pertinent information if the DMA detects a policy violation.

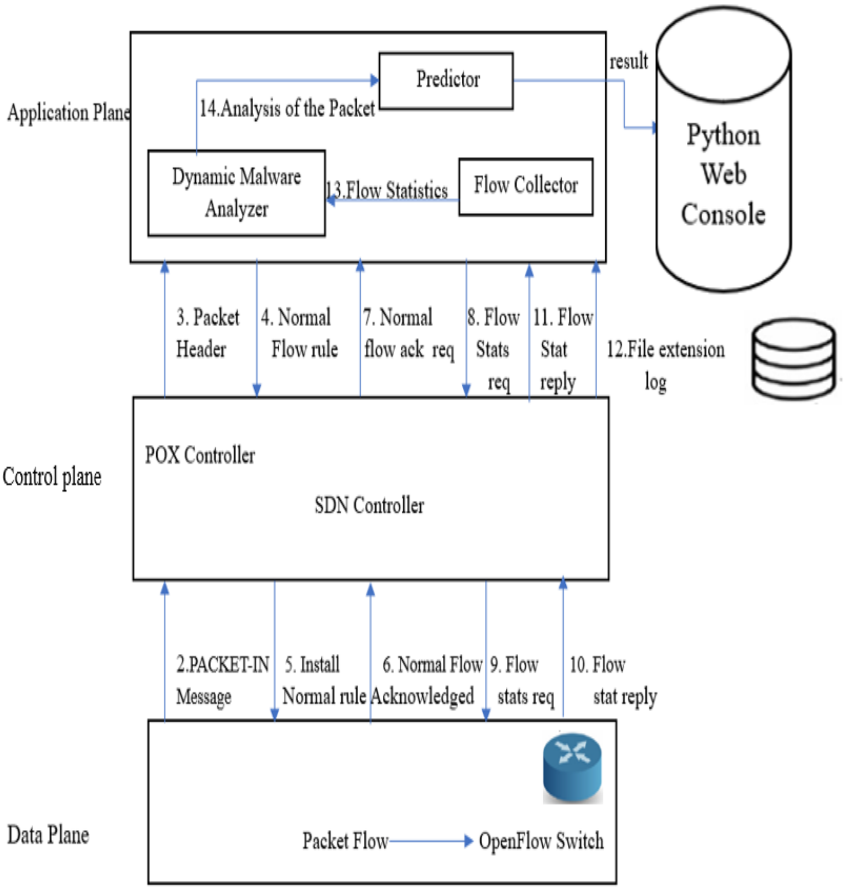

Fig. 2. Architecture of SVM-DMA

Python's Numpy, Scikit-learn, and Tensorflow libraries, together with the POX controller, are used to create the SVM-based DMA as an application [6], [7]. Out of the 41 features in the NSLKDD dataset, the module is qualified by extracting only the 8 features that are indicated in Table 2. The OpenFlowin_port field and the eight features are extracted by the POX controller from the OFPT_PACKET_IN message header during a PACKET_IN event. After the eight characteristics have been recovered, the packet's prediction scheme is applied to the trained model by the SVM-DMA module, which uses this information to determine if the packet is malignant or not. The ofp_flow_stats_request message is used by the POX controller and OpenFlow switch in the network.

The statistics data for each flow item in the table is contained in an ofp_flow_stats_reply message[2] that is provided to the Controller by the OpenFlow switches upon receipt of the flow_stats request. After extracting the eight characteristics from the flow-stats replies from the OpenFlow switches, the Controller uses the SVM classifier to forecast the extracted data against the training model.

The above-mentioned objectives are accomplished by the following information mentioned below:

The formal definition of Hyperplane is given by:

$y = V^T x + b_0$ (1)

V is the weight vector (Magnitude of the weight vector i.e determines the orientation of the separation of the Hyperplane)

b represents the distance between the origin and the plane.

The condition for the positive and negative samples is given as

$$\frac{(x2-x1)\ \vec{V}}{|V|}$$

$$\frac{x2.\vec{V} - x1.\vec{V}}{|V|} \qquad (2)$$

For positive samples, apply y=1

$1(\vec{V} . x1) + b = 1$

Then V. x1 = 1-b (3)

For negative samples, apply y=-1

$-1(\vec{V} . x2) + b = 1$

Then V . x2 = -1-b (4)

Substituting 3 and 4 in 2

$\frac{-1-b-(-1-b)}{|V|} = 1-b+1+b$

Now, 2 = d (2 margins created on both positive and negative sides of the plane)

B. RF-DMA

A Random Forest method along with SDN in which the features of POX Controller are incorporated.

A RF is made up of many decision trees. A decision tree is a model that resembles a tree and shows every scenario that might occur from a choice using the branching approach [46]. The tree is made up of leaf, branch, and root nodes, as well as directed edges and nodes. Training data is aggregated to form the root node. The categorization concepts are represent-ed by the branch nodes. Rules will be used to categorize the data that is delivered to the branch nodes. The ultimate results of categorization are represented by leaf nodes. Every route from the root node to the leaf node symbolizes a distinct categorization procedure.Figure 3 shows the workflow of RF-DMA.

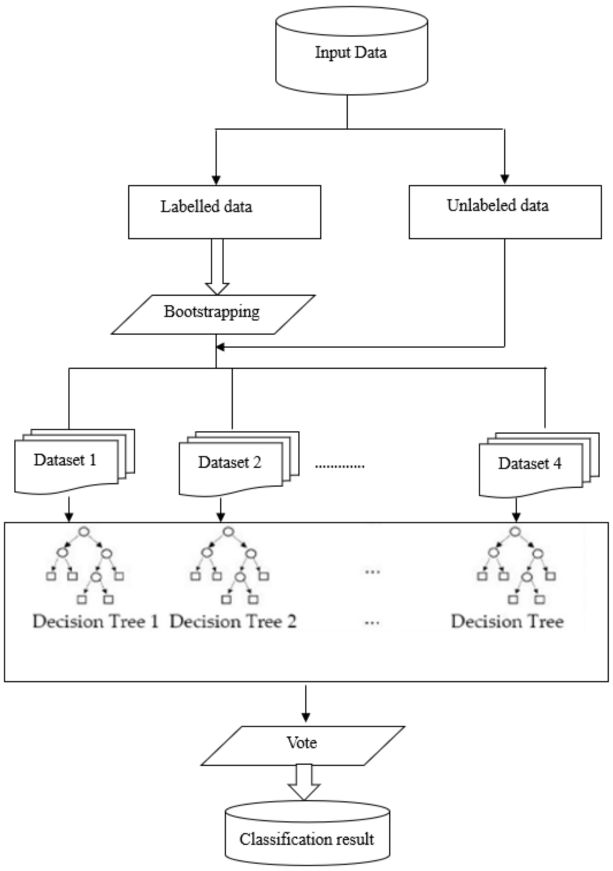

Fig. 3. Work Flow of RFA-DMA Model

Three phases are included in the RF algorithm's training and classification process, as seen in Fig. 3.

i)After splitting the original dataset into the training set and test set, N datasets are created using the Bootstrap technique, which involves taking samples from the training set N times. We have chosen the value of N=4 for our job.
ii) To classify 4 datasets, construct 4 decision trees as classification models $\{h_1(x), h_2(x),..., h_n(x)\}$. We construct a random vector for the nth tree.If there are enough trees, then $\hbar n(x) = \hbar(x,\theta_n)$. A collection of models that vary based on θ_n values may be shown in the following way:
(x,θ_n), where n = 1, 2,..., 4 }, (1)
iii)Voting through the N classification results and selecting the class that was expected to occur the most frequently as the final output without decision tree pruning. Equation (2) displays the voting decision:

$$H(x) = \sum^{4} I(h_n(x) = y) \qquad (2)$$

n=1

y means the prediction of the variable y using the nth tree with the variable x; I(·) is the indicator function.

Then we present RF-DMA algorithm which are the prototype of our proposed model and later we develop an SVM-DMA architecture which incorporates the block diagram of Figure 3.

Algorithm for RF-DMA
Step 0 : Select the input data.
Step 1 : Split the data into 4 different sets by using Bootstrapping.
Step 2 : Construct a decision tree from a given data by encapsulating the features
Step 3 : Average the decision tree based on the Voting procedure.
Step 4: Select the most voted prediction as a result.
Step 5: Consolidate the correctly predicted and incorrectly predicted as the final result.

From the algorithm and the Fig 3, we have incorporated the RF-DMA along with POX Controller features, which is the novelty of our work.

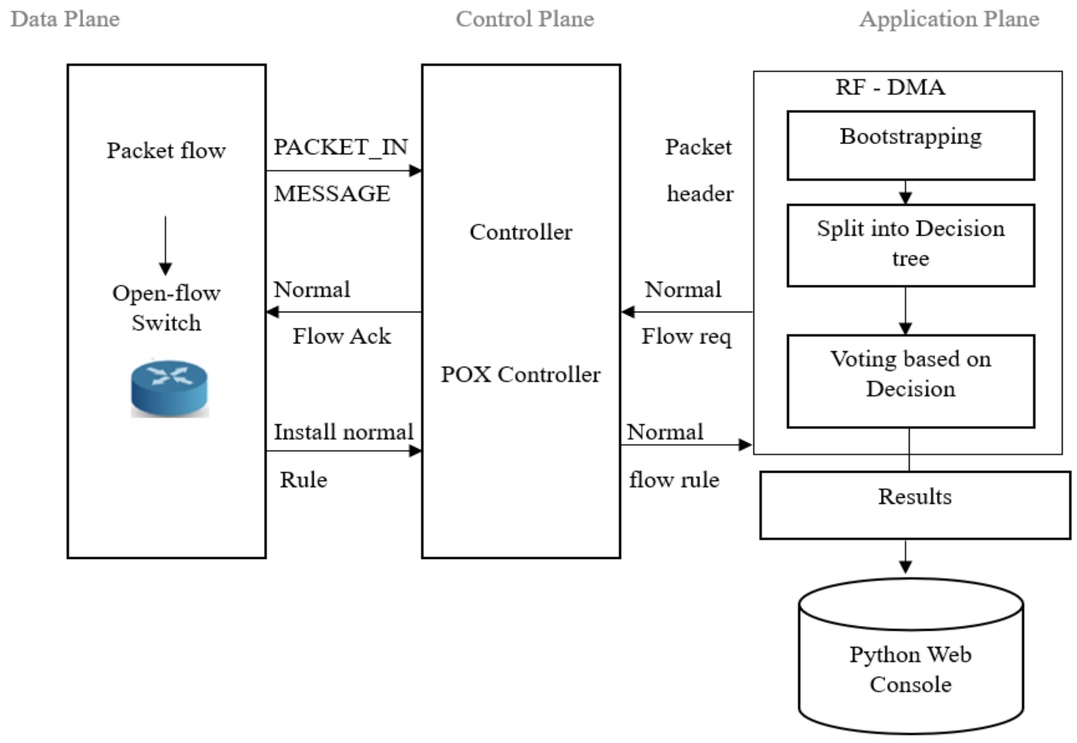

Fig. 4. Architecture of RF-DMA

Evaluation of different feature combinationsSerial No. Selected features Accuracy (%)

Table 2. Selected features of the Data

1 [4,38,29,34,37,28,25,35,3,2,22,1,32,5,36,23,33,40,31,39,7,9,24,0,12,30,11,26,27,21]
2 [4,38,29,34,37,28,25,35,3,2,22,1,32,5,36,23,33,40,31,39,7,9,24,0,12,30,11,26,27]
3 [4,38,29,34,37,28,25,35,3,2,22,1,32,5,36,23,33,40,31,39,7,9,24,0,12,30,11,26]
4 [4,38,29,34,37,28,25,35,3,2,22,1,32,5,36,23,33,40,31,39,7,9,24,0,12,30]

Table 2 shows that the combination at serial number 4 produces the maximum accuracy of 96%; as a result, the following qualities were chosen as features for the model: [4,38,29,34,37,28,25,35,3,2,22,1,32,5,36,23,33,40,31,39,7,9,24,0,12,30]

4. Results and Discussion

Our work's trials are conducted on a PC with an Intel(R) Core (TM) i7-7700 CPU operating at 3.6 GHZ and 16 GB of RAM, running Ubuntu 18. Our suggested work was implemented in a Python environment, and the Scikit-learn module [6] was used to create the SVM and RF classifier.

We employ the conventional performance assessment measures, such as the confusion matrix, precision, recall, F1-measure, and accuracy metrics, to assess the effectiveness of our suggested strategy [47], [48]. The following definitions of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are used to compute these metrics in turn:

TP: Is the quantity of actual records appropriately categorized
•TN: Is the quantity of unactual records appropriately categorized
•FP: Is the quantity of unactual records misclassified
• FN: Are the actual records numbers misclassified

Confusion Matrix can obtain results in terms of Accuracy, Precision, Recall and F-Score
N=950(Total number of values) for SVM-DMA
Total classification of Files as .exe, .dll, .api, others
i)Accuracy : TP + TN / Total
True Positive values: 854
True Negative values: 50
904/950: 0.96 = 96%

ii)Precision: TP / TP + FP
True Positive values: 854
True Negative values: 76 = 92%

iii)Recall = Actual Classified/Actual
290/297 + 283/295 + 318/322 + 35/36= 89%

iv)F-Score = F-Score : $2* \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$

$2* \frac{0.81}{1.81}$ => 0.92 = 92%

Confusion Matrix provides us with an output matrix that details the model's overall performance. It can be displayed as a table with two scopes, "Actual" and "Predicted." Additionally, there are "True Positives (TP)," "True Negatives (TN)," "False Positives (FP)," and "False Negatives (FN)" in each dimension.

The figure 5 represents the comparisons of the results that we obtained.

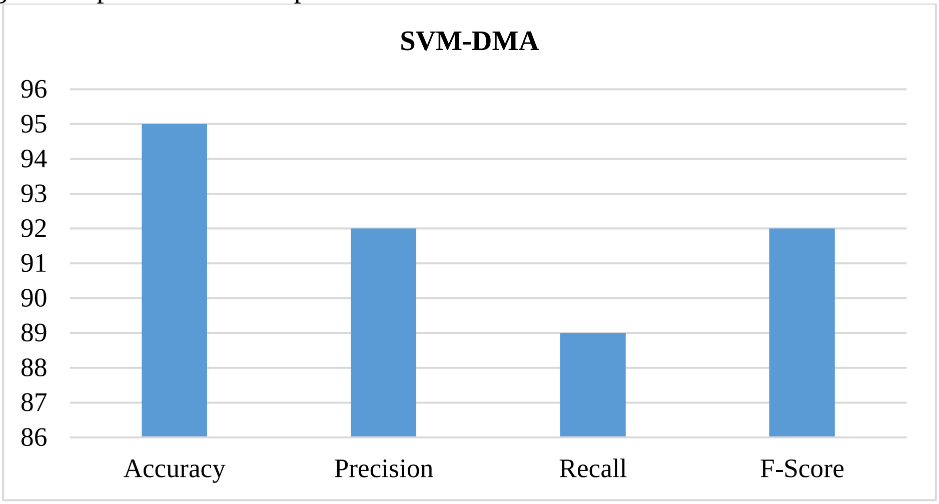

Fig. 5. Performance Comparison of SVM-DMA

The below Figure 6 represents the comparison of our work with other existing methods.

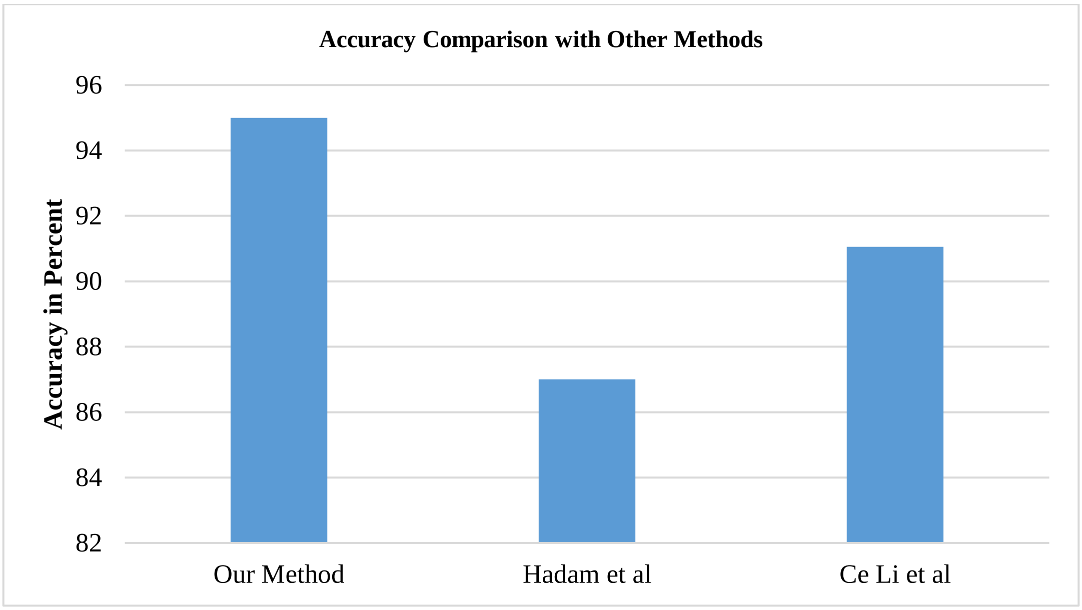

Fig. 6. Accuracy with other Methods

Again, we perform the same Confusion Matrix for RF-DMA method.

Results can be obtained by Confusion Matrix in terms of Accuracy, Precision, Recall and F-Score
N=950(Total number of values).
Total classification of Files as .exe, .dll, .api, others

i)Accuracy : TP + TN / Total
True Positive values : 865
True Negative values : 50
915/950 : 0.96 = 96%
ii)Precision : Correctly Predicted/Total Predicted
285/292+278/281+317/331+35/46
= 93%

iii)Recall = Actual Classified/Actual
290/297 + 283/295 + 318/322 + 35/36
=>91%
iv)F-Score = $2* \frac{\text{Precision*Recall}}{\text{Precision + Recall}}$

$2* \frac{0.83}{1.83}$ => 0.93 => 93

The figure 7 represents the comparisons of the results that we obtained.

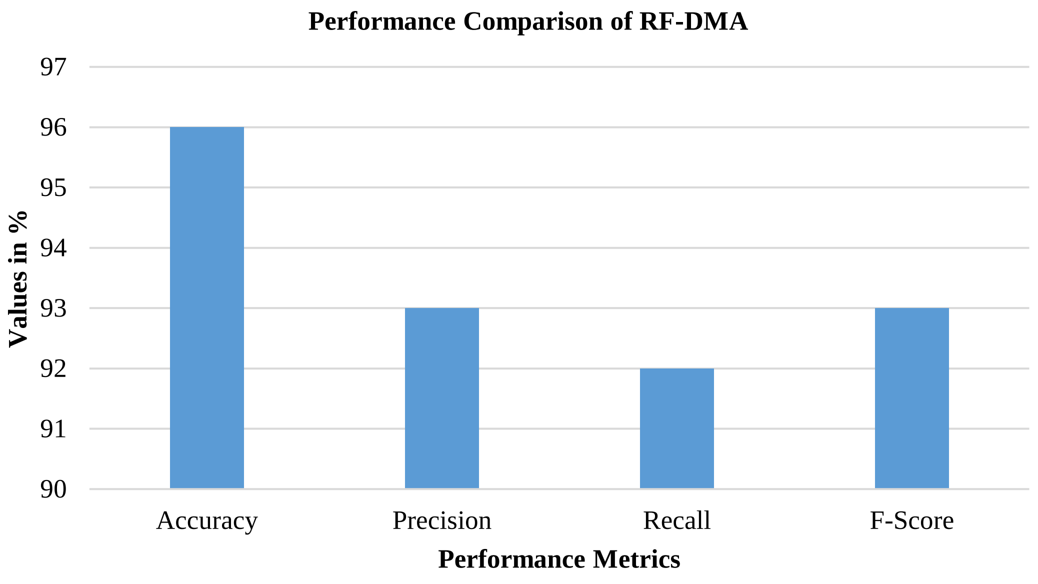

Fig. 7. Performance Comparison of SVM-DMA

The below Figure 8 represents the comparison of our work with other existing methods

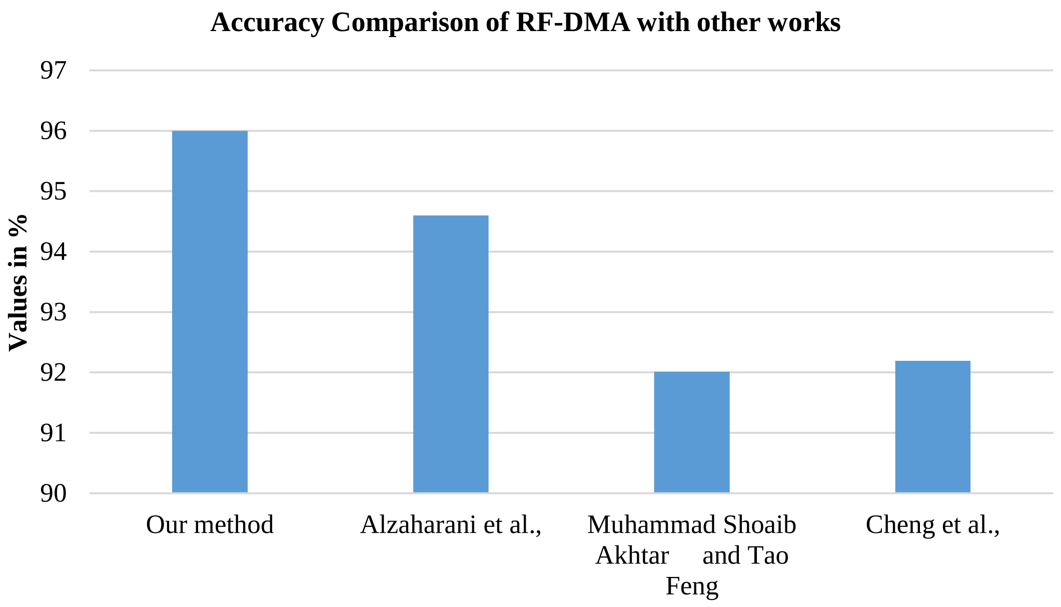

Fig. 8. Performance Comparison of RF-DMA

Table 3. Represents the comparison results of SVM-DMA and RF-DMA

S.NO	**Supervised learning model**	**Confusion Matrix (N = 950)**				
1	**SVM**	**Actual/ Predicted**	**.exe**	**.dll**	**.api**	**Others**
		.exe	271	5	7	2
		.dll	12	268	8	11
		.api	2	3	315	0
		Others	0	0	0	46
2	**Random Forest**	**.exe**	285	0	9	5
		.dll	6	278	5	6
		.api	1	2	317	0
		Others	0	1	0	35

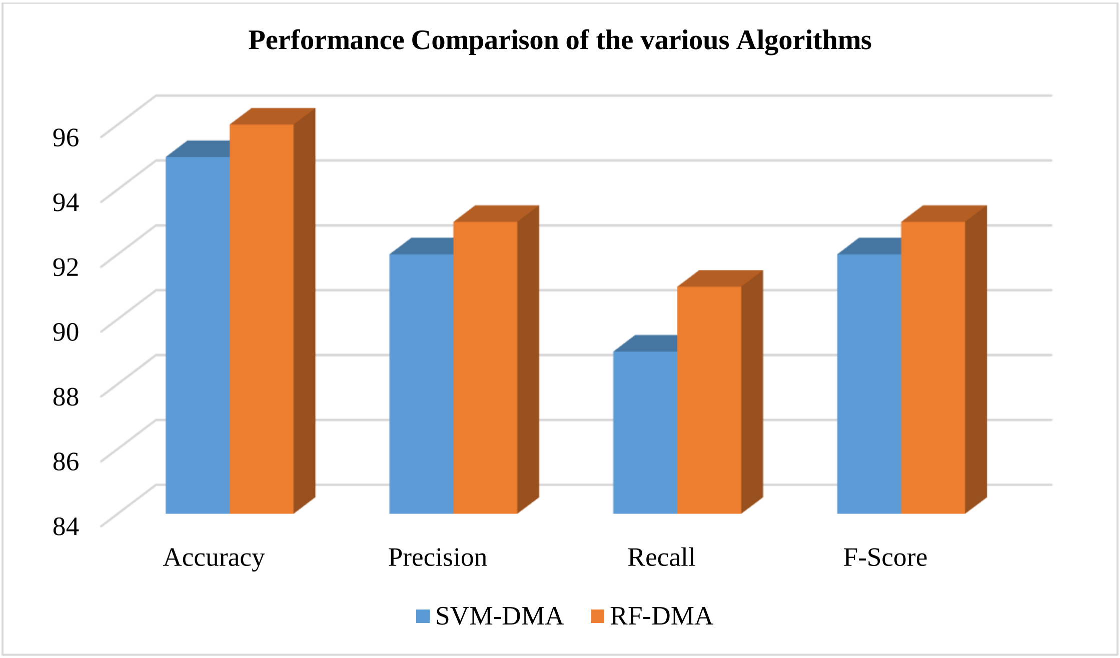

Fig. 9. Represents the comparison of SVM-DMA and RF-DMA

Table 3 displays the assessment of the two supervised learning models using the confusion matrix. The four file extension types that are used for categorization are.exe,.dll,.api, and any other file type. 950 is the representation of the confusion matrix for the number of flow occurrences. The performance of many supervised learning methods for data categorization is shown in Figure 3 using accuracy and error rate. The suggested approach has a very low error rate and up to 96% accuracy. The characteristics of our work will include additional approaches and the ability to work with intricate structures since POX Controller is skilled in doing so because malware analysis is a lengthy process. We use Machine Learning methods in our approach, and Pandas is used to present the findings. We have compared our results with a few of the existing methods that are already in use in order to better understand the outcomes.

5. Conclusion

In the SDN context, a supervised learning model for data traffic categorization is suggested. SVM, and Random Forest, are the two models employed. The combined strength of SVM and SDN along with the advantages of "selective logging" to provide a secure architecture.The features are successfully extracted and the feature vector has been generated. The suggested SVM-DMA is incorporated into SDN, and the performance has been measured in terms of Accuracy, Precision, Recall, and F-Score. The results of SVM-DMA are compared with the Existing SVM method. However it requires more hyperparamters to tune. Therefore fewer hyperparameters are required by splitting the data and the majority of the features have been consolidated in RF-DMA based on the voting. The combined strength of RF and SDN along with the advantages of "bootstrapping" to provide a secure architecture. The features are successfully extracted and placed in the Decision tree and the Prediction is computed based on the voting and the majority of the features obtained. The suggested RF-DMA are measured in the terms of Accuracy, Precision, Recall and F-Score.

Competing interest On behalf of all authors, the corresponding author states that there is no conflicts of interest.

Funding The authors did not receive support from any organization for the submitted work.

Ethical Approval This article does not contain any studies with participants or animals performed by any of the authors.

Author Contribution All Authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by VasantharajKarunakaran. Both the Authors VasantharajKarunakaran and Angelina Geetha read and approved the final Manuscript.

Data Availability The data has been taken from public site Kaggle as said above

Informed Consent Obtained from all individual participants included in this study

References

1. Ori Or-Meir, NirNissim, Yuval Elovici, and LiorRokach. 2019. Dynamic Malware Analysis in the Modern Era—A State of the Art Survey. ACM Comput. Surv. 52, 5, Article 88 (September 2020), 48 pages. https://doi.org/10.1145/3329786
2. PynbianglutHadem, Dilip Kumar Saikia, SoumenMoulik, An SDN-based Intrusion Detection System using SVM with Selective Logging for IP Traceback, Computer Networks, Volume 191, 2021, 108015, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2021.108015.
3. Shu, Zhaogang& Wan, Jiafu& Lin, Jiaxiang& Wang, Shiyong& Li, Di & Rho, Seungmin& Yang, Changcai. (2016). Traffic Engineering in Software-Defined Networking: Measurement and Management. IEEE Access. 4. 1-1. 10.1109/ACCESS.2016.2582748.
4. Wang C, Ni H, Liu L. An Enhanced Message Distribution Mechanism for Northbound Interfaces in the SDN Environment. *Applied Sciences*. 2021; 11(10):4346. https://doi.org/10.3390/app11104346
5. Wazirali R, Ahmad R, Alhiyari S. SDN-OpenFlow Topology Discovery: An Overview of Performance Issues. *Applied Sciences*. 2021; 11(15):6999. https://doi.org/10.3390/app11156999
6. Hao, J., &Ho, T. K. (2019). Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language. *Journal of Educational and Behavioral Statistics*, *44*(3), 348-361. https://doi.org/10.3102/1076998619832248 2935453.
7. Raschka, S.; Patterson, J.; Nolet, C. Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence. *Information* 2020, *11*, 193. https://doi.org/10.3390/info11040193
8. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L. and Lopez, A., 2020. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, *408*, pp.189-215.
9. Mishra, A., Gupta, N. & Gupta, B.B. Defence mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. *TelecommunSyst* 77,47–62(2021). https://doi.org/10.1007/s11235-020-00747-w
10. Xiang, Z., &Seeling, P. (2020). Mininet: An instant virtual network on your computer. In *Computing in Communication Networks* (pp. 219-230). Academic Press.

11. Noman, H. M., &Jasim, M. N. (2020, July). Pox controller and open flow performance evaluation in software defined networks (sdn) using mininet emulator. In *IOP conference series: materials science and engineering* (Vol. 881, No. 1, p. 012102). IOP Publishing.

12. Gupta, N., Maashi, M. S., Tanwar, S., Badotra, S., Aljebreen, M., &Bharany, S. (2022). A comparative study of software defined networking controllers using mininet. *Electronics*, *11*(17), 2715.

13. Pisner, Derek A., and David M. Schnyer. "Support vector machine." In *Machine learning*, pp. 101-121. Academic Press, 2020.

14. Jun, Zhao. "The development and application of support vector machine." In *Journal of Physics: Conference Series*, vol. 1748, no. 5, p. 052006. IOP Publishing, 2021.

15. Qiumei Cheng, Chunming Wu, Haifeng Zhou, Dezhang Kong, Dong Zhang, Junchi Xing, Wei Ruan, Machine learning based malicious payload identification in software-defined networking, Journal of Network and Computer Applications, Volume 192, 2021, 103186, ISSN 1084-8045,https://doi.org/10.1016/j.jnca.2021.103186.

16. H. Kim, N. Feamster, Improving network management with software defined networking, IEEE Commun. Mag. 51 (2) (2013) 114–119, http://dx.doi.org/10.1109/MCOM.2013.6461195.

17. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: Enabling innovation in campus networks, SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74, http://dx.doi.org/10.1145/1355734.1355746.

18. S. Savage, D. Wetherall, A. Karlin, T. Anderson, Practical network support for IP traceback, SIGCOMM Comput. Commun. Rev. 30 (4) (2000) 295–306, http://dx.doi.org/10.1145/347057.347560.

19. Negandhi, P., Trivedi, Y., &Mangrulkar, R. (2019). Intrusion detection system using random forest on the NSL-KDD dataset. In *Emerging Research in Computing, Information, Communication and Applications: ERCICA 2018, Volume 2* (pp. 519-531). Springer Singapore.

20. Chen, &Luo, Yu-Feng & Wen, Xinyu& Zhang, Yiyi& Yin, & Wu, Yingdan& Yao,. (2019). Pre-evacuation Time Estimation Based Emergency Evacuation Simulation in Urban Residential Communities. International Journal of Environmental Research and Public Health. 16. 4599. 10.3390/ijerph16234599.

21. B. Isong, R. R. S. Molose, A. M. Abu-Mahfouz and N. Dladlu, "Comprehensive Review of SDN Controller Placement Strategies," in *IEEE Access*, vol. 8, pp. 170070-170092, 2020, doi: 10.1109/ACCESS.2020.3023974.

22. MaxatAkbanov, Vassilios G. Vassilakis, Michael D. Logothetis, Ransomware detection and mitigation using software-defined networking: The case of WannaCry, Computers & Electrical Engineering, Volume 76, 2019, Pages 111-121, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2019.03.012

23. Rouka, Elpida&Birkinshaw, Celyn&Vassilakis, Vassilios. (2020). SDN-based Malware Detection and Mitigation: The Case of ExPetrRansomware. 10.1109/ICIoT48696.2020.9089514.

24. S. Jamalpur, Y. S. Navya, P. Raja, G. Tagore and G. R. K. Rao, "Dynamic Malware Analysis Using Cuckoo Sandbox," 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018, pp. 1056-1060, doi: 10.1109/ICICCT.2018.8473346.

25. McDole, A., Abdelsalam, M., Gupta, M., Mittal, S. (2020). Analyzing CNN Based Behavioural Malware Detection Techniques on Cloud IaaS. In: Zhang, Q., Wang, Y., Zhang, LJ. (eds) Cloud Computing – CLOUD 2020. CLOUD 2020. Lecture Notes in Computer Science(), vol 12403. Springer, Cham. https://doi.org/10.1007/978-3-030-59635-4_5

26. W. -J. Eom, Y. -J. Song, C. -H. Park, J. -K. Kim, G. -H. Kim and Y. -Z. Cho, "Network Traffic Classification Using Ensemble Learning in Software-Defined Networks," 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2021, pp. 089-092,
27. J. M. Ceron, C. B. Margi and L. Z. Granville, "MARS: An SDN-based malware analysis solution," *2016 IEEE Symposium on Computers and Communication (ISCC)*, Messina, Italy, 2016, pp. 525-530, doi: 10.1109/ISCC.2016.7543792.
28. Damodaran, A., Troia, F.D., Visaggio, C.A. *et al.* A comparison of static, dynamic, and hybrid analysis for malware detection. *J ComputVirol Hack Tech* 13, 1–12 (2017). https://doi.org/10.1007/s11416-015-0261-z
29. Ce Li, Zijun Cheng, He Zhu, Leiqi Wang, QiujianLv, Yan Wang, Ning Li, Degang Sun, DMalNet: Dynamic malware analysis based on API feature engineering and graph learning,Computers & Security, Volume 122, 2022, 102872, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2022.102872.
30. Akhtar, M.S.; Feng, T Malware Analysis and Detection Using Machine Learning Algorithms. Symmetry 2022, 14, 2304. https://doi.org/10.3390/sym14112304
31. Dutta, N., Jadav, N., Tanwar, S., Sarma, H.K.D., Pricop, E. (2022). Introduction to Malware Analysis. In: Cyber Security: Issues and Current Trends. Studies in Computational Intelligence, vol 995. Springer, Singapore. https://doi.org/10.1007/978-981-16-6597-4_7
32. Leon, R.S., Kiperberg, M., Leon Zabag, A. *et al.* Hypervisor-assisted dynamic malware analysis. *Cybersecur* 4, 19 (2021). https://doi.org/10.1186/s42400-021-00083-9
33. Amanowicz, M.; Jankowski, D. Detection and Classification of Malicious Flows in Software-Defined Networks Using Data Mining Techniques. Sensors 2021, 21, 2972. https://doi.org/10.3390/s21092972
34. Ahmed, N.; Ngadi, A.b.; Sharif, J.M.; Hussain, S.; Uddin, M.; Rathore, M.S.; Iqbal, J.; Abdelhaq, M.; Alsaqour, R.; Ullah, S.S.; et al. Network Threat Detection Using Machine/Deep Learning in SDN-Based Platforms: A Comprehensive Analysis of State-of-the-Art Solutions, Discussion, Challenges, and Future Research Direction. Sensors 2022, 22, 7896. https://doi.org/10.3390/ s22207896.
35. Hamzah Ahmed Faraj Al-Rubaye,Machine Learning and Network Security, Vol. 20, S3 (2024): Nanofabricated Materials for Optical Communication and Intelligent Manufacturing,
36. Wang, Tao & Li, Jingcong. (2019). An improved support vector machine and its application in P2P lending personal credit scoring. IOP Conference Series: Materials Science and Engineering. 490. 062041. 10.1088/1757-899X/490/6/062041.
37. MyoMyintOo, SinchaiKamolphiwong, ThossapornKamolphiwong, SangsureeVasupongayya, "Advanced Support Vector Machine- (ASVM-) Based Detection for Distributed Denial of Service (DDoS) Attack on Software Defined Networking (SDN)", *Journal of Computer Networks and Communications*, vol. 2019, Article ID 8012568, 12 pages, 2019. https://doi.org/10.1155/2019/8012568
38. Kurshid, Bimer, et al. "The Potential of Ultra-Wideband Printed Rectangular-Based Monopole Antennas." *National Journal of Antennas and Propagation* 5.2 (2023): 14-20.
39. K.-P. Lin, M.-S. Chen, Efficient kernel approximation for large-scale support vector machine classification, in: Proceedings of the 2011 SIAM International Conference on Data Mining, 2011, pp. 211–222, http://dx.doi.org/10.1137/1.9781611972818.19,
40. R. Vijayanand, D. Devaraj, B. Kannapiran, Support vector machine based intrusion detection system with reduced input features for advanced metering infrastructure of smart grid, in: 2017

4th International Conference on Advanced Computing and Communication Systems (ICACCS), 2017, pp. 1–7, http://dx.doi.org/10.1109/ICACCS.2017.8014590.

41. R. Chen, K. Cheng, Y. Chen, C. Hsieh, Using rough set and support vector machine for network intrusion detection system, in: 2009 First Asian Conference on Intelligent Information and Database Systems, 2009, pp. 465–470, http://dx.doi.org/10.1109/ACIIDS.2009.59.
42. B.S. Bhati, C.S. Rai, Analysis of support vector machine-based intrusion detection techniques, Arab. J. Sci. Eng. 45 (4) (2020) 2371–2383, http://dx.doi.org/10.1007/s13369-019-03970-z.
43. Tang Y, Gu L, Wang L. Deep Stacking Network for Intrusion Detection. Sensors (Basel). 2021 Dec 22;22(1):25. doi: 10.3390/s22010025. PMID: 35009568; PMCID: PMC8747112.
44. Cabarkapa, D., &Rancic, D. (2021). Performance Analysis of Ryu-POX Controller in Different Tree-Based SDN Topologies. *Advances in Electrical & Computer Engineering*, *21*(3).
45. Ban Mohammed Khammas, Ransomware Detection using Random Forest Technique, ICT Express, Volume 6, Issue 4, 2020, Pages 325-331, ISSN 2405-9595, https://doi.org/10.1016/j.icte.2020.11.001, http://archive.ics.uci.edu/ml/datasets/kdd+cup+1999+data.
46. Tavallaee, M., et al. (2009). A detailed analysis of the KDD CUP 99 data set. In Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009.
47. C. Ferri, J. Hernandez-Orallo, R. Modroiu, An experimental comparison of performance measures for classification, Pattern Recognit. Lett. 30 (1) (2009) 27–38,http://dx.doi.org/10.1016/j.patrec.2008.08.010, http://www.sciencedirect.com/science/article/pii/S0167865508002687.
48. Karunakaran, V. ., &Geetha, A. . (2023). Performing Dynamic Malware Analysis in Software Defined Network using LSTM Technique. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(2s), 411–419. https://ijisae.org/index.php/IJISAE/article/view/3641