Intrusion Detection Framework for DDOS and MITM Security Attacks in Software Defined Network

Carmen Villalobos Garcia
Computer Science Department
Cleveland State University
Cleveland, Ohio, United States
c.villalobosgarcia@vikes.csuohio.edu

Sathish Kumar Ph.D
Electrical Engineering and Computer Science Department
Cleveland State University
Cleveland, Ohio, United States
s.kumar13@csuohio.edu

Abstract - Software Defined Networking (SDN) has transformed our perception on the way networks are managed and deployed in an effective and efficient manner. However, it has also presented new challenges in the aspect of security. In this article, we present a comprehensive intrusion detection framework designed to address security attacks in SDN environments, with a focus on Distributed Denial of Service (DDoS) attacks and Man-in-the-Middle (MITM) attacks. Our suggested framework uses the full potential of SDN's capabilities to monitor, analyze and detect the real-time network traffic, including DDoS and MITM attacks. Additionally, it utilizes the gathered data to create a DDoS dataset, which is employed for training, testing and prediction through machine learning algorithms (ML). The primary objective of this study is to offer an evaluation of the accuracy and an effective method for detecting DDoS and MITM attacks, with the utilization of Mininet and the Ryu controller to simulate the network environment. After we applied ML techniques to detect DDoS attacks, we concluded that these algorithms are efficient for the detection of the attack. There are algorithms that have better accuracy than the others with an overall detection average of 63%.

Keywords – Software Defined Networking, Machine Learning, Distributed Denial of Service, Man-in-the-Middle.

I. INTRODUCTION

The emergence of SDN represents a shifting paradigm aiming to replace the traditional networking by separating the network intelligence (control plane) from the infrastructure layer (data plane). With this implementation of the control plane and data plane, the network forwarding devices such as switches, become simple forwarding devices. Moreover, the intelligence of the network is implemented in the centralized controller which is often referred as the brain of the network. SDN offers a variety of benefits that promise to increase the flexibility, efficiency, and scalability in network operations, such as the dynamic creation of network flow rules. However, alongside these advantages, SDN brings a set of vulnerabilities. One of which is a single point of failure. In the event that the controller is either compromised or experiences downtime causing the entire network become susceptible to disruption.

The significant rise of DDoS attacks is a matter of primary concern. These malicious cyberattacks are orchestrated to severely disrupt the operations of a computer network or service by inundating it with an excessive volume of illegitimate traffic. Within the scale of DDoS attack types, our focus will be based on Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP) and Transmission Control Protocol Synchronize (TCP-SYN) Flood.

MITM is a type of cyberattack where an attacker intercepts and alters the communication between two parties without their knowledge. The attacker utilizes ARP poisoning, which sends falsified Address Resolution Protocol (ARP) messages on a local network. These messages are designed to associate the attacker's MAC address with the IP address of a legitimate device, causing network traffic intended for that device to be redirected to the attacker.

The objective of this research is to study the programmability and flexibility of SDN in order to detect DDoS and MITM attacks.

This paper contributes to the literature by proposing a comprehensive and multifaceted approach to intrusion detection of DDoS attacks in SDN. The combination of detection and classification not only focuses on detecting DDoS attacks in SDN using ML algorithms but also explores into classifying different types of DDoS attacks, including ICMP, UDP, and TCP-SYN floods. This emphasis on both detection and classification provides a more thorough understanding of the attack

landscape and contributes to a more comprehensive defense strategy. In addition, we normalized the dataset and use dimensionality reduction techniques as PCA and FastICA, to identify important parameters that can be used for attack detection. This contributes to enhanced accuracy by reducing noise and extracting critical features, setting the work apart as more rigorous and detailed. The utilization of real-time traffic captures for evaluation ensuring that the research is validated in practical scenarios.

In addition, our work involves practical implementation using tools like Ettercap and ARP poisoning to simulate the MITM attack. This practical approach which connects theory and real world scenarios, makes our work more applicable. The method that we have proposed for MITM attack detection which uses the tuple of IP and MAC addresses is a distinct and effective method. MITM attack is a serious security threat that can have devastating consequences, for example the loss of sensitive information such as login credentials, passwords and personal data.

II. BACKGROUND AND RELATED WORK

The fusion of machine learning algorithms and SDN for detecting DDoS attacks represents an exciting and evolving area of research. A typical SDN enviroment consists of a controller, switches and hosts. There exists a communication channel between the controller and the switches. The communication is defined by the OpenFlow protocol. This protocol enables the controller (Ryu in our case) to communicate with the switches, providing routing instructions, requesting port and flow statistics, and managing newly arriving packets. The OpenFlow objective was to break the limitations on the traditionally network communication by providing more flexibility and control over the network, as shown in Figure 1.

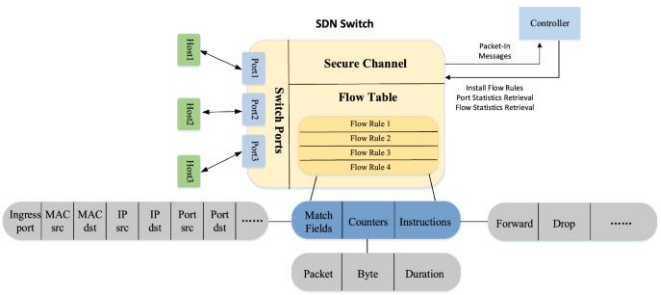


Figure 1. SDN environment.

When a switch receives a new packet from a flow, it initiates a packet-in request to the controller, asking it to establish the appropriate matching rules. Subsequently, the controller manages the global network traffic, allowing the remaining packets of the flow to be processed directly by the established rules, as shown on Figure 2.

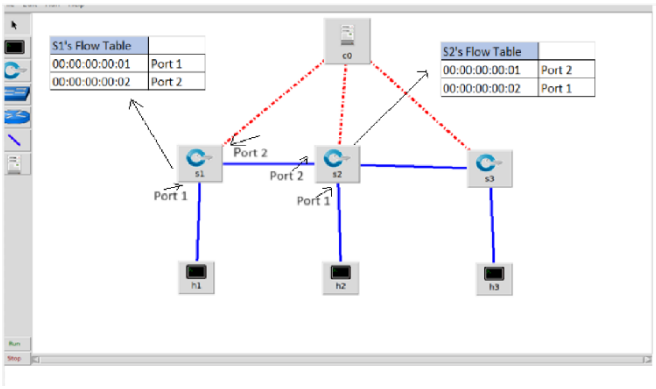


Figure 2. Packet forwarding.

The related work has a growing interest in the unique capabilities of SDN to enhance the accuracy, efficiency, and timeliness of DDoS intrusion detection. Various studies have investigated the identification of relevant features, such as packet rates, flow durations, and payload sizes, to optimize the performance of ML models. Some studies have applied deep learning techniques, such as Convolutional Neural Networks (CNNs) [5] and Recurrent Neural Networks (RNNs), for DDoS detection. Other studies are based on the evaluation of ML based on Intrusion Detection Systems (IDS) [7] for DDoS attacks that involves benchmarking against traditional methods.

In the context of MITM attack, there are many studies that analyze the pattern recognitions techniques to identify the attack. Other studies have explored the inconsistencies of the controller. There are several works done regarding the MITM detection on Multi-SDN controller [18] where they found some vulnerabilities on the different layers of the SDN. Additionally, researchers propose strategies to prevent unauthorized devices from gaining a foothold and launching MITM attacks. One technique to perform a MITM attack is to use ARP-poisoning. ARP is a protocol used to map IP addresses to MAC addresses. When a device on a network needs to find the MAC address associated with a particular IP address, it sends out a broadcast message, essentially asking, "Who has this IP address, and what's your MAC address?" The device that holds that IP address responds with its MAC address. ARP-poisoning is done by constantly sending out fake ARP reply packets to the devices an attacker wishes to attack. This means that by the time a legitimate ARP broadcast is sent to discover the MAC address, the attacker has already provided a fraudulent response. In essence, it tricks the network into thinking that the attacker's MAC address is associated with the IP address in question,

diverting traffic to the attacker's machine and enabling them to intercept, snoop, or manipulate network communications.

III. SIMULATION ENVIROMENT AND METHODOLOGY

We used Mininet to create and study the behavior of our SDN network. Mininet is a network emulation software that generates a virtual network environment composed of hosts, switches, controllers, and links. The MiniEdit platform provides a suitable graphical interface for experimenting with SDN and OpenFlow concepts. Mininet supports Python API to create and test networks. After examining various SDN controllers, we landed on the Ryu controller [2]. This open-source software framework is built for creating and deploying applications in SDN environments. Ryu controller empowers the development of the network control logic, boasting user-friendliness, customization capabilities, and advanced development processes. Once familiarized with Mininet and Ryu, we proceeded to generate our dataset for DDoS attacks. To generate benign traffic, we employed tools like *iperf* to create UDP traffic, *ping* for ICMP, and a *web-server* for TCP-SYN. To generate illegitimate traffic, we employed the *hping3* tool to simulate DDoS traffic attack such as ICMP, UDP, and TCP-SYN floods. Using the tools previously mentioned, the benign and illegitimate traffic is collected by capturing the traffic with Ryu controller in a csv file.

In our study, we have utilized the powerful tools provided by the scikit-learn library to develop and implement our ML models to detect DDos attacks in SDN environments. Our detection approach involves the utilization of ML algorithms. These algorithms are Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Naive Bayes, Decision Tree and Random Forest. Logistic Regression estimates the probability of an event occurring. This model is based on the logistic function and estimates the relationship between input features and outcome probability. KNN is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. SVM identifies a hyperplane that maximizes the separation between classes. Naive Bayes is a probabilistic classifier using Bayes' theorem. It seeks to model the distribution of inputs of a given class or category. Decision Trees are hierarchical structures that split data based on feature values. Each internal node represents a decision, and each leaf node represents a class or value prediction. Random forest combines the output of multiple decision trees to reach a single result. By aggregating predictions from multiple trees trained on random subsets of data and features, it improves accuracy and reduces overfitting.

The process of training and testing ML algorithm models involves generating prediction models and evaluating their performance using a confusion matrix to calculate the accuracy. We can observe the classification process of DDos attack on Figure 3.

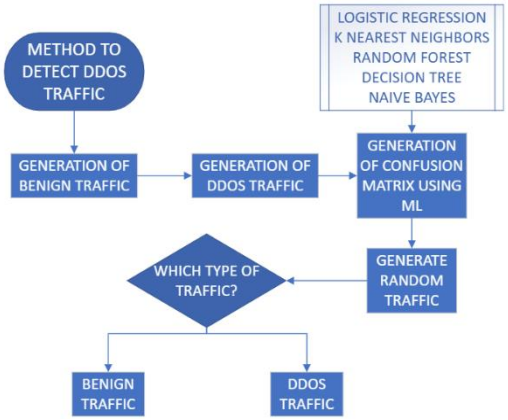


Figure 3. Classification Process Model of DDoS attack.

In our experiment one of the objectives is to identify which ML algorithm has the best performance on precision, accuracy, recall and F1 score. In the data preprocessing phase, we incorporate the utilization of data cleaning, normalization, feature extraction as Principal Component Analysis (PCA), and Fast Independent Component Analysis (FastICA).

MITM threat model implementation the Address Resolution Protocol (ARP), a protocol that correlates the IP address with the MAC address of a network device. When a host is trying to communicate with a destination host, in the cases where the MAC address of the destination host is unknown, we need a mechanism to discover the MAC address associated with the IP of the destination host. ARP protocol is used to discover the MAC address. This action triggers an ARP request, enabling us to retrieve the MAC address of the intended destination. Furthermore, the design of the ARP protocol allows that an ARP can be sent to a destination at any time, specifying the MAC associated with an IP. This action causes an update in the ARP's table, refreshing the IP-MAC entry. This characteristic is exploited by ARP Poisoning attacks to manipulate the MAC associated with an IP, substituting it with the attacker's MAC address.

Ettercap has been used to carry out ARP poisoning attacks in MITM. In the context of ARP poisoning, Ettercap forges ARP messages within the network to associate fraudulent MAC addresses with legitimate IP addresses, enabling the interception and redirection of network traffic. Ettercap intercepts and potentially modifies communication between two parties, all without their awareness of the attacker's presence in-between. In addition, once the attack has begun, the attack's host begins to receive the data stream from the original source system and subsequently forwards the information to the original destination host, therefore the

attack goes undetected. The classification process of MITM attack is shown on Figure 4. The controller will capture the incoming traffic, if the IP and MAC addresses are already stored, it will proceed to compare the stored information with this new tuple and check if they have been poisoned. If the tuples compared are not the same, we have detected a MITM.

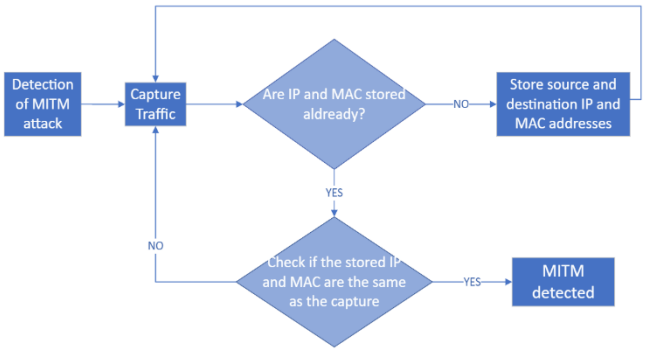
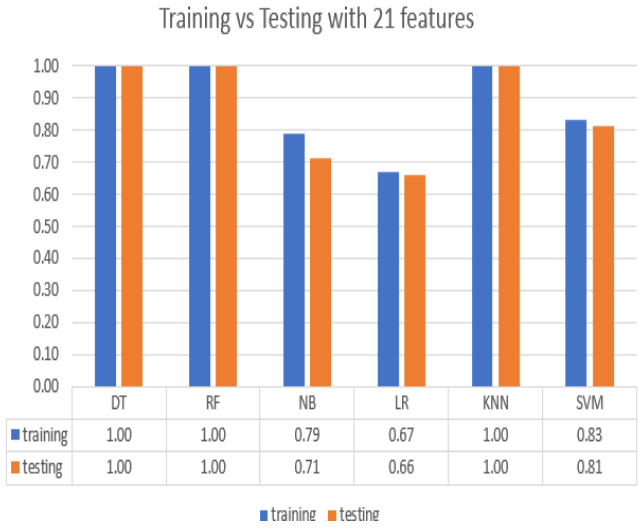


Figure 4. Classification Process Model of MITM attack

IV.EXPERIMENTATION AND RESULTS DISCUSSION

A. DDoS Attack

To evaluate our approach, we have done several experimentations to compare the accuracy results of different models. For the first test, without the use of normalization or PCA, we used 21 features and partitioned the collected data into two separate files using a 70:30 ratio. We trained the models with 1,867,266 rows for training and 800,000 rows for testing. Then, we obtained the accuracy results for all the models, as shown in Figure 5.



	DT	RF	NB	LR	KNN	SVM
training	1.00	1.00	0.79	0.67	1.00	0.83
testing	1.00	1.00	0.71	0.66	1.00	0.81

Figure 5. Accuracy of ML models using 21 features.

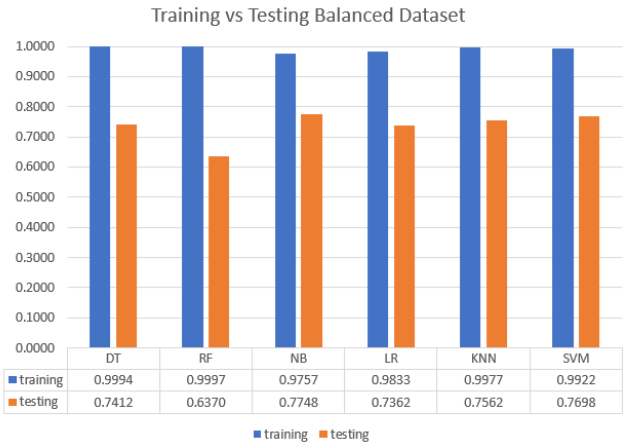
We can observe from Figure 4 how the accuracy of all models is remarkably high when using all the features in the dataset. The models that exhibit the best performance are Decision Tree, Random Forest, and K-Nearest Neighbors.

For the next experiment, we employed normalization and PCA. We normalized the dataset within a range of 0 to 1. In PCA, the features are the original variables that describe the observations, while the principal components are the new variables calculated from these original features to summarize the information more efficiently. In our case we selected 21 features of our dataset and 12 components on the PCA. When using PCA, to find the optimal number of components, we calculated the sum of variances using the total number of features in our dataset, which was 21. From there, we needed to compare the dataset's sum of variances while reducing the number of components with little to no alteration of the dataset's sum of variances. The optimal number of components to use in our dataset was 12. The sum of variances has been reduced to 1% from the original. We can see the process just explained on Figure 6.

Components	Sum of Variances
21	1.262862
18	1.262862
15	1.262804
13	1.262262
12	1.259926
10	1.211322
8	1.136906

Figure 6. Results of sum of variances using different components.

After we obtained the optimal number of components to use on the PCA, we wanted to compare the accuracy of the ML models on balanced and unbalanced datasets. To begin with the balanced dataset experimentation, we trained all the models with 400,000



	DT	RF	NB	LR	KNN	SVM
training	0.9994	0.9997	0.9757	0.9833	0.9977	0.9922
testing	0.7412	0.6370	0.7748	0.7362	0.7562	0.7698

Figure 7. Accuracy of ML models on balanced dataset using 21 features with normalization and PCA with 12 components.

benign rows and 100,000 DDoS rows. For testing, we used 200,000 benign rows and 200,000 DDoS rows.

We can see how the accuracy of the models decreased by the use of PCA compared with Figure 4. Naive Bayes, KNN and Support Vector Machine have the best accuracy.

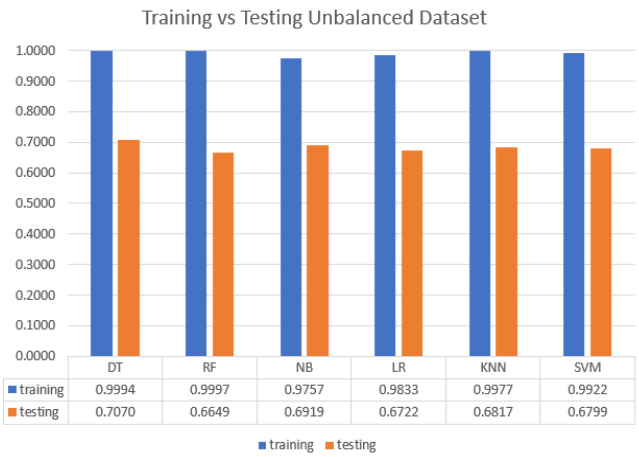


Figure 8. Accuracy of ML models on unbalanced dataset using 21 features with normalization and PCA with 12 components.

To continue, we trained the models as we did before, with 400,000 benign rows and 100,000 DDoS rows. However, for testing, we utilized an unbalanced dataset consisting of 200,000 benign rows and 100,000 malicious rows.

We can compare how the results on the unbalanced datasets are less accurate than on balanced datasets by looking at Figure 7 and 8. We can also compare the accuracy, precision, recall and f1-score on Figure 9.

Overall, the models present a better accuracy, precision, recall and F1-score on balanced datasets. We also tested the models with another feature extraction technique, FastICA, to compare the results with PCA as we can see in Figure 9.

			CONFUSION		MATRIX			MACRO AVG	
		TP	TN	FN	FP	ACCURACY	PRECISSION	RECALL	F1 SCORE
DT	Balanced	188341	108147	91853	11658	0.7412	0.9417	0.6722	0.7844
	Unbalanced	187567	24533	75467	12431	0.7070	0.9378	0.7131	0.8102
RF	Balanced	199999	54798	145202	0	0.6370	1.0000	0.5794	0.7337
	Unbalanced	198871	600	99400	1127	0.6649	0.9944	0.6667	0.7982
NB	Balanced	198340	111563	88437	1659	0.7748	0.9917	0.6916	0.8149
	Unbalanced	199998	7581	92419	0	0.6919	1.0000	0.6839	0.8123
LR	Balanced	183903	110577	89423	16096	0.7362	0.9195	0.6728	0.7771
	Unbalanced	171224	30436	69564	28774	0.6722	0.8561	0.7111	0.7769
KNN	Balanced	198565	103915	96085	1434	0.7562	0.9928	0.6739	0.8029
	Unbalanced	194952	9580	90420	5046	0.6818	0.9748	0.6832	0.8033
SVM	Balanced	199999	107902	92098	0	0.7698	1.0000	0.6847	0.8128
	Unbalanced	199999	3989	96011	8	0.6799	1.0000	0.6756	0.8064

Figure 9. Accuracy, Precision, Recall, F1-score of ML models on balanced and unbalanced datasets.

As shown in Figure 10, the accuracy varies depending on the feature extraction method that we use. It is highly important to be aware of the dataset and identify the best method according to it.

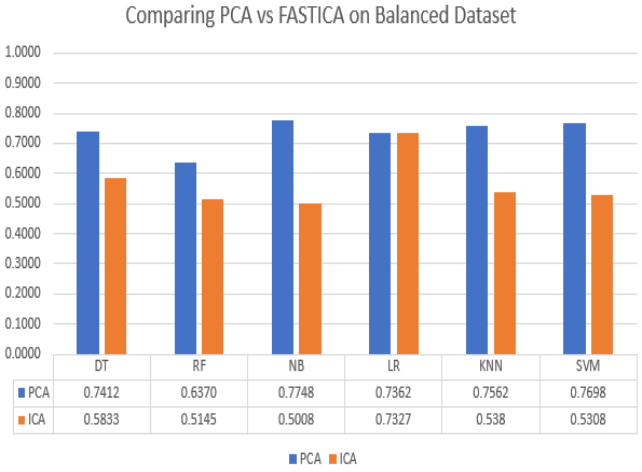


Figure 10. Comparing accuracy of ML models using PCA and FASTICA.

All the previous steps have been applied using techniques without any human observation. After conducting these tests, we examined the potential of the features in our dataset regarding DDoS attacks to verify their significance in detection. These characteristics provide us with key information of selecting appropriate features to detect the attack. The DDoS attack sends a large volume of traffic to the target to flood its bandwidth or processing capacity. Thus, by observing the distinctions between benign and illegitimate traffic, we were able to observe how these fields present significant differences. Therefore, we used the brute-force approach and selected these 6 features. After numerous tests, we concluded that accuracy improves using the following six features: IP source, packet count, packets per second, packets per nanosecond, bytes per second, and bytes per nanosecond.

In the next experimentation, we used the same balanced dataset from Figure 7 and select these 6 features on this dataset. We normalized the dataset and perform the test without the use of PCA. We compared these results with the testing results we obtained on Figure 11. Reducing the number of features significantly diminishes both training and detection execution times. This impact is noticeable not only in terms of CPU time but also in the efficient utilization of resources, including memory. This effect becomes particularly pronounced when dealing with larger datasets.

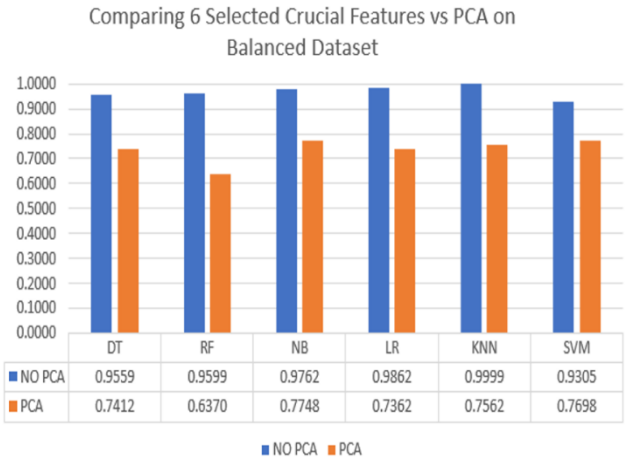


Figure 11. Comparing the accuracy of the ML models using normalization with 6 crucial features versus the accuracy using normalization and PCA with 21 features and 12 components.

After exploring the fundamentals of DDoS detection, we conducted experiments to classify the type of attack by verifying the protocol that we used. To achieve this, we constructed a dataset for model training, comprising 400,000 rows of benign traffic and 300,000 rows of DDoS traffic. The DDoS traffic subset comprised 100,000 rows of ICMP attacks, 100,000 rows of UDP attacks, and 100,000 rows of TCP-SYN attacks. For testing, we used 150,000 rows each of benign and DDoS traffic. The DDoS traffic subset included 50,000 rows of ICMP attacks, 50,000 rows of UDP attacks, and 50,000 rows of TCP-SYN attacks. We performed different experiments to determine which ML model can classify the different types of DDoS attack.

DECISION TREE

Features	6 – Hand Picked	21	21	21
Normalization	Yes	Yes	Yes	Yes
PCA	Yes 3 components	Yes 12 components	Yes 18 components	No
UDP accuracy	1.0	0.25272	0.49852	1.0
ICMP accuracy	1.0	0.0	0.0	1.0
TCP accuracy	0.99999	0.0025	0.0141	1.0

RANDOM FOREST

Features	6 – Hand Picked	21	21	21
Normalization	Yes	Yes	Yes	Yes
PCA	Yes 3 components	Yes 12 components	Yes 18 components	No
UDP accuracy	0.00062	0.46258	0.57196	1.0
ICMP accuracy	1.0	0.0	0.0	1.0
TCP accuracy	1.0	0.00078	0.01116	1.0

NAIVE BAYES

Features	6 – Hand Picked	21	21	21
Normalization	Yes	Yes	Yes	Yes
PCA	Yes 3 components	Yes 12 components	Yes 18 components	No
UDP accuracy	0.14338	0.79954	0.79954	1.0
ICMP accuracy	0.16352	1.0	1.0	0.99998
TCP accuracy	0.15882	0.4003	0.4003	1.0

LOGISTIC REGRESSION

Features	6 – Hand Picked	21	21	21
Normalization	Yes	Yes	Yes	Yes
PCA	Yes 3 components	Yes 12 components	Yes 18 components	No
UDP accuracy	1.0	0.7367	0.7586	1.0
ICMP accuracy	0.0	0.88404	0.86374	1.0
TCP accuracy	1.0	0.76414	0.76838	1.0

K-NEAREST NEIGHBOR

Features	6 – Hand Picked	21	21	21
Normalization	Yes	Yes	Yes	Yes
PCA	Yes 3 components	Yes 12 components	Yes 18 components	No
UDP accuracy	1.0	0.83486	0.49378	1.0
ICMP accuracy	0.99996	1.0	1.0	1.0
TCP accuracy	0.99988	0.83746	0.53126	0.12288

Figure 12. Accuracy on different methods when classifying different types of DDoS attacks.

As shown in Figure 12, we can see how all the models perform effectively when utilizing 21 features and no PCA. Decision Tree and KNN have high accuracy when using the 6 hand picked parameters with PCA of 3 components. Across all the tests, the models, Logistic Regression and KNN are the ones with the highest accuracy on all the tests and are very constant.

B. MITM attack

For the MITM detection process, the controller takes on the role of capturing incoming traffic, extracting the ARP packet containing both the source and destination IP and MAC addresses. It's not just ARP packets; when an IPv4 packet arrives, we also extract the Ethernet frame and retrieve the corresponding MAC address. This process allows us to evaluate whether we already have the MAC address or to compare it in order to determine if it is a MITM attack. This tuple of information is then stored, if there was not a tuple previously stored as we can see on Figure 13. As new incoming traffic arrives, we perform a comparison between the newly formed tuple and the stored one. If the two tuples differ, it indicates the occurrence of a MITM attack.

```
ICMP
MAC Src: 00:00:00:00:00:01 MAC Dst: 00:00:00:00:00:02
IP Src: 10.0.0.1 IP Dst: 10.0.0.2
----------------------------
----------------------------
ICMP
MAC Src: 00:00:00:00:00:02 MAC Dst: 00:00:00:00:00:01
IP Src: 10.0.0.2 IP Dst: 10.0.0.1
----------------------------
----------------------------
ICMP
MAC Src: 00:00:00:00:00:01 MAC Dst: 00:00:00:00:00:02
IP Src: 10.0.0.1 IP Dst: 10.0.0.2
----------------------------
----------------------------
ICMP
MAC Src: 00:00:00:00:00:02 MAC Dst: 00:00:00:00:00:01
IP Src: 10.0.0.2 IP Dst: 10.0.0.1
----------------------------
----------------------------
```

Figure 13. IP and MAC addresses stored after a ICMP

For experimentation, we set up the most basic network configuration with three hosts, a switch, and a controller. A *pingall* command is executed to establish MAC and IP addresses associations. Following this, the corresponding ARP requests will have been generated to map MACs to hosts' IPs. For instance, in Host 1, we have the MAC and IP of Hosts 2 and 3 respectively. Similarly, Host 2 holds the MAC and IP of Hosts 1 and 3, while Host 3 contains the MAC and IP of Hosts 1 and 2. All IP-MAC associations are legitimate.

Utilizing the flexibility of the Ryu controller, we captured traffic and generated tables containing these legitimate IP-MAC associations. We then launched a MITM attack using Ettercap as shown in Figure 14, executing ARP poisoning. This attack leads to

fraudulent communication between Host 1 and Host 2; Host 1 holds the IP of Host 2 but the MAC of Host 3. As a result, the traffic passes through Host 3. To prevent Host 2 from becoming aware of the attack, Host 3 forwards the traffic to Host 2. A similar scenario unfolds in the communication between Host 2 and Host 1.

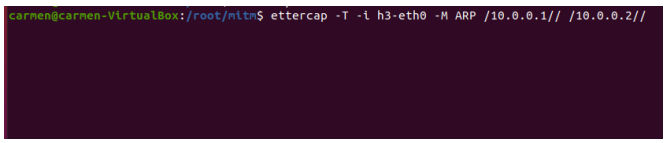


Figure 14. Launching the MITM attack on Host 3

For attack detection, since we initially stored legitimate IP-MAC tuples in a table, any incoming traffic, whether legitimate or part of a MITM attack, is compared with the previously stored tuple. If the captured tuple doesn't match the stored one, a MITM attack is detected, as shown on Figure 15.

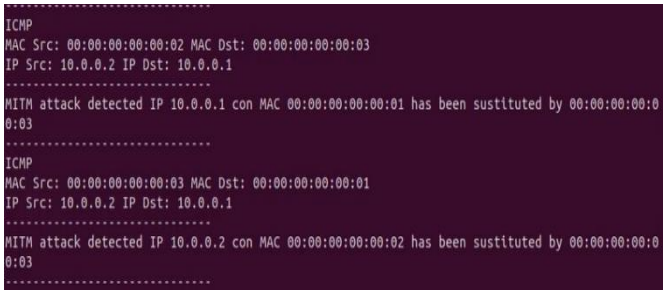


Figure 15. Detection of MITM attack.

Regarding attack mitigation, we can approach it with various methods. One straightforward approach is to immediately drop the packet upon detection. Once we have detected the mismatch in the IP-MAC tuple, the controller will intervene and instruct switches to discard the malicious packets. Preventing them from reaching their intended destination. Another mitigation strategy involves creating a network flow to block traffic to and from the attacking host, Host 3. By configuring rules in the network's switches and routers, the controller can direct traffic to be filtered or redirected. For instance, traffic originating from Host 3 can be blocked entirely, preventing it from interacting with other hosts within the network. In a SDN environment is easier to detect this MITM attacks since the information goes through the controller. In a regular network, if the data is within the broadcast domain it doesn't traverse through a central firewall making it challenging to detect.

V. CONCLUSION

In this paper we have generated a dataset to proceed with the exploration of different machine learning techniques for detecting and classifying DDoS attacks. The study has yielded valuable points regarding the effectiveness and potential implications of DDoS attack. The investigation reveals that the Random Forest algorithm stands out as a robust method for detecting DDoS attacks. However, when combined with Principal Component Analysis (PCA), the model's accuracy is negatively impacted. This suggests that the reduction of feature dimensions through PCA might not always be beneficial for DDoS detection, emphasizing the importance of feature selection and engineering. For classifying different types of DDoS attacks, the K-Nearest Neighbor (KNN) algorithm emerges as the most suitable choice. This implies that KNN can effectively differentiate between various attack patterns based on the selected features.

The accuracy of the models experiences a significant boost after we have meticulously curated a set of six key features encompassing network metrics such as IP source, packet and byte counts per second and nanosecond. This achievement, with an average accuracy of 96.81%, underscores the importance of thoughtful feature selection in enhancing the models' performance.

In our experiments, we observed how the balanced datasets contribute to improved accuracy in DDoS detection and classification. Conversely, when using unbalanced datasets, the models struggle due to the lack of attack instances in the training data. This leads to biased learning, resulting in lower accuracy compared to balanced datasets. This underscores the need to ensure that the training data accurately represents the diverse range of potential attacks, leading to better generalization to real scenarios.

In the context of SDN, the study highlights a critical observation – if a DDoS attack remains unchecked and unmitigated, the SDN controller becomes vulnerable to its effects. This underscores the importance of integrating effective detection and mitigation strategies within the SDN framework to protect the network against potential controller disruptions.

This experimental results for attack scenarios about DDoS underscores the significance of choosing the right ML techniques, feature selection methods, and dataset considerations when addressing DDoS attack detection and classification. The study's insights not only contribute to the advancement of cybersecurity strategies but also provide valuable guidance for designing robust defense mechanisms in dynamic network environments.

Furthermore, this paper distinguishes itself by offering a comprehensive, meticulously executed, and practically relevant approach to intrusion detection and classification of DDoS attacks in SDN using ML algorithms.

The MITM attack experimentation and results stand out for the unique practical implementation, theoretical analysis, specific attack tools, and focus on detection within SDN. By addressing a crucial aspect of security with a practical approach, contributing with valuable insights that connects theory and real-life application.

VI. FUTURE WORK

As we look ahead to future research directions, our focus will be on studying and exploring mitigation approaches of DDoS and MITM attacks in a SDN environment. Additionally, we plan to analyze and compare different SDN controllers, such as POX, OpenDaylight (ODL), Floodlight to evaluate our security approach. Evaluating the controllers' performance in terms of real-time attack detection, response speed, resource utilization, and adaptability would provide valuable insights for selecting the most suitable controller for specific network environments. In addition, we would like to focus our future research on generating diverse types of attack scenarios. Creating a broader spectrum of attack scenarios that would enable a more comprehensive evaluation of the effectiveness of our security mechanisms.

VII. REFERENCES

[1] H. Bisht, M. Patra and S. Kumar, “Detection and Localization of DDoS attack during Inter-Slice Handover in 5G Network Slicing”, 20th IEEE International Conference on Consumer Communications, January 2023

[2] Jeeva Chelladhurai, Pethuru Raj Chelliah and Sathish A.P. Kumar, “Securing Docker Containers from Denial of Service (DoS) Attacks” in 13th IEEE International Conference on Service Computing (IEEE SCC), pp. 856-859, 2016

[3] Zhang, J., et al. (2019). DDoS attack detection method based on support vector machine. Journal of Physics: Conference Series, 1331(4), 042089.

[4] K. Zhang y X. Qiu, "CMD: A Convincing Mechanism for MITM Detection in SDN,"IEEE International Conference on Consumer Electronics (ICCE), IEEE, 2018.

[5] Y. Cao, et al, "Detecting and Mitigating DDoS Attacks in SDN Using Spatial-Temporal Graph Convolutional Network" in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 06, pp. 3855-3872, 2022. doi: 10.1109/TDSC.2021.3108782

[6] B. Pande, G. Bhagat, S. Priya and H. Agrawal, "Detection and Mitigation of DDoS in SDN," in 2018 Eleventh International Conference on Contemporary Computing (IC3), Noida, India, 2018 pp. 1-3. doi: 10.1109/IC3.2018.8530551

[7] Kumari, K., Mrunalini, M. Detecting Denial of Service attacks using machine learning algorithms. J Big Data 9, 56 (2022). https://doi.org/10.1186/s40537-022-00616-0

[8] M. Azab and J. A. B. Fortes, "Towards Proactive SDN-controller Attack and Failure Resilience," Advanced Computing and Information Systems Laboratory, ECE, University of Florida, Gainesville, FL, USA, Tech. Rep. 7876169, 2017.

[9] T. Xu, D. Gao, P. Dong, H. Zhang, C. H. Foh and H. -C. Chao, "Defending Against New-Flow Attack in SDN-Based Internet of Things," in IEEE Access, vol. 5, pp. 3431-3443, 2017, doi: 10.1109/ACCESS.2017.2666270.

[10] M. S. Elsayed, N. -A. Le-Khac, S. Dev and A. D. Jurcut, "Machine-Learning Techniques for Detecting Attacks in SDN," 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT), Dalian, China, 2019, pp. 277-281, doi: 10.1109/ICCSNT47585.2019.8962519.

[11] Kim, H., et al. (2016). A machine learning approach for DDoS attack detection and packet classification. Journal of Information Processing Systems, 12(3), 464-475.

[12] P, Karthika & Karmel, A.. (2023). Simulation of SDN in mininet and detection of DDoS attack using machine learning. Bulletin of Electrical Engineering and Informatics. 12. 1797~1805. 10.11591/eei.v12i3.5232.

[13] K. Salah, N. Z. Bawany y J. A. Shamsi, "DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions," Systems Research Laboratory, FAST-National University of Computer and Emerging Sciences, Karachi, Pakistan; Electrical and Computer Engineering Department, Khalifa University of Science, Technology and Research, PO Box 573, Sharjah, UAE, February 2017

[14] Mohammad Z Masoud, Yousf Jaradat, and Ismael Jannoud. On preventing arp poisoning attack utilizing software defined network (sdn) paradigm. In Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on, pages 1–5. IEEE, 2015.

[15] PISK MITM. TLS MitM attack on OpenFlow switch using Ryu controller app. https://github.com/m-kostrzewa/PSIK-MitM, 2017. [Online; accessed june-2017].

[16] KaiZhangand XiaofengQiu,“CMD:AConvincing Mechanism for MITMDetectioninSDN,”IEEE International Conference on Consumer Electronics (ICCE), IEEE,2018.

[17] A. Sebbar, M. Boulmalf, M. D. Ech-Cherif El Kettani, y Y. Baddi, "Detection of MITM Attack in Multi-SDN Controller," ELIT-Internationale University of Rabat, Rabat, Morocco, Tech. Rep.

[18] N. Saritakumar, K. V. Anusuya, y B. Balasaraswathi, "Detection and Mitigation of MITM Attack in Software Defined Networks," Assistant Professor, ECE, PSG College of Technology, Coimbatore, India, Tech. Rep.

[19] Sebbar, M. Boulmalf, M. Dafir Ech-Cherif El Kettani and Y. Baddi, "Detection MITM Attack in Multi-SDN Controller," 2018 IEEE 5th International Congress on Information Science and Technology (CiSt), Marrakech, Morocco, 2018, pp. 583-587, doi: 10.1109/CIST.2018.8596479.

[20] H. Ma, H. Ding, Y. Yang, Z. Mi and M. Zhang, "SDN-Based ARP Attack Detection for Cloud Centers," 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), Beijing, China, 2015, pp. 1049-1054, doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.195.

[21] J. D'Orsaneo, M. Tummala, J. McEachen and B. Martin, "Analysis of Traffic Signals on an SDN for Detection and Classification of a Man-in-the-Middle Attack," 2018 12th International Conference on Signal Processing and Communication Systems (ICSPCS), Cairns, QLD, Australia, 2018, pp. 1-9, doi: 10.1109/ICSPCS.2018.8631762.