Advances in Computer Science Research, volume 65

3rd International Conference on Communications, Information Management and Network Security (CIMNS 2018)

A Lightweight Traffic Anomaly Detection Model in SDN Based on Decision Tree

Dong Li* and Zizhun Li

Network and Computing Center, HuaZhong University of Science and Technology

Wuhan, Hubei, CHINA

*Corresponding author

***Abstract*—This paper establishes a SDN-based network traffic anomaly detection model based on decision tree. Firstly, five statistical indexes are presented to describe the behavior of network traffic, then normal and abnormal traffic data to train the machine learning model to detect traffic anomaly. Four machine learning algorithms, decision tree, k-nearest neighbor, support vector machine and naive Bayes, are used to predict the detection rate, false alarm rate and other indicators. Experiments shows that decision tree algorithm is suitable for detecting traffic anomaly in SDN network.**

Keywords—SDN; network security; anomaly detection

I. INTRODUCTION

Traffic monitoring is an essential service to network management. Attackers always make use of a botnet or some other malicious software to send massive packets, which will exhaust the target's resources such as bandwidth, memory, CPU, etc. Traffic monitoring aims to discover problems in network traffic and mitigate the hostile activities [1].

Anomaly detection is an important traffic monitoring method to identify the unknown malicious behavior in network. The essence of it is how to classify the network traffic. In traditional network, machine learning is widely used to do it. These techniques are divided into two main classes: one is supervised learning, the other is unsupervised learning. The supervised learning algorithms, such as Naive Bayes, Supported Vector Machine are suitable for the identification of known attacks. Unsupervised learning algorithms, such as K-means, Expectation Maximization, are suitable for detecting unknown attacks. Except for algorithms, data collecting and processing is also very important for traffic anomaly detection.

SDN (Software-Defined Networking) is a new network management platform which separates the control and data planes. Whereby the network intelligence and state can be logically centralized, and the underlying network infrastructure can be abstracted from applications. What's more, SDN has the capabilities of software-based traffic analysis, logically centralized control, global view of the network, and dynamic updating of forwarding rules, which make it easy to detect network traffic anomaly [2].

In related studies, paper [3] presents an overview of using SDN to detect network attacks and concludes that SDN is a platform suitable for the mitigation of DDoS attacks, mainly because of the use of standard protocols, services and interfaces, thus facilitating the deployment of new solutions. In real network, traffic anomaly is always triggered by DDoS and port scan attacks. Paper [4] proposes a solution based on flow collection, feature extraction and flow classification to detect DDoS attack. The traffic features used are the *average number of packets per flow, average bytes per flow, average duration per flow, percentage of pair-flows, and growth of single-flows.* Based on these five indexes, Self organizing maps is used to establish a detection model to perform traffic anomaly detection. Paper [5] proposes an sFlow-based mechanism to obtain information from the network and uses information theory for packet classification, then changes flow rules to block malicious flows.

This paper aims to make use of the controller to collect the traffic information to perform lightweight detection. Compared to anomaly detection in traditional network and related researches, our method has two contributions:

1. We do not need any extra traffic analyzing and measuring tools. In traditional even in SDN network, some traffic tools, such as netflow, sflow is necessary for traffic anomaly detection, but we just use the controller to obtain statistical data without increase the burden of network devices.

2. We make use of some statistical indexes in related paper [4] and put forward some new indexes to describe the traffic anomaly, via comparing, our method is more accurate and effective. Some detailed description lists as follows.

II. DECISION TREES

Decision tree is one of the most efficient classification algorithms. There are three kinds of nodes in a decision tree: root node, decision node and end node. Except the end nodes, each node will split items of training set at that node into more homogenous partitions. A generic decision tree algorithm is characterized by three properties:

The feature selection measure: It chooses the attribute to split training data with high accuracy and evaluates the ability of each attribute to classify training items.

The partitioning strategy: It divides the current training set into several subsets by taking into account the values of the selected test attributes.

The stopping criteria: It ends the growth of a part of the tree and consequently declares the training subset as a leaf.

For decision tree, the attributes are discrete or continuous, here we will especially focus on continuous attributes and

polynomial fit will be used as follows to classify the network traffic.

$$f(X) = c_d X^d + c_{d-1} X^{d-1} + \cdots + c_1 X + c_0 \quad (1)$$

Fit coefficients $c_d, \ldots c_0$ are used to minimize Mean Squared Error (MSE) on training data:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(X_i))^2 \quad (2)$$

In fact, decision tree induction algorithms aim at minimizing two parameters: misclassification rate and tree size.

Misclassification rate: The main objective of a classifier is to pursuit best classification rates. A model not only should correctly classify training data, but also should have a good generalization capacity in order to correctly classify any other new item. Misclassification rate can effectively balance the generalization and accuracy.

Tree size: For better generalization capabilities and better classification CPU time and memory usage, most decision tree learning algorithms tend to build small trees. In order to build such trees, decision tree induction algorithms build decision trees with most discriminating attributes near the root in order to minimize the number of tests needed to discriminate between the different classes. Moreover, decision tree algorithms often involve pruning phases which eliminate subtrees which do not ensure significant classification gain. Another mechanism used for minimizing tree size is the one of stopping criteria which stop the growth of the tree during the induction phase.

III. TRAFFIC ANOMALY DETECTION MODEL

In traditional anomaly detection model, the key is to define normal profiles. If one traffic pattern doesn't consistent with the normal profiles, it is called anomaly. But in real-world network, abnormal traffic triggered by attacks often presents some similar features of normal traffic. For example, a large number of faked IP packets are generated in the network, but if the switch doesn't validate the source IP address, they will be forwarded normally [1].

Feature values can be used to distinguish the type of traffic. In software-defined network, we can obtain the flow table information of switches through the controller.

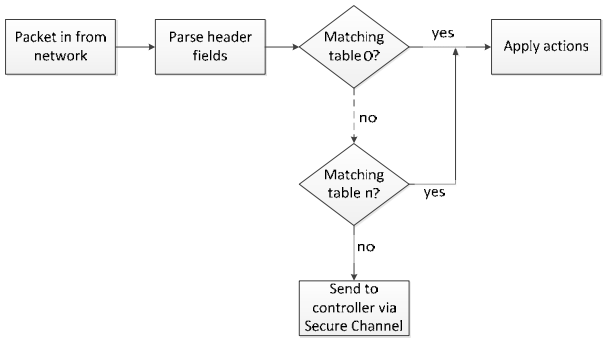

FIGURE I. PACKET FLOW IN AN OPENFLOW SWITCH

TABLE I. FIELDS FROM PACKETS USED TO MATCH AGAINST FLOW ENTRIES

Ingress Port	Ether source	Ether dst	Ether type	VLAN id	VLAN priority	IP src	IP dst	IP proto	IP ToS bits	TCP/UDP src port	TCP/UDP dst port

A. SDN Overview

Software defined network decouples the control plane and data plane of network. The control plane includes three parts: the OpenFlow protocol, a network operating system, and network applications. The OpenFlow (OF) protocol provides a common interface to control how packets are forwarded by accessing the data plane's internal flow tables, configuration, and statistics[6].

In SDN network, only hosts and switches that are registered on the controller are allowed to exchange packets. Every packet which arrives in an OF switch has its header matched against flow entries in the switch's Flow Table. In case some flow entry successfully matches the packet's header its statistics are updated (for instance, the number of bytes and of packets are increased). Otherwise, if no flow entry (in the switch's Flow Table) matches the packet's header, then OF switch will send a message to controller to ask for flow entry to process the unknown packets. In its turn, the controller may add, according to the defined policy, a new flow entry to the Flow Table of every switch as required for policy enforcement. Thus, the traffic generated by all hosts connected to a given OF switch will populate the Flow Table of the switch [7]. The procedure that packets are processed is shown in Figure 1 and the fields from packets used to match flow entries are shown in table 1.

B. Collecting Flows Using Floodlight/OpenFlow

The collection of flow entries from an OF switch is performed at predetermined time intervals (in this paper it's 5s) by the controller. From this collection important features are extracted to classify traffic as normal or as an attack. As the collector gathers samples from all OF switches authenticated by Floodlight, the switch ID is used to help the classifier module pinpoint in which OF switches DDoS flooding attacks were detected.

The definition of the time interval to collect flow entries is of great importance. If collection is made at infrequent time intervals, then there will be a delay to detect an attack and consequently a reduction of the time available for a possible mitigation. On the other hand, if the time interval for collection is too short, there will be an increase of packets requesting flows which will lead to an increase in the overhead of our detection mechanism [8].

By default, every OpenFlow switch registered at the controller is automatically added into the detection loop. But the network administrator can also restrict sample collection to those switches that are most relevant. As the controller manages information of network switches in a centralized way, we are able to monitor all selected switches and analyze their traffic from the point of view of a DDoS attack. This analysis is done by the controller in our method.

C. Selecting Features

In this paper, via comparing and analyzing we construct five statistical indexes to establish the decision tree model. Firstly assuming N denotes the number of flow entries in flow table, P_i denotes the number of packets of flow entry F_i, B_i^j denotes the number of Bytes in packet P_i of flow entry F_i. Based on these information, the value of these features can be calculated as follows:

(1) APF (Average of Packets per Flow entry): APF is an important network traffic monitoring index. For example, in DDoS, port scan attacks, the average number of packets per flow entry will decrease or increase dramatically. In traditional network, we can obtain the value of this index directly via some network measurement tools, such as netflow. But netflow is based on sampling, it cannot detect the low-frequency attack and will do harm to the performance of devices.

$$APF = \sum_{i=1}^{N} P_i / N \quad (3)$$

(2) ABF (Average of Bytes per Flow entry): ABF indicates the payload size which are processed by the flow table in one time window. It is often very small when attack like DDoS happens.

$$ABF = \sum_i \sum_j B_i^j / N \quad (4)$$

(3) PPF (Percentage of Pair-Flow-entries): Pair flow entries mean two flow entries which handle the packets of one direction and the reverse. For example flow entry X=(srcIP=A, destIP=B) and flow entry Y=(srcIP=B, destIP=A) are pair flow entries, PPF can be calculated by:

$$PPF = 2 * PF / N \quad (5)$$

PF indicates the number of flow entries which have matched pair flow entry. PPF is a special feature in SDN network. Massive packets without response will result in significant increase of PPF.

(4) SPF (Source IP number of packets matched by flow table): It reflects the variation of source IP number in continuous time unit, this index can measure the faked source IP address in the traffic and is sensitive to spoofing attack. It can be calculated as follows:

$$SPF = \frac{srcIP\_num}{interval} \quad (6)$$

SrcIP_Num is the number of source IPs of packets which are matched by the single flow table in the given time interval.

(5) ADF (Average of Duration per Flow entry): Every flow entry has its own life time. In OF switch, if a flow entry doesn't have packets matched in the given time slot, it will be deleted by the system. As stated in paper [4], this feature can decrease the number of false positives when there is a small number of packets exchanged between applications.

$$ADF = \sum_{i=1}^{N} D_i / N \quad (7)$$

D_i indicates the duration of the i-th flow entry. Abnormal traffic will always lead to the decrease of ADF.

IV. Experiment And Evaluation

For evaluating the proposed method, we established an experimental platform based on MININET, the controller is Floodlight and the server has 64GB memory and 32 core CPU. Firstly, we design a network to simulate real network traffic, the topology is shown in Figure 2.

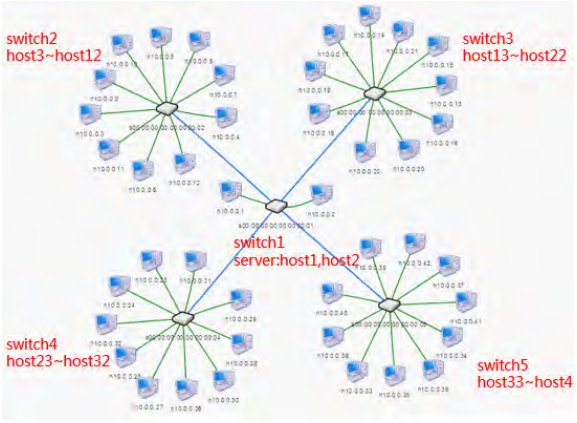

FIGURE II. EXPERIMENTAL SDN NETWORK TOPOLOGY

The simulated network includes 5 OpenFlow switches (s1-s5) and 42 hosts (h1-h42), where h1, h2 serve as servers for other hosts. Hosts can randomly access the servers h1, h2 to simulate normal network activity and generate normal TCP traffic, UDP traffic, and ICMP traffic. Hosts can also use the hping3 tool to initiate SYN Flood, UDP Flood, and ICMP Flood attacks on the servers to simulate abnormal traffic on the network.

For testing the effectiveness of the above five index, we collect data in our simulated network to calculate the value in every time window. The trend of these five indexes is shown as Figure 2-6. From these graphs, we can see that these indexes is sensitive to abnormal network traffic triggered by SYN flood, UDP flood and some other types of DDoS attack.

OpenFlow has many kind of message types, here we mainly use controller-to-switch message. controller can send read-state OpenFlow messages which are used to collect statistics from the switches flow-tables,ports, and the individual flow entries, for example, *OFPST_FLOW* stats request can be used to get the individual Flow Statistics.

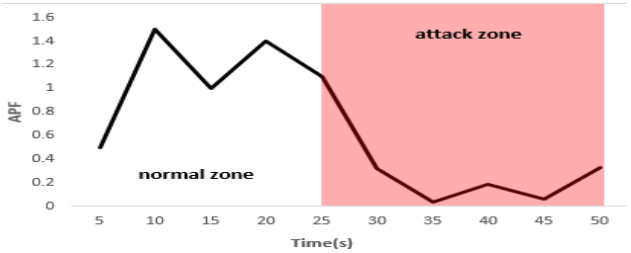

FIGURE III. TREND OF APF IN NORMAL AND ABNORMAL PERIOD (SEC)

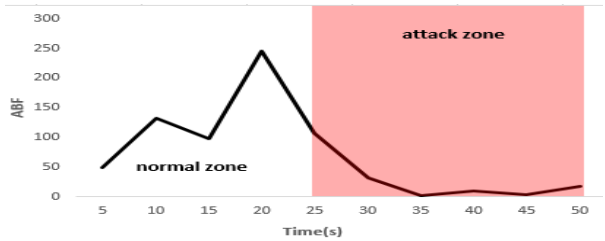

FIGURE IV. TREND OF ABF IN NORMAL AND ABNORMAL PERIOD (SEC)

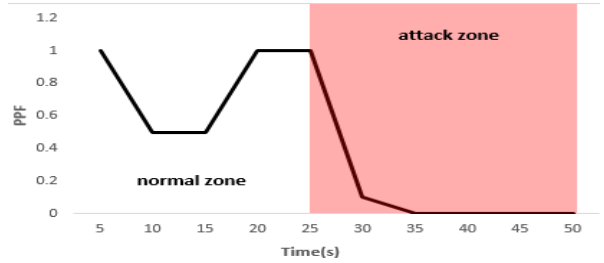

FIGURE V. TREND OF PPF IN NORMAL AND ABNORMAL PERIOD (SEC)

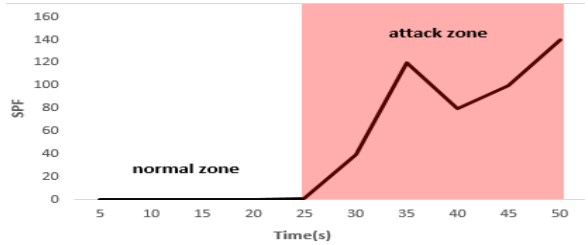

FIGURE VI. TREND OF SPF IN NORMAL AND ABNORMAL PERIOD (SEC)

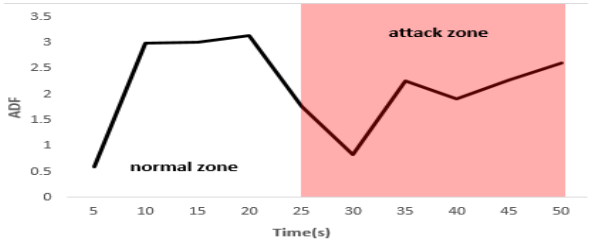

FIGURE VII. TREND OF ADF IN NORMAL AND ABNORMAL PERIOD (SEC)

As shown in Figure 3-Figure 7, when attack begins at 25s, the value of the five statistical indexes changes remarkably. The above experiment verifies the effectiveness of the selected indexes.

For testing the difference of decision tree algorithm and some other machine learning algorithms, we collect 1677 flow entries in normal and abnormal period for training and testing, as shown in Table 2.

TABLE II. TRAINING AND TESTING DATA SET

	Training	Testing	Total
Normal	541	361	902
Abnormal	465	310	775
Total	1006	671	1677

TABLE III. EXPERIMENTAL RESULTS

Algorithm	DR(%)	FR(%)	Consumed time(ms)
DT	95.16	2.49	0.181
KNN	85.48	2.77	2.489
SVM	85.16	3.32	7.465
NB	89.03	1.39	0.206

Decision tree, k nearest neighbor, support vector machine, and naive Bayes algorithms are used to train and test the data sets. The detection rate, false alarm rate, and test time of the four algorithms are shown in Table 3.

From the experimental results, we can see that decision tree has the highest detection rate of 95.16% and the false alarm rate is 2.49%, also it has the lowest test time of 0.181ms. Although Naive Bayes has the lowest false positive rate of 1.39%, its detection rate lower than decision tree dramatically. The detection rate and false alarm rate of k-nearest neighbor and support vector machine are also obviously inferior to the decision tree.

After comparing the classification effects of the four algorithms, we can see that decision tree is the best one to detect abnormal traffic in the network.

V. CONCLUSION AND FUTURE WORK

As to the centralized architecture, SDN is more sensitive to security than traditional network. In this paper we present a lightweight traffic anomaly detection method based on decision tree in SDN network, it can detect the abnormality caused by network attacks with high efficiency.

In the future, we will perform it in real network and train the model with large-scale traffic data. It will be used in dynamic network management and monitoring.

ACKNOWLEDGMENT

This work was Supported by National Key R&D Program of China (2017YFB0801703).

REFERENCES

[1] V. Sommer, Anomoly Detection in the SDN Control Plane, Master's thesis, Technical University of Munich, Munich, Germany, 2014.

[2] Hu F, Hao Q, Bao K. A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation. Communications Surveys & Tutorials IEEE, 2014, 16(4):2181-2206.

[3] Cox J H, Chung J, Donovan S, et al. Advancing Software-Defined Networks: A Survey[J]. IEEE Access, 2017, PP(99):1-1.

[4] Braga R, Mota E, Passito A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. IEEE, Conference on Local Computer Networks. IEEE Computer Society, 2010:408-415.

[5] Ashraf J, Latif S. Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques. Software Engineering Conference. IEEE, 2014:55-60.

[6] Qazi Z A, Lee J, Jin T, et al. Application-awareness in SDN. Computer Communication Review, 2013, 43(4):487-488.

[7] Drutskoy D, Keller E, Rexford J. Scalable Network Virtualization in Software-Defined Networks. IEEE Internet Computing, 2013, 17(2):20-27.

[8] Nanda S, Zafari F, Decusatis C, et al. Predicting network attack patterns in SDN using machine learning approach. Network Function Virtualization and Software Defined Networks. IEEE, 2017.