DDoS Attacks Detection and Mitigation in SDN using Machine Learning

Obaid Rahman
Department of Systems and Computer Eng.
Carleton University
Ottawa, Canada
obaidrahman@cmail.carleton.ca

Mohammad Ali Gauhar Quraishi
Department of Electrical and Computer Eng.
University of Ottawa
Ottawa, Canada
mqura082@uottawa.ca

Chung-Horng Lung
Department of Systems and Computer Eng.
Carleton University
Ottawa, Canada
chlung@sce.carleton.ca

***Abstract*—Software Defined Networking (SDN) is very popular due to the benefits it provides such as scalability, flexibility, monitoring, and ease of innovation. However, it needs to be properly protected from security threats. One major attack that plagues the SDN network is the distributed denial-of-service (DDoS) attack. There are several approaches to prevent the DDoS attack in an SDN network. We have evaluated a few machine learning techniques, i.e., J48, Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbors (K-NN), to detect and block the DDoS attack in an SDN network. The evaluation process involved training and selecting the best model for the proposed network and applying it in a mitigation and prevention script to detect and mitigate attacks. The results showed that J48 performs better than the other evaluated algorithms, especially in terms of training and testing time.**

Keywords—SDN, DDoS, Machine Learning, J48, Weka

I. INTRODUCTION

Software Defined Networking (SDN), has gained popularity these days due to the benefits it provides such as scalability, flexibility, monitoring, and ease of innovation [1]. It is an emerging networking paradigm that separates the core networks control logic (policies defined by the controller) from the underlying routing and switching elements that simply operate as packet forwarders [2].The data plane consists of the network elements such as switches (OpenFlow or simply OF switches) that are managed by the controller in the control plane [3]. The separation of the control plane and data plane plays a critical role in providing superior performance in large-scale, high-speed computing systems. It also provides a simplified network management since configuration and management are handled only through the controller [1][2]. To perform network upgrades and adjustments the administrator does not need to access and reconfigure thousands of devices individually in the network. They can simply enforce policy and network configuration requirements centrally in real-time via the controller [4].

The controller needs several core services to manage the data plane. It allows for its information to be exchanged with the services in the application layer to provide network functions such as routing, load balancing, intrusion detection [1] . All the services and applications that are implemented in the application layer are mapped to the whole network by a network operating system, installed on the controller providing a high level of network control, automation and optimization [1]. The applications use several Application Programming Interfaces (API's) such as Java API's to communicate locally with the controller or representational state transfer (REST) API for remote communication with the controller [3].

However, the very reason that puts SDN network in the forefront and makes it popular also exposes it to a variety of new security threats. One of these that has the most devastating effect on an SDN network is the distributed denial-of-service (DDoS) attack. DDoS can overwhelm the controller or the OpenFlow (OF) switch if the network is not properly protected. There are a wide variety of documentations on how to protect the SDN network from DDoS attacks. Intrusion detection systems (IDSs) are used in the network to sniff the packets and alert the administrator when a DDoS attack is detected. One such technology attracting researchers focuses on using machine learning to detect DDoS attacks. However, protecting the SDN network from threats is still an active research area. In this paper we focus on one such approach which aims to determine the most suitable machine learning algorithm to identify a DDoS attack in a live network and subsequently mitigate it.

The rest of the paper is organized as follows: Section II describes SDN network operations and threats to SDN planes. Section III introduces various types of DDoS attack and how their detection and mitigation methods. Our main work starts from section IV where we describe online dataset creation, which is used to train and test J48, Support Vector Machine (SVM), K-Nearest Neighbors (K-NN) and Random Forest models. Section III also presents the evaluation of the models based on various performance measures. Next in section V the selected machine learning model is applied in the test SDN network and the proposed DDoS attack detection and prevention framework is presented. Finally, Section VI presents conclusions and future work.

II. SDN OPERATIONS AND THREATS

A. Topology Discovery in SDN

The first process before packets are forwarded in an SDN network involves data plane topology discovery. The controller learns and keeps an up-to-date information of the data plane topology using the OFDP (OpenFlow Discovery Protocol) standard. The network devices advertise their identities,

978-1-7281-3851-0/19/$31.00 ©2019 IEEE
DOI 10.1109/SERVICES.2019.00051

capabilities and neighbors in the network using the Link Layer Discovery Protocol (LLDP) to the controller [3].

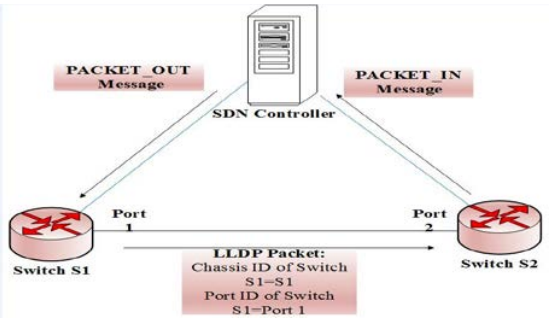

Figure 1: Link discovery with LLDP (adapted from [3])

For example, consider Figure 1, for the controller to learn about the connection between Switch S1 on port 1 and Switch S2 on port 2, the controller needs to initiate link discovery. The controller sends a PACKET_OUT message containing instructions to Switch S1 to forward LLDP packet through its port 1. The LLDP packet that is sent by Switch S1 has its Chassis ID and Port ID information. After Switch S2 receives the LLDP packet through its port 2, it generates a PACKET_IN message and sends it to the controller, so that it can add the link between Switch S1 and S2.

B. *Packet Forwarding*

The most common protocol that is used to enable secure communication between the SDN controller and switch is OpenFlow. This communication is done over the controller's southbound interface. The controller is responsible for managing and creating flow rules which have the matching fields and instructions regarding a flow in its flow table. These rules are sent to the SDN OF switch which then adds the entries in its own flow table. Consider the diagram in Figure 2 below, when a new packet is received by the OF switch S1 as shown by arrow (1). Switch S1 will look up its flow table and see if it has a matching entry from previous flows. If a match is found then it will be forwarded accordingly as shown by (5). However, in case if this was the first packet to the destination, the switch will forward this packet to the controller as a PACKET_IN message as shown by (2). The controller will then look up its own flow table and see if a match is found, and then it will create a flow rule based on the instruction in its flow table. It will then send the new rule (shown by arrow (3)) to the switch, which will store it in its flow table to be used for any subsequent packets for the same flow.

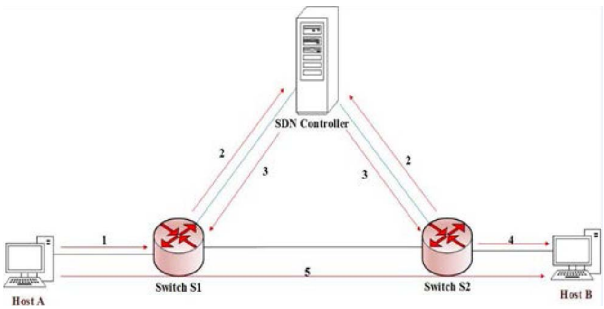

Figure 2: SDN Packet Forwarding (adapted from [2])

C. *Threats to SDN Planes*

However, the same concept of control plan and data plane separation introduces a new problem by increasing the attack surface of the SDN network when compared to conventional networks [1][4]. SDN network is vulnerable to various traditional attacks such as spoofing, tampering, repudiation, information disclosure, denial of service, elevation of privilege, unauthorized access (of switches, controller and applications), data leakage (in the form of flow rule discovery and packet processing timing analysis), malicious applications (to install fake rules), man-in-the-middle attack, and other configuration issues [1].

SDN network consists of three layers: Application, Control and Data Forwarding. As shown in Figure 3, the Application layer interacts with the control layer through the Northbound API, while the Control layer interacts with the data layer through the Southbound API using the OpenFlow protocol. The various resources that can be impacted by attacks in an SDN network can be seen in Figure 4.

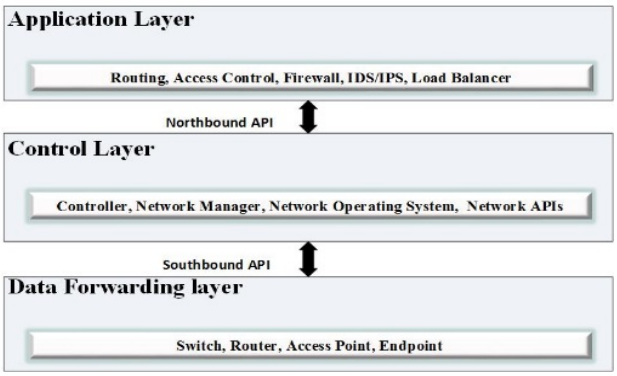

Figure 3: SDN Planes. Adapted from [1]

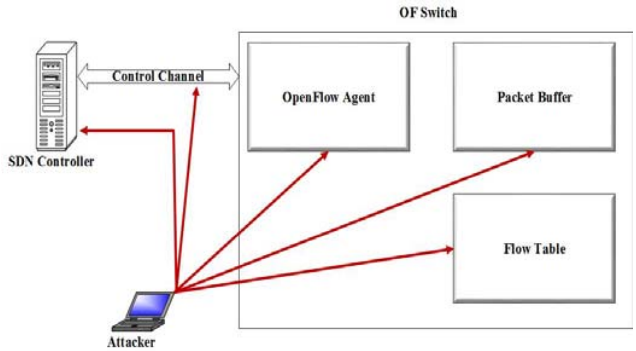

Figure 4: Resources vulnerable to attacks in SDN (adapted from [1])

D. *Security Threats to Application layer*

The application layer consists of several independent applications such as routing, access control, firewall, IDS, intrusion prevention system (IPS), and load balancer. These are installed, managed or removed by the administrator based on the requirements of the network and they communicate with the

control layer to execute their functions. The centralized architecture of SDN makes application deployment easy. However, this introduces security vulnerabilities such as Unauthorized/Unauthenticated Access Control, policy violation and flow rule contradiction. Applications gather information about network statistics from the controller and this can be used with malicious intent to compromise the networks security policy or to add flow rules that can manipulate and disrupt the network functionality [1]. There is also an absence of any common standard or process for software development for SDN. This can cause conflicts between different security goals of different software. It can also create problems of interoperability between applications of various SDN providers and manufacturers, running in a common SDN-controlled infrastructure [4]. The arrows (1) and (2) in Figure 5, show the main security challenges of the SDN Application Layer.

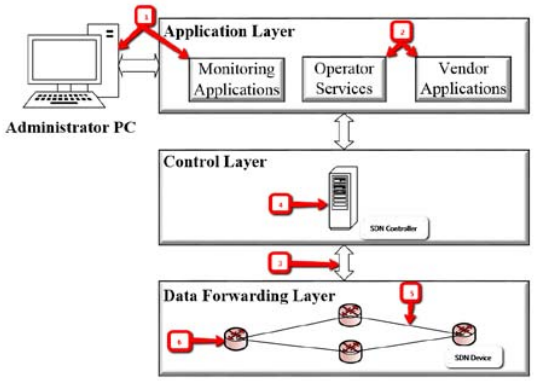

Figure 5: Vulnerable Surfaces in SDN (adapted from [4])

Unauthorized/Unauthenticated Access Control

If an attacker can gain access through vulnerabilities in SDN administrative station or poorly designed third-party application, they can gain access to sensitive information about the controller and inflict maximum damage to the SDN network. As we know, attacks such as brute force are becoming very sophisticated. They can use any existing vulnerability in the SDN applications to gain unauthorized access to the SDN network and control or destroy the entire network configuration from a single point.

Flow Rule Contradiction

Because of the lack of common standard between the SDN applications, it becomes hard to detect a compromised application. After an attacker can gain unauthorized access to the controller or exploit a weakness in an existing application, they can create random or false rules and generate flows to gather information about the entire network. For example, and attacker can create rules to redirect flows to a packet sniffer and gain sensitive information.

E. Data forwarding layer

The Data forwarding layer is responsible for forwarding the packets based on the controller instruction. It consists of networking devices such as switches, routers, access point and endpoints (hosts and servers) all interconnected to make up the SDN network topology. The network devices communicate with the controller through dedicated physical or logical links to get instructions (flow rules) on forwarding of packets [1]. The routers and switches can easily be configured by the controller in a granular and flexible manner [4].

When a packet arrives at an OF switch, it looks up the packet header for a matching flow rule in its flow table. Once a matching rule is found, the OF switch forwards the packets accordingly. However, if no matching rule is found, the packet header or the whole packet based on the network policy will be forwarded to the controller through the control channel by the OpenFlow agent in the switch [1][4]. If the network policy dictates only packet header forwarding to the controller, the OF switch stores the data portion of the packet in its packet buffer. Vulnerabilities in this process can be exploited to forge switch flow rules as well as perform flood attacks. The arrows (3), (5) and (6) in Figure 5, show the main security challenges of the SDN data forwarding layer.

Forged Switch Flow Rules and Flooding Attacks

Due to the lack of intelligence in the OF switches, they have no way of differentiating genuine flows form malicious ones, which makes them exposed to attacks. If an attacker floods the switch with a large number of packets with different source and destination addresses for which the switch has no rules in its flow table, the switch will send the packets to the controller to generate flow rules. The controller will install these new flow rules into the switch's flow table. But since the size of the flow table is limited, it may soon overflow with no room for storing more rules. Until space is created in the flow table by deletion of older rules that expire, the switch will not be able to forward any traffic which causes packet drops and high latency of regular packets and overall network disruption. Secondly, the OF agent can also be overloaded by flooding of packets. If the packets arrive at a rate higher than what the OF agent can handle, it will be unable to forward packets to the controller, thus resulting in packet drop. Thirdly, the OF switch's buffer for storing the data portion of the packet can also overflow, requiring entire packets to be sent to the controller which adds to more latency and packets drops [1][4].

F. Control Layer

The main components of the control layer are the controller, network manager, network operating system as well as network APIs, as shown in Figure 3. The controller installs flow rules and responds to requests of the OF switches and together with the network operating system it controls the entire SDN network. The control layer acts as a mediator between the Application layer and data forwarding layer, by translating application layer requirements to the data layer [1]. Since the controller is the central and most important unit performing all decision making in the SDN network, it is very important to secure it from unauthorized access [4]. The communication to the SDN controller is done through secure lines to try to reduce exposure of the controller to security threats. However, the inherent nature of SDN operation makes it vulnerable, such that an attacker can simply make the controller unavailable to

respond [2].The most common threat to the network controller is the DDoS attack. The controller can be overwhelmed if it receives a flood of bogus a flood of PACKET_IN messages, causing it run of our memory and computing resources. The controller's response to switch requests will be slow and after it completely runs our of memory and resources, it will be unable to perform its functions which may bring down the entire SDN network [1]. The arrows (3) and (4) in Figure 5 show the main security challenges of the SDN control layer

Control channel

The controller and the OF switch communicate via an out-of-band channel which uses a dedicated physical link connection to the controller, or an in-band channel which uses existing data plane connections between the switches. When an attacker initiates a DDoS attack by flooding the controller with many PACKET_IN messages, the channel links capacity can be exhausted resulting in congestion, delay and packet drops. The in-band channel link will be impacted more severely by an attack, as thy are used for both control and data forwarding.

Denial-of-Service (DoS) attacks

The separation of control and data plane of the SDN network provides many benefits, but it also introduces a new challenge of its exposure to several types of attacks. One such attack that has shown to have a devastating effect on the SDN network is the Dos or Distribute Dos. According to [5], "A Distributed Denial of Service attack is an attack on a server where a massive number of packets are sent to create an outage or service degradation for legitimate users or depriving the organization of necessary computer services, such as access to the Internet, email, on premise, hosted, or cloud services" .

III. DDoS Classifications and Features

A. DDoS Classification

As already mentioned, a DDoS attack aims to flood a network with a large number of packets to various destinations. DDoS attacks can flood the victim network in several ways: TCP, UDP, ICMP flood attack, Random IP Flood attack as well as using Botnets.

TCP Flood

The most common DDoS attack is the TCP flood attack. The TCP flood attacks send a huge amount of TCP connection requests to the victim without acknowledging the SYN-ACK of the victim server. The victim server is left with many half open connections. These half connections consume all or most of its resources making it unavailable to authentic users [6].

ICMP Flood

Another type of DDoS is the ICMP flood attacks also called smurf attack. It floods a victim with a large number of ICMP packets using spoofed source IP addresses. The victim server will respond to unsuspecting owner of the spoofed IP address with the ICMP replies. This will affect the performance and availability of both the victim server and the actual owner of the spoofed IP address [6].

UDP Flood

The third type of DDoS attacks are the UDP flood attack. They flood the victim with a large number of UDP packets. One such example is the DNS amplification attack in which the attacker spoofs the source IP address of the victim and sends a small request to the DNS server. The DNS server replies with large responses affecting the performance of the victim. The UDP flood attack can also be triggered by simply flooding the victim with a large number of UDP packets to make it unavailable to regular users [6].

Random IP Flood

DDoS attacks can also be started by generating random IP packets so that the controller is kept busy trying to respond to the bogus packets and thus unable to respond to other legitimated traffic [4]. For the DDoS attack to be successful it needs time to achieve a high rate malicious packets and it can also occur at a specific time of the day on a regular basis [2]. For example, it occurs every day at 7 am.

Botnets

A more complicated and deadly form of the DDoS attack is a Botnet. Botnets are an army of compromised computers [5]. Several easy to use tools to generate attacks are freely available or at a little cost. Generating attacks is a big business, and anyone can easily find the recourses or hire others to perform any kind of attack thorough the Internet. A botnet is created by installing a malware on a computer through dubious means. It could be realized using phishing, SPAM emails, website links or downloads, sent to unsuspecting users. The malware uses the infected computer and connect with its command and control (C&C) server of the botnet owner. The C&C server then sends instructions to all the infected computers (can be several thousand computers), through peer to peer communication and collaboration and controls them to damage a victim's network/servers. The botnet increases its size and effectiveness by combining the capacity of the infected computers to generate and transmit large attack volumes.

B. DDoS Detection and Prevention

DDoS attacks have become more sophisticated with the development of new technology. Normally, network traffic consists of a combination of normal and maliciously traffic. This traffic needs to be monitored and analyzed by organizations to prevent violation of policies and protect against attacks [7]. Blocking of the supposed attackers IP does not help in mitigating the attack due to IP spoofing; hence, we need to focus on the characteristics of the attack to be able to mitigate it. DDoS attacks are very difficult to detect because of their similarity to normal traffic. However, there are some features that are common to all DDoS attacks. Firstly, the malicious packets that are used for the DDoS attack have the same destination and port addresses. These packets also have a typical size, which is different from the size of legitimate traffic [2]. Organizations deploy various preventive measures such as firewalls, access control lists, antivirus software's and IDS/IPS to prevent from unauthorized access [8] [9]. Malicious activities should be caught as soon as they occur. Therefore, the speed of detection, accuracy and reliability of the IDS is vital

to ensure it performs its functions properly without any negative impact to the organization [10].

One major approach that has found increased popularity in the research community to provide an efficient IDS solution is the use of machine learning techniques, especially in SDN [8][10]. Machine learning deals with the construction and study of systems that can learn from data without being explicitly programed to do so [8]. They discover the relationship between the inputs and output from a sample data [7].

Machine learning techniques can be used to identify DDoS attacks and normal traffic and thus to mitigate DDoS attacks. Some of the features that can help to detect a DDoS attack are number of packets, average of packet size, number of bytes, packet and bit rate, etc. [6]. Our paper investigates some commonly used machine learning classifiers for anomaly detection, i.e., SVM, Random Forest, J48 and K-NN.

IV. SDN DDoS Detection and Mitigation Model

A. Training and Testing Dataset

To generate normal and DDoS packets (ICMP and TCP floods), *hping3* program was used in a Python script. DDoS and normal traffic were captured separately using *tshark* to avoid confusion in labeling the dataset. The DDoS traffic was flooded at a rate of 78 packets per seconds and was captured for 30 minutes. After the DDoS capture was complete, the normal traffic was run, and it was captured for 3 hours to balance the number of DDoS and normal traffic.

The *tshark* captured files, are then converted from packet capture (*pcap*) to *CSV* format to extract the relevant features and at the same time filter out irrelevant data such as address resolution protocol (ARP). In this paper, we focus on DDoS detection and mitigation in an SDN network using machine learning, thus we need to use features that can easily be extracted without overloading the network. To meet our goal, we focused on 24 packet level data features described in [11], since they are faster and simpler to generate.

B. Data preprocessing

The raw data capture undergoes preprocessing to make it suitable to train the proposed model and avoid overfitting [12]. The following issues were handled using Weka preprocessing.

- The Classifier column was converted from numeric data type to nominal type for classification.
- The missing values for nominal and numeric attributes in the dataset were replaced with the modes and means from the training data.
- The dataset rows were randomized because during initial combining, the normal and DDoS traffic were in a series (not a random mix of normal and DDoS traffic).
- Due to the random nature of normal traffic and fast packet arrival rate of DDoS traffic, the training dataset had about 566,611 DDoS packets and only 85,853 normal traffic. To avoid the chances of an overfitting model, the Class Balancer Weka filter was used to reweight the instances in the data so that each class had the same total weight. Therefore, we had balanced set of 326,232 instances of both normal and anomalous classes for training dataset and 80,915 instances of normal and DDoS classes for testing dataset.
- Finally, the results of the preprocessing were saved and used to train and test J48, Random Forest (RF), SVM, K-NN for performance evaluation.

TABLE I. COMPARISON BETWEEN J48, RANDOM FOREST, SVM, KNN

Algorithm	Recall	F-Score	Kappa Statistic	Root mean squared error	Training Time (Sec)	Testing Time (Sec)	Sensitivity	Specificity	Precision	Accuracy
J48	1	1	1	0.0001	17.43	3.03	1	1	1	1
RF	1	1	1	0.0914	171.11	5.19	1	1	1	1
SVM	1	1	1	0	168.59	1.97	1	1	1	1
K-NN	1	1	1	0	0.13	15957.7	1	1	1	1

To rule out the chances of an over fitting model, the model was evaluated using various train and test splits, 10-fold cross validation as well as separate training and testing datasets. From our analytical comparison of J48, Random Forest (RF), SVM, KNN, as shown in TABLE I, it was concluded that J48 classifier is best suited for our scenario due to its high accuracy, specificity, sensitivity, Kappa, Precision, Recall, F-Score and fast training as well as testing time. More details about the performance measures can be found in [7]. We saved the trained J48 model to be used for classification of online DDoS and normal network traffic in our SDN network.

V. Experiment Results

The entire experiment is carried out on Ubuntu (18.10) Virtual machine setup on VMware with a 4GB of RAM and 40GB hard drive space. Mininet (2.3) is used for creating the SDN network with a RYU (4.3) controller, while Weka (3.9.3) is used for training and testing our machine learning model. The SDN network contains the following units: RYU controller, OpenFlow switch, two web servers W3/W4, attacker PC, normal PC and host h5, as shown in Figure 6.

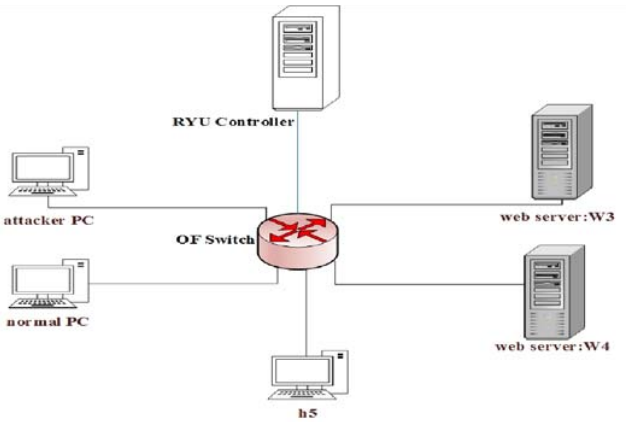

Figure 6: Test SDN Network

The proposed framework uses the following four scripts:

- *Regular Traffic script* – It generates normal HTTP and ICMP packets randomly to all the PC's and servers.
- *DDoS Traffic script* – It floods the web servers with ICMP and TCP packets at a rate of 78 packets per second with randomly generated spoofed source IP address using hping3.
- *Detection script* – It uses the selected J48 model to classify the incoming packets on the OF switch to DDoS and normal packets.
- *Mitigation script* – It uses a REST message to add flow entry through the controller to block the DDoS attackers port on the OF switch.

A. The detection processes

The detection process involves the following steps:

- *Tshark* captures traffic on each interface for a duration of 30 seconds and saves the captures as separate *pcap*.
- These *pcap* files are then converted to separate *CSV* files, which are supplied to Weka. A column called label is added to classify the packets into DDoS and normal. After necessary preprocessing is performed to make the captures compatible with the training dataset the files are saved in an *arff* formats.
- Finally, the *arff* files are supplied to the J48 model, which then uses the information in the *arff* files to provide prediction of the packets into DDoS (1) or normal (0) classes.

B. DDoS Mitigation Process

The prediction of our model from the above detection process is then supplied to our mitigation script. If the prediction from the detection phase is a 1 value for DDoS traffic, then a REST message is sent to the controller to block DDoS computer ports on the OF switch for 30 seconds. However, if the prediction is a 0 value for normal traffic, then no command is sent to the controller and packets from the PC continue unaffected. Our detection and mitigation process worked with high accuracy and takes approximately 10-15 seconds to complete.

VI. Conclusion and Future Work

In this paper, we designed an SDN framework to detect and protect the controller and the OF switch from DDoS attacks. This framework involves training a machine learning model with captured data to predict DDoS attacks. The prediction is then used by our mitigation script to make decisions in our SDN network. We evaluated SVM, K-NN, J48, and Random Forest with online captured data. Our experiment results showed J48 to be the most suitable classifier for our network.

The following directions could be exploited for future research. In our scenario, the attackers port is blocked for 30 seconds and the port is unblocked afterwards for proof-of-concept. However, another approach could be to create a separate analysis server for research and development in the proposed network. This server should have large bandwidth, memory and processing resources. When a host's packets are predicted to be a DDoS attack, then a command to create a flow entry from the attacker's port to the analysis server is sent through the controller. All new packets from the attack computer will be sent to the analysis server for a period. The analysis server which is equipped with powerful analysis and monitoring capabilities further scrutinizes the captured packets to improve the detection and effectiveness of the entire process. Secondly, the time to detect a DDoS attack may be further reduced by minimizing the number of steps for classifying packets using more efficient machine learning tools.

References

[1] "Toward an Optimal Solution Against Denial of Service Attacks in Software Defined Networks - ScienceDirect." [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18302930#bb40. [Accessed: 30-Mar-2019].

[2] N. I. G. Dharma, M. F. Muthohar, J. D. A. Prayuda, K. Priagung, and D. Choi, "Time-Based DDoS Detection and Mitigation for SDN Controller," in 2015 17th Asia-Pacific Network Operations and Management Symposium, 2015, pp. 550–553.

[3] C. Lin, C. Li, and K. Wang, "Setting Malicious Flow Entries Against SDN Operations: Attacks and Countermeasures," in 2018 IEEE Conference on Dependable and Secure Computing, 2018, pp. 1–8.

[4] C. Tselios, I. Politis, and S. Kotsopoulos, "Enhancing SDN Security for IoT-Related Deployments through Blockchain," in 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, 2017, pp. 303–308.

[5] J. Smith-perrone and J. Sims, "Securing Cloud, SDN and Large Data Network Environments from Emerging DDoS Attacks," in 2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence, 2017, pp. 466–469.

[6] B. Zhang, T. Zhang, and Z. Yu, "DDoS Detection and Prevention Based on Artificial Intelligence Techniques," in 2017 3rd IEEE International Conference on Computer and Communications, 2017, pp. 1276–1280.

[7] S. Choudhury and A. Bhowal, "Comparative Analysis of Machine Learning Algorithms along with Classifiers for Network Intrusion Detection," in 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials, 2015, pp. 89–95.

[8] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN Based Network Intrusion Detection System using Machine Learning Approaches," Peer-Peer Netw. Appl., vol. 12, no. 2, pp. 493–501, Mar. 2019.

[9] R. R. R. Robinson and C. Thomas, "Ranking of Machine Learning Algorithms Based on the Performance in Classifying DDoS Attacks," in IEEE Recent Advances in Intelligent Computational Systems, 2015, pp. 185–190.

[10] L. Yang and H. Zhao, "DDoS Attack Identification and Defense Using SDN Based on Machine Learning Method," in 15th International Symp. on Pervasive Systems, Algorithms and Networks, 2018, pp. 174–178.

[11] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Towards Generating Real-life Datasets for Network Intrusion Detection," J Netw. Secur., vol. 17, pp. 683–701, 2015.

[12] P. Pandey and R. Prabhakar, "An Analysis of Machine Learning Techniques (J48 amp;amp; AdaBoost)-for Classification," in 2016 1st India International Conference on Information Processing, 2016, pp. 1–6.