INTRUSION DETECTION USING NETWORK FLOW FEATURE FOR SOFTWARE-DEFINED NETWORKS

Van Dan Duong[1], *Nam Khanh Tran*[1], *Minh Thanh Ta*[1,*]

Abstract

In recent years, Software-Defined Networking (SDN) is a new network architecture that has been gaining popularity. This promises to simplify network control and management with centralized control of the network, but it also increases the risk of a single point of failure (SPOF) in the network. To mitigate SPOF, more cyber security research is needed on SDN networks. On the other hand, intrusion detection systems (IDSs) play a crucial role in SDN security by dealing with external threats. Machine learning-based IDSs are well-suited for SDN because they can be trained on a centralized controller. However, there is limited research on SDN intrusion detection systems. Existing literature often treats SDN intrusion detection as similar to intrusion detection in traditional computer systems. This approach can be problematic because SDN networks have different characteristics than traditional computer systems. In this paper, we propose a new method for SDN intrusion detection using machine learning. Our method addresses the problem of data imbalance, which is a common problem with machine learning datasets. We also evaluate our method on the most recent public SDN intrusion detection dataset. Our results show that our method can achieve high accuracy and low false alarm rates. Finally, we evaluate the performance of our method in two different SDN scenarios: with and without load balancing. Our results show that our method can achieve high performance in both scenarios.

Index terms

Software-Defined Network, intrusion detection system, machine learning, data augmentation.

1. Introduction

1.1. Overview

The increasing complexity of network technologies, both hardware and software, has made manual network operation and management inefficient and inflexible. To address

[1]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: thanhtm@lqdtu.edu.vn
DOI: 10.56651/lqdtu.jst.v12.n02.752.ict

these challenges, modern networks are adopting automation and softwareization using advanced technologies such as SDN, which is a network architecture that separates the control plane from the data plane. The control plane is responsible for making decisions about how traffic should flow through the network, while the data plane is responsible for carrying out those decisions. This separation of concerns allows for centralized network control, which can improve efficiency, flexibility, and scalability. In addition to centralized control, SDN also offers a number of other benefits, such as:

- Improved security: SDN can be used to implement security policies more easily and effectively.
- Increased visibility: SDN provides a single view of the entire network, which makes it easier to troubleshoot problems.
- Reduced costs: SDN can help to reduce costs by simplifying network operations and management.

As the need for network complexity continues to grow, SDN is becoming an increasingly important technology for modern networks. By automating and softwarizing network operations, SDN can help to improve efficiency, flexibility, scalability, security, visibility, and cost-effectiveness.

Instead, the data plane just forwards the traffic based on the configurations provided by the control plane. The control plane can control and view the network centrally, so it can make smart decisions to manage and operate the network securely. The advantage of SDN is indisputable, and it is increasingly widely used in data centers and enterprises. It is estimated that by 2027, the SDN market could reach 52 billion USD [1].

With a series of advantages of SDN, it is gradually growing in popularity, along with the attention of attackers. The control plane of SDN will control and manage the network centrally. Once attacked, the attacker has the ability to control the entire network, inadvertently turning SDN into an attack target.

By infiltrating the SDN server, they can gain unauthorized access and take away important system information. Attacking SDN's controller is their ultimate goal. As we know, the controller is considered the brain of SDN. They can perform attacks like Denial-of-Service (DoS), Distributed Denial-of-Service (DDoS), Brute Force attacks, and many more.

It is extremely important to detect such attacks quickly so that prevention methods can be implemented. Normally, we would be able to think of IDS [2] as a common way to detect such attacks. More importantly, current IDS mostly use rules to detect attacks that have certain weaknesses, while IDS rely on network traffic analysis and packet characteristics to detect intrusions is showing superiority in terms of advantages. Supposedly, that traffic does not fall under the rules that the network administrator sets for a traditional IDS, but with an IDS using packet analysis, they can still be detected. This improves the quality of detecting malicious packets and attacks on network systems, helping network administrators quickly prevent attacks.

Using Machine Learning (ML) to analyze network packets and build IDS is not too strange. ML proved to be significantly superior in accuracy to traditional IDS. However, previous researchers have not focused on detecting the details of these attacks. This, if detected, will help administrators take appropriate measures to prevent attacks. Furthermore, most datasets they use to train ML algorithms are imbalanced. This leads to learning algorithms biased towards a label that is more visible when training and produces errors when predicting the flow later.

1.2. Our contributions

This paper focuses on developing an IDS for SDN to detect and predict which type of attack. To do that, we propose a new method to get balance the original data set and apply it to ML algorithms to improve accuracy. We believe that our work makes a significant contribution to the field of SDN security and can help to improve the security of SDN networks.

Our contributions to this paper are explained as follows:

- **Create a balanced dataset:** The dataset collected from the wild will have a lot of imbalance, most of which are recorded when there is no attack. This makes the learning models greatly skewed. The fact that the data to train the model is a very important balanced dataset will determine the success of the IDS for SDN. Therefore, our first contribution in this paper can be mentioned as the creation of a balanced SDN network dataset.
- **Discover the best method:** Applying machine learning, experiment, and find suitable methods and algorithms to detect and classify attacked data. Help improve IDS performance.
- **Find a suitable load balancing algorithm:** Evaluate the performance of web servers in SDN with and without load balancing. Find a load-balancing algorithm suitable for the problem. Help improve the performance of the load-balancing system.

1.3. Roadmap

The rest of this paper is arranged as follows. Section 2 discusses the related work and theory analysis. In Section 3, the proposed method algorithm is described in detail. The experimental results and comparisons are shown in Section 4. Finally, a short conclusion is drawn in Section 5.

2. Related work

Intrusion detection is a difficult problem [3], which has been attempted by previous researchers to solve particularly in SDN. Most recent research works focus on applying ML [4]–[6] to build IDS for SDN. Deep Learning (DL) is also used in some research, but not much. In this section, we will discuss some of the previous studies.

In the paper [7], the authors have built an IDS system using ML algorithms on the available data set. They experiment with basic machine learning algorithms. The results show that using ML to build IDS gives relatively good results, reducing dependence on leading experts and replacing the need for a large amount of data to train models. However, the method proposed by the authors has many limitations, not overcoming the inherent weaknesses of the data set as well as improving the efficiency of IDS attack detection. The training dataset is still imbalanced and the accuracy is not high.

In more detail, the paper "InSDN: A Novel SDN Intrusion Dataset" presented IDS using ML in SDN has greatly contributed to generating a relatively large and rich dataset of attacks in SDN [8]. This is the foundation for subsequent researchers to develop highly transparent ML-using IDSs for SDN systems. The authors also tested the generated dataset with basic ML algorithms and achieved very good results with specific detection of each type of attack. However, in this paper, the authors have not solved the imbalance problem and are using a model to detect attacks in binary form, which has not been done at the multi-class level.

In another paper [9], the authors proposed a method to detect early attacks in SDN networks. The method proposed by the authors is to take a set of 9 features and then transform them through CNN to obtain a 128-dimensional feature vector. Then, let that feature vector go through algorithms like XGBoost and Random Forest (RF) for training. The solution method of the author group is relatively good and practical, and the training time is relatively low. However, they can only do it at the level of detecting attacks and not being attacked, not solving the problem of what type of attack. Furthermore, their data imbalance problem has not really been paid attention to and resolved.

There are also some studies [10] on applying ML and DL to IDS problems. They summarize in detail the methods that have been applied, and the technologies applied are relatively new. However, the authors use data sets that are not large enough, and the data imbalance is not resolved.

In the paper "Network intrusion detection in software defined networking with self-organized constraint-based intelligent learning framework" [11] the authors have also outlined a number of ways to use ML to detect network intrusions, but the problem they focus on solving is to reduce the error when the model predicts as well as to avoid problems such as overfitting. The paper also has many advantages, however, the solution proposed by the authors still has many limitations.

The application of Artificial Intelligence (AI) to systems is gradually gaining popularity. There are many research works related to network intrusion detection using AI. A group of authors synthesized methods to apply AI network intrusion detection in SDN [12]. However, the paper has not yet provided solutions for the weaknesses of the existing works.

In the paper "Survey on SDN based network intrusion detection system using machine learning approaches" [13], the authors have synthesized methods of network intrusion detection using Machine Learning. The paper also mentioned tools that can develop Network-based Intrusion Detection System (NIDS) in a real environment. The author group has surveyed and presented the most recent research projects. However, the methods they came up with have not yet solved the problem of data imbalance.

Applying DL is also a very positive direction. There are many groups of authors who have applied different deep learning methods to realize that. In the paper "A novel approach to network intrusion detection system using deep dearning for sdn: futuristic approach" [14] the authors used 12 features out of 41 features in the NSL-KDD dataset by the feature selection method. The results of the paper are relatively good and capable of practical implementation. However, the paper uses a general data set for the IDS system, and it has not been specialized for SDN.

With previous studies, we propose a new method. Our method proposes to balance the data and detect the details of each type of attack.

3. Our proposed method

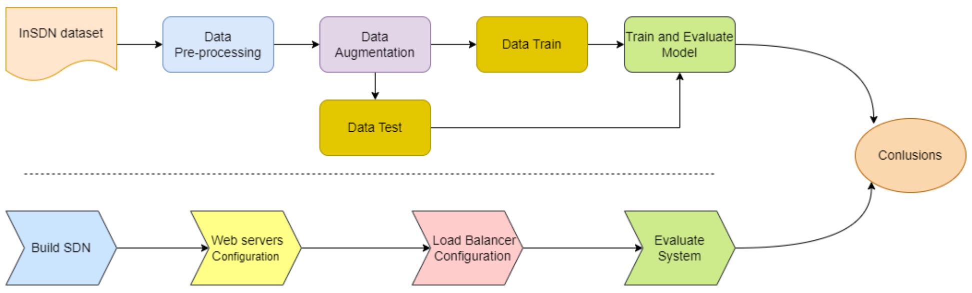

Fig. 1. Pipeline for experiments of this paper

In traditional networks, IDSs are often placed behind firewalls. IDS usually gets data from the SPAN port of the switch so that it can take the data exchanged in the network and make predictions. In SDN, all traffic network is passed through the SDN controller. Therefore, we recommend placing the IDS behind the firewall and getting the traffic network from the SDN controller. In this paper, the pipeline we perform experiments is shown in Fig. 1. Our proposed IDS is set up as shown in Fig. 2. After training the model, ML obtained the model with the best classification ability. We apply that model to network intrusion detection for SDN. Network traffic will go from the client through the SDN controller and to the server. When going through the SDN controller, the network traffic is routed through the IDS. At this point, if IDS detects an anomaly and classifies it as intrusive, an alarm is issued.

In this section, we focus on solving data imbalance problem of InSDN dataset [7]. Then, conduct experiments with ML algorithms to evaluate the network intrusion detection results before and after data equalization.

In an equivalent experiment, we setup networks with Mininet[1] for virtual environment

[1]http://mininet.org/

networks and Ryu[2] for the controller of the networks. We also configure load balancing to compare the performance of the system with and without a load balancer.

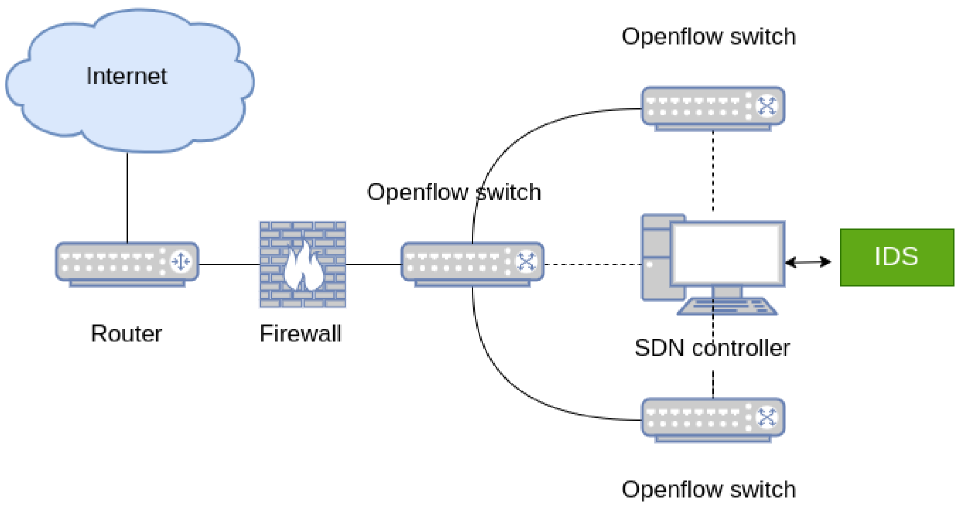

Fig. 2. IDS location in SDN

3.1. Proposed approach

Data imbalance is one of the big problems in ML [15], where the majority class (the class that is most common in the data) is much more prevalent than the minority class (the less common class). This can lead to ML models that are biased towards the majority class and can make it difficult to detect minority class instances. There are many ways to deal with data imbalance, such as down-sampling, over-sampling, cost-sensitive learning, and so on. However, using down-sampling will reduce the data significantly. This may lead to missing training data for the model.

In this paper, we propose a method to balance the data using over-sampling. Using Synthetic Minority Oversampling Technique (SMOTE) [16] to be precise. It was described by Nitesh Chawla *et al.* published in the paper “SMOTE: synthetic minority over-sampling technique.” [17] in 2002. SMOTE operates by identifying instances that are proximate in the feature space [18], establishing a connection between these instances, and generating a fresh sample along that connection.

We designed and built a basic SDN using Mininet and Ryu controllers. This servers to evaluate the performance of the SDN system with load balancing and without system load balancing.

3.2. Data pre-processing

We proceed to clean the existing dataset. Any records with the "Nan" field will be deleted. We then proceed to solve the data imbalance problem by using methods such as

[2]https://ryu-sdn.org/

over-sampling and down-sampling. However, we found that the down-sampling method lost extreme data, so we did not continue to use this method in our experiments. Then, split the dataset into train, test, and validate sets at the rate of 80, 10, and 10.

3.3. Data augmentation

SMOTE is a ML method designed to address class imbalance in classification tasks. It works by generating synthetic instances for the minority class, using interpolation between existing minority class data points. This helps balance the class distribution and improve the performance of classification algorithms on the minority class. For SMOTE data augmentation method, the features of the generated data will be different from the original data, but a similar proportionality will be maintained. The process of creating synthetic samples ensures that the features of the new samples accurately reflect the relationship between the features of the original sample and its nearest neighbors.

For each feature in the original sample, a neighboring sample is chosen, and a new sample is created by adding a fraction of the difference between the neighboring sample's feature and the original sample's feature to the original sample's feature.

In this study, we employ SMOTE [18] as a means to enhance the current dataset. By leveraging SMOTE, our aim is not only to expand the data but also to establish a balanced representation across different devices. To achieve this, we utilize the largest number of device samples and enhance SMOTE to generate an equivalent number of samples for the remaining devices [19].

Our approach consists of two sequential steps. Initially, we employ the Imblearn library [20], specifically utilizing the over-sampling technique, to augment the dataset. Subsequently, we apply this method to our specific dataset and achieve comparable outcomes as Fig 3.

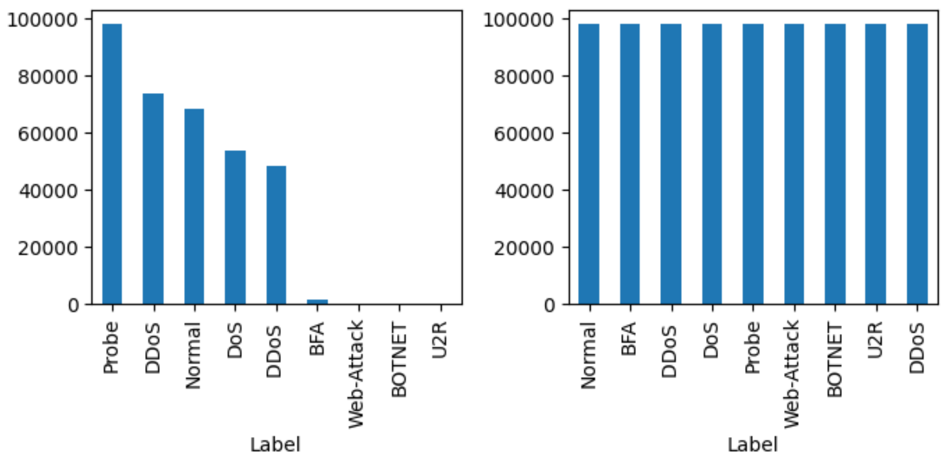

Fig. 3. Before and after using data augmentation

4. Results and comparison

4.1. Experimental environment

Our experimental environment is on a PC with the following configuration: Intel i7-10700f CPU, 32 GB RAM, and 16GB VRAM RTX A4000 VGA card. We choose Python language and jupyter notebook platform to perform the experiments. When we train the model, we use Python version 3.6.13, sklearn 0.24.2, tensorFlow 2.6.2, numpy 1.19.2, pandas 1.1.5, and some other tools to analyze and train the model in the experiment.

In this paper, we use some popular ML algorithms for testing such as Random Forest (RF), XGBoost (XGB), LightGBM(LGBM), and some others. They require the use of relatively different libraries. RF belongs to sci-kit learn library, XGB belongs to XGBoost library and LGBM belongs to lightgbm library.

4.2. Dataset preparation

In this section, we describe the dataset that we use to train our ML algorithms. We use the InSDN dataset [8], a relatively large dataset created in 2020. This dataset contains many different types of SDN network data. Including data on various attacks such as Dos, DDos, Web-attack, and many other types of attacks.

This dataset includes 343889 different records of network traffic in SDN. It consists of 84 columns of which 83 columns are network features and 1 column is label. Probe attacks accounted for the largest share up to 98129 records, and the U2R attacker was the smallest with 17 records.

As we can see, this is a relatively unbalanced dataset. The difference in records between labels is relatively large. We solve the data imbalance problem by using over-sampling with the SMOTE method. The results after using we get a balanced dataset as Fig 3.

4.3. Result of machine learning application in intrusion detection

We tested in turn with algorithms such as Decision Tree, RF, XGB, LR, and LGBM. First, we experiment with the original data set. Next, we experiment with the balanced and enriched data set.

The algorithms we use are from libraries such as sklearn, lightGBM, and XGBoost. They were all tested with the default parameters provided by the respective libraries. Throughout the experimentation process, we did not alter any default parameters for the algorithms within these libraries.

Looking at table 1, and Fig. 4 we see that the results when not applying the method to balance the data and after applying there are obvious differences. Before applying, the data set was unbalanced, so indexes like F1-score were very low, and there was a

Table 1. Results using the methods

Method	Accuracy	Precision	Recall	F1-score
Decision Tree	0.9755	0.9311	0.9457	0.9398
Random Forest	0.9915	0.9604	0.9815	0.9704
XGBoost	0.9833	0.9645	0.9890	0.9835
Logistic Regression	0.6755	0.5545	0.6208	0.6932
LightGBM	0.8156	0.7839	0.7943	0.7108
Decision Tree + Aug	0.9972	0.9955	0.9912	0.9983
Random Forest + Aug	**0.9995**	0.9932	0.9967	0.9936
XGBoost + Aug	0.9988	0.9962	**0.9990**	**0.9986**
Logistic Regression + Aug	0.6955	0.6549	0.6188	0.6898
LightGBM + Aug	0.9954	**0.9987**	0.9943	0.9981
InSDN [8] + Random Forest	-	0.9942	0.9969	0.9955
NIDS-DL [14] + CNN	0.9863	0.9845	0.9898	0.9872

certain difference between the indexes. After applying, we see the results are improved a lot, and the F1-score is improved a lot.

The algorithm for the highest accuracy is RF with 99.95%. However, the XGB algorithm has the largest Recall and F1-scores at 99.90% and 99.86%, respectively. This index reflects the learning level of the model with the unbalanced data set. With the current stats showing, the XGB algorithm is doing the best.

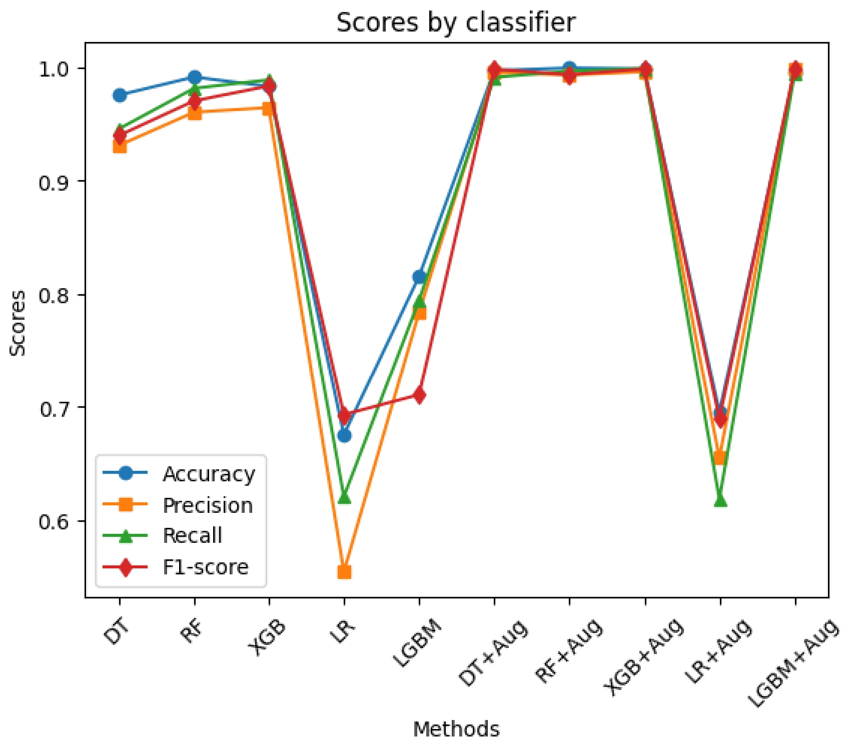

Fig. 4. Scores of the methods

4.4. Evaluation of software-defined networking system with load balancer

Managing system resources is critically important as it directly impacts the optimization of system performance, particularly in computer networks and distributed systems. Load balancing systems help optimize servers and system resources, preventing overload and imbalance, thus enhancing the overall operational efficiency of the system.

In this section, we have built a simple SDN network. By using Mininet and Ryu controller, we can easily build a simple SDN network. We use the simplest topology, star, to design the network. The network consists of 10 clients, three web servers, one switch, and one controller.

Setup three web servers to open port 80, providing HTTP services. With the web server installed as Apache. Program the Ryu controller to test all 3 cases: non-load balancer, load balancer with round robin, and load balancer with weight round robin. With weight round robin we set the weights for servers 1,2,3 to be 1,3,5, respectively.

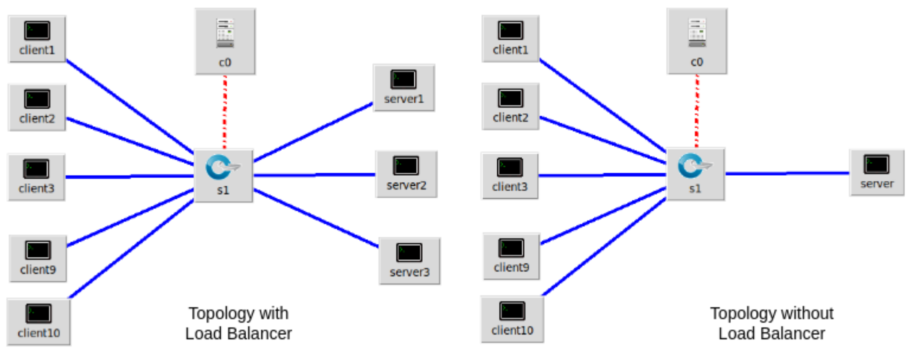

Fig. 5. Topology of SDN

We measure the response time of the system in all three experiments. To perform the measurement, we set up the network topology, as shown in Fig. 5. We used Mininet and Ryu controller to build the network. For the load balancer experiment, we programmed the Ryu controller to coordinate the traffic to the respective servers. With the round robin algorithm is 1,1,1 and with the weight round robin is 1,3,5.

Table 2. System response time

Method	max	avg	min
Non-LB	23.8ms	13.2ms	0.5ms
LB-RR	**13.8ms**	**7.2ms**	**0.4 ms**
LB-WRR	15.3ms	7.5ms	0.51ms

As shown in table 2, the response time of LB-RR is smaller than the others. The response time of the system with load balancing is comparatively much better than the system without load balancing. In particular, the round robin algorithm proved to be superior when there were significantly better coefficients. The fastest response time was 0.4 ms, and the longest was 23.8 ms.

Based on the research findings, we propose an SDN system utilizing load balancing and constructed with the Round Robin algorithm. This system can be configured with any server and network topology, ensuring adaptability to the system architecture. In practical scenarios, if servers have different configurations, the weight round robin algorithm should be employed to ensure optimal system performance.

5. Conclusions

SDN is a relatively new and highly applicable field. However, it also has many potential security problems when centralizing control in the controller. The study of network intrusion detection methods is extremely necessary for the system.

In this paper, we have proposed a ML-based IDS for SDN using network traffic. Our method achieved high accuracy and low false alarm rates in both SDN scenarios. We believe that our method can be used to improve the security of reality SDN networks. On the other hand, we also propose a method to solve the data imbalance problem. This is a relatively complex problem and our solution has yielded very good results. At the same time, we also evaluated the performance of the SDN network with and without the load balancing system.

In the future, we plan to study DL models for SDN intrusion detection. We will investigate the use of different DL models for SDN intrusion detection. We will also explore the use of transfer learning, which is the practice of using a model trained on one task to solve a different task. In addition, we will work with some companies to test the actual implementation of our model on their network. This will allow us to evaluate the performance of our model in a real-world setting. We believe that our work has the potential to make a significant contribution to the field of SDN security. By studying deep learning models and testing the actual implementation of our model on a given company, we can develop a more effective IDS for SDN networks.

References

[1] J. A. Sava, “Software-defined networking (SDN) market size worldwide from 2021 to 2027,” Available at https://www.statista.com/statistics/468636/global-sdn-market-size/ (2023/25/5).

[2] A. Chawla, B. Lee, S. Fallon, and P. Jacob, “Host based intrusion detection system with combined CNN/RNN model,” in *ECML PKDD 2018 Workshops*, C. Alzate, A. Monreale, H. Assem, A. Bifet, T. S. Buda, B. Caglayan, B. Drury, E. García-Martín, R. Gavaldà, I. Koprinska, S. Kramer, N. Lavesson, M. Madden, I. Molloy, M.-I. Nicolae, and M. Sinn, Eds. Cham: Springer International Publishing, 2019, pp. 149–158. ISBN 978-3-030-13453-2

[3] Y. Al-Nashif, A. A. Kumar, S. Hariri, Y. Luo, F. Szidarovsky, and G. Qu, “Multi-level intrusion detection system (ML-IDS),” in *2008 International Conference on Autonomic Computing*, 2008, pp. 131–140. Doi: 10.1109/ICAC.2008.25

[4] S. S. Gopalan, D. Ravikumar, D. Linekar, A. Raza, and M. Hasib, “Balancing approaches towards ML for IDS: A survey for the CSE-CIC IDS Dataset,” in *2020 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2021, pp. 1–6. Doi: 10.1109/ICCSPA49915.2021.9385742

[5] A. S. Dina and D. Manivannan, “Intrusion detection based on machine learning techniques in computer networks,” *Internet of Things*, vol. 16, 2021, p. 100462. Doi: 10.1016/j.iot.2021.100462

[6] P. Parkar and A. Bilimoria, “A Survey on Cyber Security IDS using ML Methods,” in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2021, pp. 352–360. Doi: 10.1109/ICICCS51141.2021.9432210

[7] M. Almgren and E. Jonsson, “Using active learning in intrusion detection,” in *Proceedings. 17th IEEE Computer Security Foundations Workshop*, 2004, pp. 88–98. Doi: 10.1109/CSFW.2004.1310734

[8] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, “InSDN: A Novel SDN Intrusion Dataset,” *IEEE Access*, vol. 8, 2020, pp. 165 263–165 284. Doi: 10.1109/ACCESS.2020.3022633

[9] M. S. Towhid and N. Shahriar, “Early Detection of Intrusion in SDN,” in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, 2023, pp. 1–6. Doi: 10.1109/NOMS56928.2023.10154272

[10] J. Hussain and V. Hnamte, “Deep learning based intrusion detection system: Software defined network,” in *2021 Asian Conference on Innovation in Technology (ASIANCON)*, 2021, pp. 1–6. Doi: 10.1109/ASIANCON51346.2021.9544913

[11] A. Bhardwaj, R. Tyagi, N. Sharma, A. Khare, M. S. Punia, and V. K. Garg, “Network intrusion detection in software defined networking with self-organized constraint-based intelligent learning framework,” *Measurement: Sensors*, vol. 24, 2022, p. 100580. Doi: 10.1016/j.measen.2022.100580

[12] S. Dahiya, V. Siwach, and H. Sehrawat, “Review of ai techniques in development of network intrusion detection system in sdn framework,” in *2021 International Conference on Computational Performance Evaluation (ComPE)*, 2021, pp. 168–174. Doi: 10.1109/ComPE53109.2021.9752430

[13] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, “Survey on sdn based network intrusion detection system using machine learning approaches,” *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, Mar 2019, pp. 493–501. Doi: 10.1007/s12083-017-0630-0

[14] M. R. Hadi and A. S. Mohammed, “A novel approach to network intrusion detection system using deep learning for SDN: Futuristic approach,” in *Machine Learning &amp Applications*. Academy and Industry Research Collaboration Center (AIRCC), Jun 2022. Doi: 10.5121/csit.2022.121106

[15] S. Ray, “A quick review of machine learning algorithms,” in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019, pp. 35–39. Doi: 10.1109/COMITCon.2019.8862451

[16] P. Jeatrakul, K. W. Wong, and C. C. Fung, “Classification of imbalanced data by combining the complementary neural network and smote algorithm,” in *Neural Information Processing. Models and Applications*, K. W. Wong, B. S. U. Mendis, and A. Bouzerdoum, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 152–159. ISBN 978-3-642-17534-3

[17] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” *J. Artif. Intell. Res. (JAIR)*, vol. 16, 06 2002, pp. 321–357. Doi: 10.1613/jair.953

[18] G. Douzas and F. Bacao, “Geometric smote a geometrically enhanced drop-in replacement for smote,” *Information Sciences*, vol. 501, 2019, pp. 118–135. Doi: 10.1016/j.ins.2019.06.007

[19] Z. Wang, J. Hu, G. Min, Z. Zhao, and J. Wang, “Data-augmentation-based cellular traffic prediction in edge-computing-enabled smart city,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, 2021, pp. 4179–4187. Doi: 10.1109/TII.2020.3009159

[20] G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning,” *Journal of Machine Learning Research*, vol. 18, no. 17, 2017, pp. 1–5.

Manuscript received 24-10-2023; Accepted 21-12-2023.

■

Van Dan Duong is a cadet of Le Quy Don Technical University in Vietnam. His research interests lie in the area of deep learning, steganography, and computer vision.
E-mail: duongvandan2806@gmail.com

Nam Khanh Tran obtained an impressive IT engineering degree from Le Quy Don Technical University in Vietnam in 2020. He has since been contributing to the academic community as a lecturer at the same university, specializing in the research fields of cybersecurity, technology network, and image processing. E-mail: khanhtn107195@lqdtu.edu.vn

Minh Thanh Ta is currently an associate professor and vice dean of Institute of Information and Communication Technology in Le Quy Don Technical University, Vietnam. He is also a Postdoctoral Fellow of the Department of Mathematical and Computing Sciences at Tokyo Institute of Technology.He received his B.S. and M.S in Computer Science from National Defense Academy, Japan, in2005 and 2008 and his Ph.D. from Tokyo Institute of Technology, Japan, in 2015, respectively. He is a member of IPSJ Japan and IEEE. His research interests lie in the area of watermarking, network security, and computer vision.
E-mail: thanhtm@lqdtu.edu.vn

PHÁT HIỆN XÂM NHẬP SỬ DỤNG TÍNH NĂNG LUỒNG MẠNG CHO MẠNG ĐƯỢC XÁC ĐỊNH BẰNG PHẦN MỀM

Dương Văn Dần, Trần Nam Khánh, Tạ Minh Thanh

Tóm tắt

Trong những năm gần đây, Mạng được xác định bằng phần mềm (SDN) là một kiến trúc mạng mới đang trở nên phổ biến. Điều này hứa hẹn sẽ đơn giản hóa việc kiểm soát và quản lý mạng với khả năng kiểm soát mạng tập trung, nhưng nó cũng làm tăng nguy cơ xảy ra lỗi một điểm (SPOF) trong mạng. Để giảm thiểu SPOF, cần có thêm nghiên cứu về an ninh mạng trên mạng SDN. Mặt khác, hệ thống phát hiện xâm nhập (IDS) đóng một vai trò quan trọng trong bảo mật SDN bằng cách xử lý các mối đe dọa bên ngoài. IDS dựa trên máy học rất phù hợp với SDN vì chúng có thể được huấn luyện trên bộ điều khiển tập trung. Tuy nhiên, nghiên cứu về hệ thống phát hiện xâm nhập SDN còn hạn chế. Các tài liệu hiện có thường coi việc phát hiện xâm nhập SDN tương tự như phát hiện xâm nhập trong các hệ thống máy tính truyền thống. Cách tiếp cận này có thể có vấn đề vì mạng SDN có những đặc điểm khác với hệ thống máy tính truyền thống. Trong bài báo này, chúng tôi đề xuất một phương pháp mới để phát hiện xâm nhập SDN bằng cách sử dụng máy học. Phương pháp của chúng tôi giải quyết vấn đề mất cân bằng dữ liệu, đây là một vấn đề phổ biến với bộ dữ liệu học máy. Chúng tôi cũng đánh giá phương pháp của mình trên tập dữ liệu phát hiện xâm nhập SDN công khai gần đây nhất. Kết quả của chúng tôi cho thấy phương pháp này có thể đạt được độ chính xác cao và tỷ lệ cảnh báo sai thấp. Cuối cùng, chúng tôi đánh giá hiệu suất của phương pháp trong hai kịch bản SDN khác nhau: có và không có cân bằng tải. Kết quả này cho thấy phương pháp của chúng tôi có thể đạt được hiệu suất cao trong cả hai kịch bản.

Từ khóa

Mạng được xác định bằng phần mềm, hệ thống phát hiện xâm nhập, học máy, tăng cường dữ liệu.