Towards Prediction of Security Attacks on Software Defined Networks: A Big Data Analytic Approach

Emre Unal, Sonali Sen-Baidya and Rattikorn Hewett
Department of Computer Science
Texas Tech University, Lubbock, TX, USA
Emails: Emre.Unal@ttu.edu, Sonali.Sen-Baidya@ttu.edu, and Rattikorn.Hewett@ttu.edu

***Abstract* —Cyber-physical systems (CPS) tightly integrate physical and computing processes by monitoring and control data interacting between them via underlying networks. Software Defined Network (SDN) Technology has increasingly become essential in many advanced computer networks, including those in modern CPS, to provide flexible and agile network development. Despite many benefits that SDN offers, malicious attacks that can eventually prevent network services are unavoidable. Among the most predominant attacks on SDN controller layer, *Link Discovery Attack* and *ARP (Address Resolution Protocol) Spoofing Attack* are fundamental in that they are the gateways of many other SDN threats and attacks. To defend these attacks, most existing techniques either rely on relatively complex data validation techniques or use thresholds that can be subjective and unable to detect more than one type of attacks at a time if one deciding factor is used. While Big data technology, particularly machine learning, has been widely used for intrusion/anomaly detection, little has been done in SDN. This paper explores how well this technology can be used to predict these SDN attacks. By employing typical machine learning algorithms on simulated data of routing in SDN when attacks occur, preliminary results, obtained from four machine learning models, show the average area under ROC curve of over 96% and 92% for sample size 50,970 (12 switches) and 60,000 (20 switches), respectively. Further experiments show near-linear scaling in training time for the best performing algorithm when sample size grows up to 100,000.**

Keywords— Software-Defined Networking; SDN-specific security; Link Discovery attack; ARP Spoofing attack; Machine Learning, Data Analytic Applications

I. Motivation

There has been enormous development of information technology and its applications in the past two decades. with the rise of the Internet usage in industry, business and societies that impact our everyday living and human life, building scalable, reliable and robust networks becomes one of the most important tasks for network development. In today's world, Cyber-physical systems (CPSs) are everywhere from infrastructures such as transportation and power grid systems to automated operations in industry to services in business organizations. CPSs are systems that tightly integrate physical and computing processes by monitoring and control data interacting between them via underlying networks. Sensed data from physical systems are used by cyber systems to compute and determine appropriate action or parameter controls of the physical system components that in turn generate resulting behavioral data as a feedback to the cyber systems. The scalability and complexity of CPS compound the challenges of traditional networks to meet a huge demand for rapid development of flexible and reliable networks. To overcome these challenges, Software Defined Network (SDN) technology has been proposed [10, 11, 13]. It has increasingly become essential in many advanced computer networks, including those underlying networks in modern CPS, to provide flexible and agile network development. By separating functions of control and data (i.e., control plane and data plane), one can program to specify SDN controller to direct switches to deliver network services easily. SDN provides benefits in network programmability, customizability, development flexibility, and lower processing expenses. These mechanisms help to improve the network productivity as a whole and also allow better network management. Despite valuable advantages, like other networking architectures, SDN is unavoidably susceptible to security threats and attacks [2, 5, 8, 17, 18].

There are many attacks (e.g., flow table saturation attack, control plane saturation attack, Topology poisoning attack, spoofing attack, denial-of-service attack, eavesdropping attack) on network infrastructures today which not only compromises the availability of networks, hosts and services, but also the confidentiality and integrity of the network data. Many of these attacks are easy to launch (e.g., spoofing attack [1, 3, 12, 16]) and difficult to trace back to identify attackers [18]. Most importantly, some attacks can lead to DoS (Denial of Service) attack [4, 14, 20] causing devastating consequences of loss of services. Because SDN also includes some functions of traditional networking, many of SDN security attacks also occur in traditional network architectures.

This paper focuses on two attacks of SDN, namely *Link Discovery Attack* and *ARP (Address Resolution Protocol) Spoofing Attack*. In general, the former is SDN-specific and the latter is not (except when it occurs in the SDN-specific architectural components (e.g., links interacting with the SDN controller) In Link Discovery Attack, an attacker aims to deceive the SDN controller's perception by fabricating/ forging a new link in the network that does not exist [7, 9, 19]. By exposing the traffic that traverses over this malicious link to interception and manipulation, the attacker can obtain the network information and take control over the traffic [7]. On the other hand, in ARP Spoofing Attack [1, 3, 12, 16], an

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

attacker aims to pretend to be static gateway of a network by spoofing the SPA (Sender Protocol Address) and SHA (Sender Hardware Address) in the ARP message to create an invalid address mapping [3]. Because ARP treats each request or reply independently, a host can readily accept information from ARP replies without a corresponding request. ARP does not have mechanisms to authenticate the sender, replies, or integrity of information [1]. Thus, an attacker can reply with a broadcast to the network with its MAC address as the physical address of the gateway. In this paper, we consider ARP spoofing on the communication links between SDN controller and switches. Both of these attacks are SDN-specific and serious attacks as they are the front door of many other network threats and attacks, e.g., man-in-the-middle and denial of service (DoS) attacks.

To defend these attacks, many techniques have been proposed [1, 3, 9, 12, 16, 19]. However, most either rely on relatively complex validation techniques (e.g., ARP's Network Address Translation [3]) or use thresholds (e.g., [1, 19]) that can be subjective and unable to detect more than one type of attacks at a time. As Big data technology, particularly machine learning, has been widely used for intrusion/anomaly detection in many CPS, however, little has been done in SDN. Our research aims to explore how well Big data technology can be used to predict these SDN attacks. In particular, a set of typical machine learning algorithms [22] is applied to simulated data of SDN routing communication to predict the presence of each type of these attacks.

The rest of the paper is organized as follows. Section II describes related work and Section III presents overview of SDN and the two SDN attacks. Section IV discusses the proposed approach followed by Section V giving experiments and the results. Section VI concludes the paper.

II. Related Work

Much research has studied SDN security threats at various architectural layers: data layer, controller layer, and application(s) layer [2, 5, 8, 17, 18]. Link discovery attacks are described in [6, 8, 17] along with other network topology poisoning. Hong et al. [6] proposed a security extension to the SDN controller to provide a real-time detection of the network topology misuse. Smyth et al. [19] presents a technique to detect Link Fabrication Attack by observing if the packet traffic exceeds normal threshold. For example, Wang et al. [20] use transfer time to detect fabricated links and man-in-the-middle attacks. While these approaches work reasonably well, its accuracy depends on the set threshold that tends to be subjective. Unlike these approaches, our machine learning approach does not assume a pre-determined threshold.

ARP Spoofing attacks have been studied in both traditional LAN networks and SDN. Several techniques [1, 3, 12, 16] have been proposed to prevent ARP spoofing (e.g., ARP authentication, compare with trusted database, configure ARP entries manually, etc. [1]). Alharbi et al. [3] presents a mechanism to prevent ARP spoofing by making sure that the potentially spoofed information is kept away from any hosts and thereby prevents ARP cache poisoning. For reply based attack, the mechanism will only accept ARP replies for which it has a corresponding request. Abdelsalam et al. [1] does port level ARP packet count monitoring to detect large volume of incoming malicious packets and stops them. This requires threshold to help make decisions. None of the above techniques make use of behavioral data of SDN executions to build a predictive model of each kind of attack as our proposed approach.

While machine learning has been applied to anomaly detection and intrusion detection, at the time of this writing machine learning applications to SDN attacks are few or only mentioned with no empirical studies.

III. Background

This section describes an overview of SDN in Section *A* and the two SDN-specific attacks in Section *B*.

A. Software Defined Networking (SDN)

SDN [10, 11] is an emerging architecture that can be viewed in three planes (or layers). SDN breaks the vertical assimilation by separating the control logic (control plane) from the underlying routers and switches (data plane) that forward the traffic. This enables the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. A simplified view of this architecture has been shown in Fig. 1.

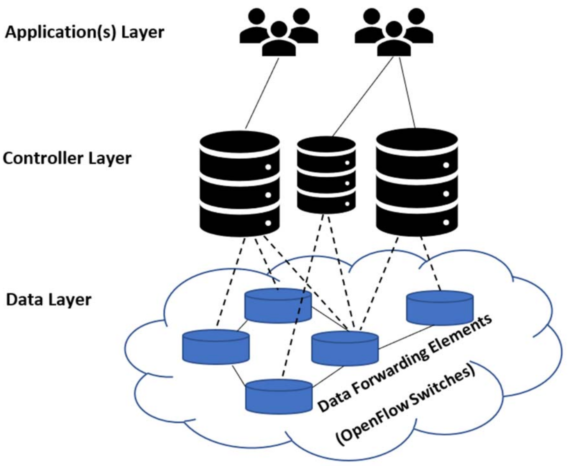

Fig.1 SDN Architecture.

The data plane consists of forwarding devices including physical and virtual switches, which are responsible for forwarding data. The switches offer the view of the programmable flow tables that define an action for each packet related to a specific flow. The control logic which is moved to an external entity, called the SDN controller lies in the control plane of the architecture. The controller is a software platform which has the complete view of the network and the ability to make optimal routing decisions, thus improving the network visibility. It exercises the direct control over the data plane through some well-defined application programming interfaces (API) based on a logically centralized, abstract view of the network. The network is programmable through software applications situated in the application plane which are running on top of the control plane. This plane has a set of applications that implement some network control functions like routing, load balancers, fault-tolerance, recovery, etc.

The separation of the control plane and the data plane can be comprehended by means of a well-defined programming interface between switches and the SDN controller. The controller directly controls the data plane elements through a well-defined application programming interface(API), as depicted in Fig. 1, known as Southbound API. The most notable of such API is OpenFlow. It is one of the first standard protocol that is used for managing the communication between the control layer and the data layer through the Southbound.

The OpenFlow switches has got one or more tables which has the packet-handling rules known as flow tables. These rules direct the devices and the routers regarding the traffic flow. The network managers use these flow tables to modify the layout of the network and the traffic flows.

Similar to Southbound, the Northbound API represents the interface through which the communication between the application layer and control layer takes place. This Northbound interface abstracts the low-level instruction sets used by southbound interfaces to program forwarding devices.

B. *SDN-specific Attacks*

SDN attacks can be initiated from malicious management applications, malicious actions in the controller, or from compromised network entities, namely hosts and switches. The centralized controller makes SDN a unique and powerful architecture to offer benefits in network development and management. Thus, we focus on security attacks that are associated with the controller plane.

1. *Link Discovery Attacks*

One of the key functionalities of the control plane is the network topology service. This service manages and updates the topological information and provides information to the application level-services like routing, network management, policy implementation, security services, etc. Thus, any malfunction or, poisoning on this service would directly impact the other dependent services in the network.

Link discovery is an important process where the controller uses OpenFlow Discovery Protocol, which leverages the Link Layer Discovery Protocol (LLDP) packet, to dynamically detect direct links between adjacent switches. During this process an attacker may attempt to create fake links in the network by manipulating the propagation of LLDP packets in the link discovery process. We call this malicious act the *Link Discovery Attack* (sometimes referred to as *Link Fabrication Attack* or *Link-Layer Discovery Protocol Attack* or *LLDP Attack*). The attacker can either generate the fake LLDP packets and send them to the target switches or relay the genuine packet received from the one target switch to another. In both cases the controller thinks that a link exists between two switches, but actually it does not. As a result, if the traffic is routed through those false links will either get dropped or go through those malicious switches, thereby letting the attacker eavesdrop on the traffic [19]. See more details in [7, 9, 19].

2. *ARP Spoofing Attacks*

ARP (Address Resolution Protocol provides the service to resolve the mapping of network layer addresses like, IP addresses to link layer addresses like, MAC addresses. To find the MAC addresses a node broadcasts an ARP requests with the Sender Hardware Address (SHA) field set to its own MAC address and the Sender Protocol Address (SPA) field set to its own IP address, while the Target Protocol Address (TPA) set to the IP address of the target node and the Target Hardware Address (THA) is initialized to a dummy value. Each recipient node will check if the TPA value in the request matches its own IP address, and if so, will respond with an ARP Reply message.

Spoofing attack occurs when an attacker impersonates another user's identity on a network. By this way, the attacker can steal other user's data or bypass the access control. ARP is a stateless protocol, which implies that it does not retain any information of any requests or replies. Because ARP treats each request or reply independently, a host can readily accept information from ARP replies without a corresponding request. ARP does not have mechanisms to authenticate the sender, replies, or integrity of information [1]. Thus, it is easy for an attacker to poison a host's ARP cache with a false IP-MAC address mapping. See more details in [1, 3, 12, 16].

IV. PREDICITION OF LINK DISCOVERY AND SPOOFING ATTACKS

A. *Proposed Approach*

Using a big data analytic approach ultimately involves applying an analytic algorithm in a distributed parallel computing framework in order to cope with enormous amount of data that do not fit in one machine. A variety of such frameworks exist with different computing paradigms (e.g., MapReduce, Storm, Spark, etc.). However, before dealing with large-scale data, we want to identify effective techniques for smaller data set size.

As a preliminary study, we propose to apply machine-learning algorithms to build a predictive model for predicting the presence of attacks. In particular, we apply four popular algorithms using different forms of models, namely, *Regression, BayesNet, Decision Tree* and *Decision Table*. Regression classifier is based on a mathematical model from a regression method in statistics where classifier is built to recognize one class at a time. Bayesnet is a probabilistic model based on bayes rule using conditional probabilities. Bayesnet is in the same family of Naive bayes classifier and Bayes Network learning algorithm. Decision tree is one of the most popular machine learning algorithms because of its easy to understand tree model and its ability to produce models with high accuracy for most data. Finally, the decision table can be viewed similar to decision tree learning where the resulting model is a table and each row represents a decision rule or a path in the tree. More details of these machine-learning techniques can be found in [22]

Most anomaly or intrusion detection techniques for networks rely on monitoring critical measures of the communication (e.g., number of packets transmitted).

Abnormal number of this measure is observed. If the difference between normal and abnormal numbers is greater than some specified threshold then potential attack is identified. While this technique is simple, it is also problematic since determining the threshold can be subjective. Furthermore, if only one performance measure is used, the threshold-based technique is unable to detect more than one type of attacks at a time. Our approach aims to overcome this limitation and at the same time to obtain increased accuracy compared to that of threshold method. With the use of machine learning our approach can predict multiple classes and as a result distinguish different types of attacks to occur. Next section deals with how we acquire data for use in our experiments.

B. *Data Simulation*

Most research studies in SDN architecture use a simulation of network and controller for evaluation. Here we use simulated data from Mininet [5] for emulating virtual SDN/OpenFlow networks, and POX, a Python-based controller for software-defined networking. The forwarding devices are Open vSwitch. We configured the Mininet with a specific network topology. The malicious switches and hosts vary depending on the test being run. The speed of the communication links was set to 1 Gbps. We assume that the links have enough bandwidth and the network has no traffic congestion. When the network functions normally, Transmission Control Protocol (TCP) packets are sent from the source host to the destination host at the transfer rate of 520 pps. The data packets for the attacks have been simulated using Scapy library [23], which is commonly used for communication packets. In this paper, our simulation considers only cases when the network has a single attack at a time. The attack can be either the Link Discovery Attack (Link fabrication, LLDP attack) or the ARP spoofing attack.

In the Link Discovery Attack, an attacker's goal is to give the controller's a perception of link that does not exist. In general, the controller discovers "Link" in the topology if there is an evidence of a packet sent between the two nodes. This evidence is shown by the recipient's node sending a message to the controller notifying a packet being sent from a sender's ID and address. Thus, we can simulate a "fake" link between two nodes by injecting a falsify message to the controller notifying a packet being sent from either a non-existing sender's ID or from a legitimate sender's ID/address but with a non-existing/unused port.

In the ARP spoofing attack, an attacker wants to spoof information of the target's node or poison the target node's ARP table that contains IP/MAC addresses of all nodes in the network. The attacker can use Ettercap tool [14] to scan through the network to identify vulnerable node to compromise. From the compromised node, the attacker sends a message to a target's node pretending to be a legitimate node in the network. This way, the target's node leaks its information by sending packets to the attacker (through the compromised node) thinking that he is legitimate. Thus we can simulate the ARP attack by sending a packet to a target's node with a falsify header's information (e.g., fake IP/MAC address). We tested the simulated attack mechanism by Wireshark [19], a packet analyzer tool to see the ARP table content.

The simulation was run on HP v7x machine having Intel(R) Core(TM) i7-5600U CPU at 2.60 GHz, and 8GB of RAM, running 64 bit Windows 10 Pro. Table I presents a detailed description of the attributes that we simulated for the prediction of the attacks in all of our experiments. Each data instance (row) maintains information about a specific node in the network during each of its communication to and from adjacent nodes and the controller during a packet routing between source and destination nodes.

TABLE I DATA DESCRIPTIONS

No.	Attribute	Description (Data Type)
1	NodeID	Id of the node (numeric)
2	NodeAddress	Address of the node (numeric)
3	NodePort	Port of the node (discrete: 1 to 4)
4	C-Address	Address of the controller (numeric)
5	C-Port	Port of the controller (discrete: 1 to 4)
6	BandwidthUsed	Utilized bandwidth between this node to one of its adjacent nodes (numeric)
7	Packet-Rcvd	# packets received by this node from all incoming links (discrete range: 1to 1112)
8	Packet-Sent	# packets sent by this node to all outgoing links (discrete: 1 to 1648)
9	Transmitted_bytes	Total # of transmitted bytes of this node both sent and received (numeric: 1 to 113382)
10	Rcvd_bytes	Total # of received bytes for this node from all incoming links (numeric: 1 to 75408)
11	Rcvd_pkt_in_Cntr	Controller's total # of received packets (numeric: 1 to 1059)
12	Sent_pkt_out_Cntr	Controller's total # of sent packets (numeric: 1 to 1648)
13	Trans_bytes_Cntr	Controller's total # of transmitted packets (numeric: 1 to 46144)
14	Class	Status of the network (No attack, LLDP attack, ARP attack)

As seen in Table I, not all of these data are necessary to be predictive factors of the classification model for attack prediction (e.g., NodeID, C-Address). However, we include them for completeness of the information about a specific node. First step of data analytic is to clean the data set based on domain knowledge. However, in this paper, we omit this step for simplicity and also to show power of machine learning algorithms.

V. EXPERIMENTS

A. *Experimental Setup*

Our experiments are divided into two scenarios: 1) small network of 13 nodes (10 switches and 3 hosts) and 2) mid-size network of 21 nodes (18 switches and 3 hosts). For each scenario, we simulated data transmission from a source host to a destination host in SDN network with no attack and with each type of attack (i.e., LLDP link fabrication attack and ARP spoofing attack). For each attack in each scenario, we simulated the attack on two specific network locations, first of which is near the source and second of which is near the destination. Recall that we only consider when only one single attack occurs at a time during an attack period. The details of these attacks will be described later. Combining all simulated data of each case in each scenario, we obtained two data sets whose characteristics can be summarized in Table II.

TABLE II DATASET SIZES IN TWO NETWORK TOPOLOGIES

Scenario	# Nodes	# No Attack Instances	#ARP Attack Instances	#LLDP Attack Instances	Data set size
1	13	10,270	20,291	20,409	50,970
2	21	20,000	20,000	20,000	60,000

By using Weka data mining tool [22], the experiments were run on Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz (4 Core CPUs and 4 Logical CPUs) with 2.5GHz, and 8192MB RAM on Windows 10 OS. Below gives results obtained from each scenario.

B. *Experimental Results*

1. *Scenario* 1: *Small Network*

A topology of a small network with three hosts (*X, Y, Z*) and 10 switches is shown in Figure 2.

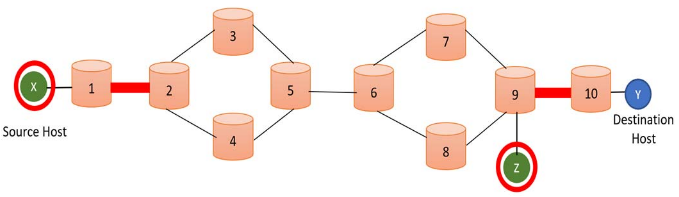

Fig.2 Small Network of 13 nodes

For LLDP attack, by compromising host *X*, an attacker aims to create a forged link between switches 1-2, giving a link fabrication attack. We also simulated another link fabrication attack between switches 9-10 at a different time. For ARP spoofing attack, *X* and *Z* are victim hosts for spoofing. The attacks are indicated as thick lines for two events of link discovery attacks and circles for two events of spoofing attacks.

After shuffling, the data are split into training (66%) and testing data sets for the rest of the data. Table III shows average results, obtained by each of the machine learning algorithms, over 10 experiments.

TABLE III COMPARISON RESULTS ON AVERAGE FOR SMALL NETWORK

Algorithm	Accuracy	TP Rate	FP Rate	ROC
Regression	99.87	0.98	0.001	0.98
Decision Tree	99.97	0.98	0.001	0.99
BayesNet	99.77	0.97	0.002	0.97
DecisionTable	99.72	0.96	0.002	0.96

As shown in Table III, all algorithms obtained over 99% of accuracy with low false positive and high ROC of over 0.95 in each of the machine learning models. Because of excellent results, we wonder if the results are over-fitting to the sample even though our sample size for this small network has 50,970 instances. To investigate further, we conduct experiment on an increased size of the network.

2. *Scenario* 2: *Mid-size Network*

A topology of a mid-size network with three hosts (*X, Y, Z*) and 18 switches is shown in Figure 3.

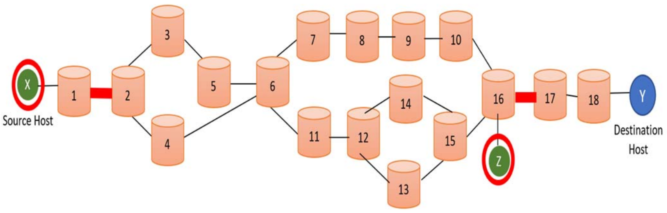

Fig.3 Mid-size Network of 21 nodes

In this scenario, for LLDP attack, by compromising host *X*, an attacker aims to create a forged link between switches 1-2, giving a link fabrication attack. At a different time, host Z was compromised and fake link between switches 16-17 was created. For ARP spoofing attack, *X* and *Z* are victim hosts for spoofing. The attacks are indicated as thick lines for two events of link discovery attacks and circles for two events of ARP spoofing attacks.

Experiments were run for various machine-learning algorithms. Table III shows average results, obtained by each of the machine learning algorithms, over 10 experiments.

TABLE IV COMPARISON RESULTS IN MID-SIZE NETWORK

Algorithm	Accuracy	TP Rate	FP Rate	ROC
Regression	79.96	0.800	0.100	0.952
Decision Tree	79.10	0.791	0.104	0.943
BayesNet	71.23	0.712	0.143	0.922
Decision Table	76.96	0.770	0.115	0.939

As shown in Table IV, Regression obtained the highest accuracy followed by Decision Tree with no significant differences. They are top two performers that obtained accuracy of about 79%, low false positive of 0.1 and high ROC of 0.9. While BayesNet and Decision Table do not receive accuracy as high, their performances are relative good. In general, the results of the mid-size network show that performance of predictive models are sensitive to size and structure of the network. Regression's and Decision Tree's performance are superior to BayesNet and Decision Table. However, the general results by ROC of both scenarios are close to one with ROC of scenario 1 slightly better. ROC is a better measure for quality of predictive models because it takes not only accuracy but also false positives into account.

3. *Scalability*

As data become large, one concern is how well these machine learning algorithms scale in terms of their training time. We investigate this by measuring the training time of each of these algorithms as the training set size grows.

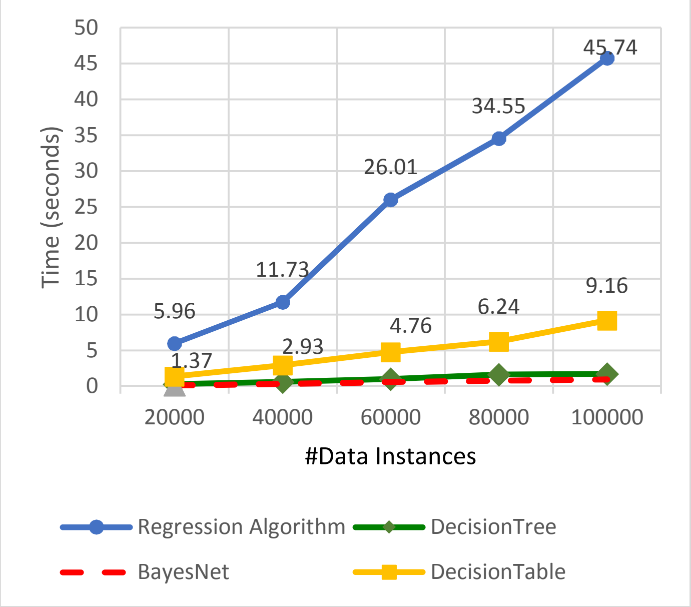

Fig. 4 Training times vs. training set sizes

As shown in Figure 4, the training time of each of the machine learning algorithms is near linear with different rates. The Regression technique takes most time while the Decision Tree and BayesNet are top two performers. These preliminary results are encouraging.

VI. Conclusion and Future Work

In this paper, security challenges in SDN network have been addressed. We propose an approach to predict attacks in the SDN networks by applying machine learning techniques instead of using the traditional technique with threshold values, which tend to be problematic due to dynamic environments of SDN. Our proposed method not only predicts the presence of attack but also attack type by a predictive model, which represents the behaviors of the network, particularly under ARP attack, LLDP Attack or no attack. However, pinpointing the location of the attack is as important as predicting the existence of attack. We note during experiments that our proposed approach has potential to locate the origin of the attack. The future work aims to identify methodology to determine the location of these attacks as well as applying machine learning algorithms in distributed and parallel computing environments for large-scale systems.

References

[1] AbdelSalam, A.M., A. El-Sisi, and V. Reddy, "Mitigating ARP Spoofing Attacks in Software-Defined Networks," in Proceedings of the International conference on Computer Theory and Applications (ICCTA), 2016.

[2] Ahmad, I., S. Namal, M. Ylianttila, and A. Gurtov, "Security in Software Defined Networks: A Survey", IEEE Communications Surveys & Tutorials, Volume: 17, Issue: 4, pp: 2317 – 2346, 2015.

[3] Alharbi, T., D. Durando, F. Pakzad and M. Portmann, "Securing ARP in Software Defined Networks," in Proceedings of IEEE 41st Conference on Local Computer Networks (LCN), pp.523-526, 2016.

[4] Ashraf, J. and S. Latif, "Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques," in Proceedings of Software Engineering Conference (NSEC), 2014 National, pp. 55–60, 2014.

[5] Eskca, B., O. Abuzaghleh, P. Joshi, S. Bondugula, T. Nakayama, and A. Sultana, "Software Defined Networks Security: An Analysis of Issues and Solutions," International Journal of Scientific & Engineering Research, Volume 6, Issue 5, pp.1270-1275, 2015.

[6] Fontes, R., S. Afzal, S. Brito, M. Santos and C. Rothenberg, "Mininet-WiFi: Emulating software-defined wireless networks," in Proceedings of 11th International Conference on Network and Service Management (CNSM), 2015, pp. 384-389. MININET

[7] Hong, S., Xu, L., Wang, H., and Gu, G., "Poisoning network visibility in software-defined networks: New attacks and countermeasures", Network and Distributed System Security (NDSS) Symposium, USENIX, 2015.

[8] Hu, Z., Wang, M., Yan, X., Yin, Y. and Luo, Z., " A comprehensive security architecture for SDN", Intelligence in Next Generation Networks (ICIN), 18th International Conference on (pp. 30-37), IEEE, 2015.

[9] Khan, S., Gani, A., Wahab, A. W. A., Guizani, M., & Khan, M. K., "Topology discovery in software defined networks: Threats, taxonomy, and state-of-the-art," IEEE Communications Surveys & Tutorials, 19(1), 303-324, 2017.

[10] Kim, H. and Feamster, N., "Improving network management with software defined networking," IEEE Communications Magazine, 51(2), pp.114-119, 2013.

[11] Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S. and Uhlig, S., "Software-defined networking: A comprehensive survey", in Proceedings of the IEEE, 103(1), pp.14-76, 2015.

[12] Masoud, M. Z., Y. Jaradat, and I. Jannoud. "On preventing ARP poisoning attack utilizing Software Defined Network (SDN) paradigm." In Proceedings of Conference on Applied Electrical Engineering and Computing Technologies (AEECT), IEEE, 2015.

[13] Mendiola, A., Astorga, J., Jacob, E. and Higuero, M., "A survey on the contributions of software-defined networking to traffic engineering," IEEE Communications Surveys & Tutorials, 19(2), pp.918-953, 2017.

[14] Mousavi, S., St-Hilaire Marc. "Early Detection of DDoS Attack Against Software Defined Network Controllers," Journal of Network and Systems Management, Vol.26(3), pp.573-591, 2018.

[15] Ornaghi, A. and M. Valleri, "Ettercap," available at URL: http://ettercap.github.io/ettercap. October, 2018.

[16] Ramachandran, V. and S. Nandi, "Detecting ARP Spoofing: An Active Technique", ICISS 2005: Information Systems Security, pp 239-250, 2005.

[17] Scott-Hayward, S., O'Callaghan, G. and Sezer, S., "SDN security: A survey," Future Networks and Services (SDN4FNS), IEEE SDN, pp. 1-7, 2015.

[18] Shin, S. and G. Gu, "Attacking software-defined networks: A first feasibility study," in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN '13), ACM, 2013.

[19] Smyth, D., S. McSweeney, D. O'Shea and V. Cionca, "Detecting Link Fabrication Attacks in Software-Defined Networks", in Proceedings of 26th International Conference on Computer Communication and Networks (ICCCN), 2017.

[20] Wang, H., L. Xu, and G. Gu, "Floodguard: A dos attack prevention extension in software-defined networks," in Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'15), June 2015.

[21] WireShark, "WireShark Official Documentation," available at URL: https://www.wireshark. org/docs, October, 2018.

[22] Witten, I.H., Frank, E., Hall, M.A. and Pal, C.J., Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016.

[23] Sacpy, available at URL: P. Biondi, http://www.secdev.org/projects/scapy/