Anomaly-based Intrusion Detection and Prevention Using Adaptive Boosting in Software-defined Network

by Dessyanto Boedi Prasetyo

Submission date: 03-Jul-2020 12:48AM (UTC+0700)
Submission ID: 1352677818
File name: perwira2019.pdf (297.34K)
Word count: 2933
Character count: 15070

2019 5th International Conference on Science in Information Technology (ICSITech)

Anomaly-based Intrusion Detection and Prevention Using Adaptive Boosting in Software-defined Network

Rifki Indra Perwira
Dept. of Informatics Engineering
UPN "Veteran" Yogyakarta
Yogyakarta, Indonesia
rifki@upnyk.ac.id

Yuli Fauziah
Dept. of Information System
UPN "Veteran" Yogyakarta
Yogyakarta, Indonesia
yuli.if@gmail.com

I Putu Retya Mahendra
Dept. of Informatics Engineering
UPN "Veteran" Yogyakarta
Yogyakarta, Indonesia
retya.mahendra@gmail.com

Dessyanto Boedi Prasetyo
Dept. of Informatics Engineering
UPN "Veteran" Yogyakarta
Yogyakarta, Indonesia
dess95@gmail.com

Oliver Samuel Simanjuntak
Dept. of Information System
UPN "Veteran" Yogyakarta
Yogyakarta, Indonesia
oliver.simanjuntak@upnyk.ac.id

***Abstract*—Anomaly-based intrusion detection and prevention technique is a technology needed in a software-defined network (SDN). The change in the SDN paradigm into a centralized architecture causes one side of weakness, namely vulnerability from denial of service (DoS) attacks. A large amount of data requested from clients to servers in a short time can be used as prediction data using decision stump to produce learning data. Learning data that have been formed will be used to make predictions. This research aims to detect and prevent DoS attacks using an anomaly-based adaptive boosting algorithm. The experimental test results obtained in this paper show that the effectiveness of the adaptive boosting algorithm in detecting attacks reaches 93.3% and can deny access in real-time. The conclusion is that the adaptive boosting algorithm can be used in building the Intrusion detection and prevention system.**

***Keywords*—adaptive boosting, intrusion detection, software-defined network, DoS**

I. INTRODUCTION

Network security, especially in software-defined network (SDN) schemes, is a new paradigm in designing, managing, and implementing systems and separating between the control plane and the forwarding plane [1]. This separation is carried out by the OpenFlow protocol which is a standard protocol on SDN. This separation has the potential to be a source of denial of service (DoS) attacks in the SDN network concept. DoS attacks will consume network resources by making requests in substantial and continuous volumes [2]. This attack consists of a collection of packets received by the switch and sent to the controller to fill the resources so that the network becomes full [3].

The importance of security in SDN is one of the exciting aspects to be reviewed and developed, including the security of intrusion detection in SDN using genetic algorithms with an accuracy of up to 90% [4]. Also, intrusion detection with an ensemble boosting approach uses data mining [5]. The pattern of DoS attacks that tends to send packets to consume resources has an unusual nature. Anomalies take advantage of habits that occur in the system and assume that if there are deviations from the patterns that happen, it can be expressed as an attack [6]. This anomaly technique makes the detection process faster and has more effective prevention considering the detected attack as a learning process [7]. There are two approaches to detect attacks, namely the rule-based approach and the anomaly-based approach [8]. The weakness of the rule-based approach is that if an incoming attack is unprecedented, it cannot detect the latest attack; whereas the anomaly-based approach exploits the habits that occur and considers that if there is a deviation from the usual pattern, then it is declared as an attack activity [9]. Therefore, this paper uses an anomaly-based approach.

In previous studies, adaptive boosting has been used to improve denoising performance that depends on noise levels and can also store edge and detailed information on images [10] as well as improve the accuracy of face detection on fingerprint [11]. Adaptive boosting is also able to increase the level of accuracy of Naive Bayes in detecting network detection intrusions [12]. The main contributions of this paper are:

1) To propose an adaptive boosting algorithm for attack detection systems on DoS.

2) To prove that the adaptive boosting algorithm can improve the accuracy of attack detection from a series of learning data generated through decision stumps.

3) To prove that the adaptive boosting algorithm can be used to build intrusion detection and prevention systems with high accuracy.

This paper is organized as follows: Section 2 presents experimental data flow, adaptive boosting, and decision stump models. Section 3 discusses iteration calculations, final predictions, and experimental results. Summary and conclusion are presented in Section 4.

II. METHODS

A. Observation

This paper uses experimental data obtained from controllers in the form of packets sent by attackers in real-time. Observation is done by collecting logs from incoming packets. The attributes that will be captured on SDN are shown in Table 1.

978-1-7281-2380-6/19/$31.00 ©2019 IEEE

TABLE I. FLOW STATISTIC

No.	Features	Description
1	datapath	The main network interface
2	in_port	Incoming Port
3	eth-dst	Ethernet connection
4	out-port	Outgoing port
5	packet	Number of packets
6	bytes	Number of bytes
7	ipv4_src	Source IP address
8	Ipv4_dst	Destination IP address

There are eight types of statistical flow attributes used. Packets and bytes are the two main data attributes that are used for the prediction process.

B. Data Flow Statistics

The final prediction in the form of an attack or not on the adaptive boosting process requires a dataset to produce a model. The dataset should be initially through the learning process. Table 2 is an example of learning data taken from the flow statistics of experimental results using hping3 tools.

TABLE II. DATA TRAIN FLOW STATISTIC

Data	Number of packets (x_1)	Number of bytes (x_2)	Label	Early prediction	W
1	31	2926	No	1	0,1
2	30	2884	No	1	0,1
3	33	3066	No	1	0,1
4	29	2776	No	1	0,1
5	19	1750	No	1	0,1
6	271800	47292936	Yes	-1	0,1
7	1106210	192480408	Yes	-1	0,1
8	1102767	191881194	Yes	-1	0,1
9	29777	5181066	Yes	-1	0,1
10	26	2436	No	1	0,1

In Table 2, data with several bytes less than 65536 bytes are categorized as an ordinary packet [13]. Data with normal conditions are given an initial prediction 1 meaning that it is not an attack and -1 meaning an attack.

C. Adaptive Boosting

The adaptive boosting algorithm gives more weight to improper observations (weak classification). The decision stump formulation can be described in (1) [14].

$$f(x) = s(x_k > c) \quad (1)$$

The value of the function f (x) will produce a prediction of value 1 if the value of element k of the vector x is higher than c (threshold), and will be set to -1 if vice versa. The number s is 1 or -1 which will produce two functions f (x), namely $x_k >$ c and $x_k \leq$ c. After that, all predictions are then combined, and the majority of votes (number) are cast to produce final prediction. Boosting iteration consists of applying weighting t = 1... T for each learning sample in (2) through (5) [15].

Given : $(x_1, y_1),\ldots..,(x_m, y_m)$ where $x_i \in$ X, $y_i \in$ Y = {-1,+1}

Inisialization $D_1(i) = \frac{1}{n}$, where n = amount of data

For t =1 to T:

Train base learner with distribution D_t

Get a weak hypothesis h_t : X → {-1,+1} with *error*

$$\epsilon_t = Pr_{i\sim D_t}[h_t(x_i) \neq y_i] \quad (2)$$

$$\alpha_t = \frac{1}{2}\lambda\nu(\frac{1-\epsilon_t}{\epsilon_t}) \quad (3)$$

Update:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & jika\ h_t(x_i) = y_i \\ e^{\alpha_t}, & jika\ h_t(x_i) \neq y_i \end{cases} \quad (4)$$

$$= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

The Zt function is a normalization factor (it is chosen so that D $_{(t+1)}$ will be distributed), so we get the final output in the equation below:

$$H(x) = \text{sign}\left(\sum_{t=0}^{T} \alpha_t h_t(x)\right) \quad (5)$$

Adaptive boosting takes a predictive approach by modeling iterations and adjusting a series of weak learners [15]. Adaptive boosting workflow is shown in Figure 1.

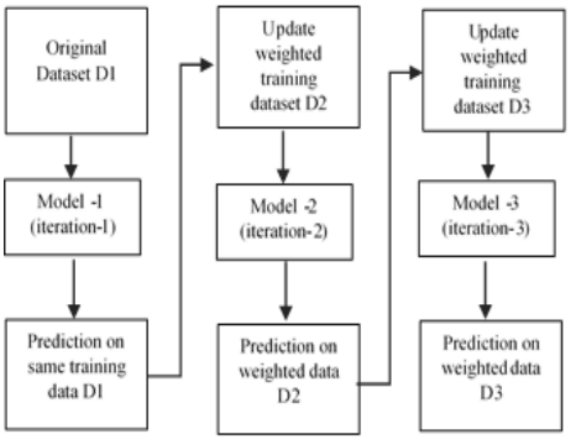

Fig. 1. The workflow of Adaptive Boosting [16]

In this paper, the iteration process is only up to two times. The iteration process can also be repeated up to three times [14]. If more iterations are performed, the best value can be obtained. Even though the iteration has reached 100, it can reach 1000 if needed [17].

D. Decision Stump Model

Decision stump is suitable for heterogeneous data in which different prediction models are more suitable for different regions [18]. Decision stump is needed to get the threshold value. Threshold value (T) can be calculated by (6).

$$T = \text{data}(x_1 \text{ or } x_2) * \text{index data} + \text{step}(x_1 \text{ or } x_2) \quad (6)$$

The threshold value formed will be compared with the data value at the index n to get a prediction. The prediction is learning data that will be used by adaptive boosting to do the next iteration. Figure 2 shows a decision stump of threshold value comparison.

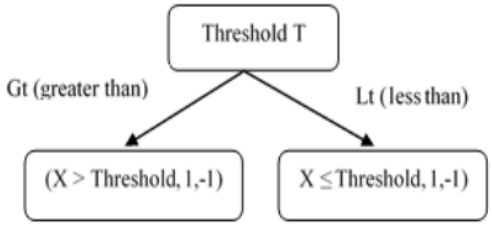

Fig. 2. Decision Stump [15]

Threshold values for each data are calculated for indices -1 through 10 with a total of 10 data so that there are 22 thresholds formed. Predictions using a decision stump can be found in the next step with 2 ways of prediction functions:

$$\text{Gt(greater than)} = \text{if label 1 than } (X > T, 1, -1) \quad (7)$$

$$\text{Lt(less than)} = \text{if label} - 1 \text{ than } (X \leq T, 1, -1) \quad (8)$$

III. Results and Discussion

Preprocessing data is taken from the statistical flow in Table 2 that will be used as learning data. After preprocessing, the data normalization process will be carried out because adaptive boosting will only classify the data into two labels, namely 1 for non-attacks and -1 for attacks. All data are given the same weight (W) at the beginning of the model formation.

A. Threshold

Decision stump as weak learn will produce threshold values of all data used as threshold values for initial predictions. The decision stump method can only work for data with only two predictive labels (1, -1). In Figure 3, the distribution of 22 thresholds is formed which is created from calculation using equation (6).

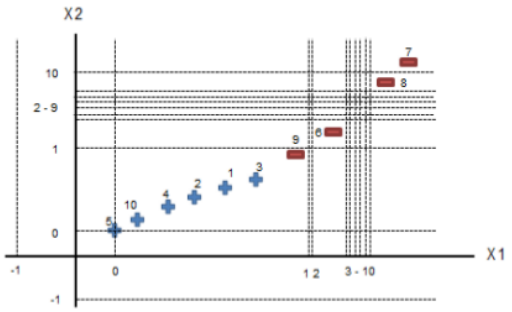

Fig. 3. Threshold Values

Positive symbol (+) means that the data have a reasonable byte value, whereas negative (-) indicates that the data have a byte value of more than 65536.

B. Calculation of the First Iteration

The threshold value using equation (6) for data index x_1 - 1 is -110600.1. Data with these threshold values are used to calculate predictions from each data, as shown in Figure 4.

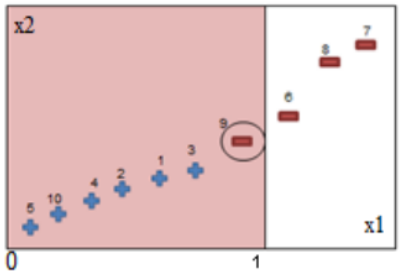

Fig. 4. First Iteration Threshold

Ten experimental data that became learning data turned out to have one datum that had the smallest number of errors, namely the ninth data ($\alpha 1 = 1.09$), so that equation (8) applies to get the prediction. The results of the threshold value can be seen in Table 3.

TABLE III. PREDICTION OF ERROR LESS THAN (LT)

Data	Number of packets (x1)	Label	Current Prediction	Error
1	31	1	-1	0,1
2	30	1	-1	0,1
3	33	1	-1	0,1
4	29	1	-1	0,1
5	19	1	-1	0,1
6	271800	-1	-1	0
7	1106210	-1	-1	0
8	1102767	-1	-1	0
9	29777	-1	-1	0
10	26	1	-1	0,1
Total error				0,6

The predicted value of each iteration will vary depending on the learning process of adaptive boosting to the threshold value with the data.

C. Calculation of the Second Iteration

The second iteration process is the same as the process in the first iteration. In the second iteration, the smallest error value occurs in data no 5. According to Figure 5, the fifth data position is in the far left of the coordinate plane so that the equation greater than (7) applies to find the current prediction.

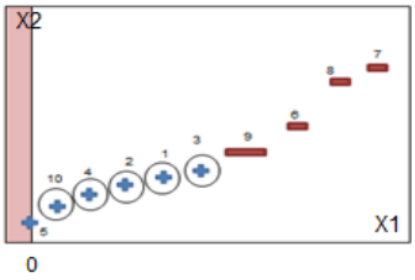

Fig. 5. Second Iteration Threshold

Based on the ten data used as learning data, it turns out that the fifth data (α_2 = 0.47) becomes a reference to look for predictions greater than (Gt) contained in Table 4.

TABLE IV. PREDICTION OF ERROR GREATER THAN (GT)

Data	Number of packets (x_1)	Label	Current Prediction	Error
1	31	1	1	0
2	30	1	1	0
3	33	1	1	0
4	29	1	1	0
5	19	1	-1	0,1
6	271800	-1	1	0,1
7	1106210	-1	1	0,1
8	1102767	-1	1	0,1
9	29777	-1	1	0,1
10	26	1	1	0
Total error				0,5

The threshold value in the second iteration obtained from equation (6) is 19. The prediction results in Table 4 can be calculated using equation (7).

D. Final Prediction

From the results of the first and second iteration in the previous stage, the threshold value for the first iteration index 1 is 110638.1 for functions less than (Lt), and the second iteration index 0 for function greater than (Gt) the threshold value is 19. Prediction of attack and not attack (h_1, h_2) on the Gt and Lt functions is calculated using equation (7) and (8). In the first iteration of the function (Lt), the prediction is positive (not attack) for any data; whereas in the second iteration of the function (Gt), the prediction is negative for any data (attack).

The voting process from a set of iterations that have been done is the final process of the adaptive boosting algorithm. The voting process is done using equation (5). This process is the final prediction of a set of datasets from the iteration results. Final calculation:

= sign (α1. (h1) + α2. (h2) = sign (1,09.1) + (0.47.-1) = 0.62 (not attack). The conclusion of the voting process shows that with the number of packets 33 and the number of bytes 3066 (data number three), Table 2 is categorized as not attack.

E. Confusion Matrix Test

In this paper, a confusion matrix is used to test the accuracy of the adaptive boosting algorithm. Algorithms can be applied in various topologies because the features used do not depend on the topology. All features in the topology are processed on the controller. But in this confusion matrix, the test uses a star topology with one controller, three switches, one attacker and 25 hosts. In this test, a correction process is carried out on the label with the results of the classification produced by adaptive boosting, which is implemented into IDPS. The measurement process uses 0.3 of the amount of data contained in the learning data with the argument test_size = 0.3, with a random state = 5. The experimental test results show a 93.3% accuracy rate of adaptive boosting in detecting DoS attacks or not.

IV. CONCLUSION

The adaptive boosting algorithm does several iterations to get enough learning data for the predictive voting process. At each iteration, error detection can occur due to small learning data. Error detection can be minimized by a large amount of learning data and a large number of repetitions. Adaptive boosting can be implemented as intrusion detection and prevention on software-defined networks. IDPs with adaptive boosting and decision stump has an accuracy of 93.3% capable of detecting attacks and preventing access in real-time.

V. REFERENCES

[1] A. Abubakar and B. Pranggono, "Machine Learning-Based Intrusion Detection System For Software Defined Networks," *IEEE Electronics ISSN: 2472-7601,* 2017.

[2] P. M. Ombase and N. P. Kulkarni, "Survey on DoS Attack Challenges in Software Defined Networking," *International Journal of Computer Applications (ISSN : 0975 – 8887),* vol. 173, no. 2, 2017.

[3] Y. Qiao and R. Yu, "Software-Defined Networking (SDN) and Distributed Denial of Services (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges," *IEEE Communications Survey & Tutorials,* vol. 18, 2016.

[4] S. Riya, K. Payal, S. Kalyani and K. Madhuri, "Intrusion Detection System based Software Defined Networking," *International Journal of Advance Engineering and Research Development,* vol. 4, no. 1, 2017.

[5] S. D. Snehlata and K. W. Kapil, "Intrusion Detection System Using New Ensemble Boosting Approach," *International Journal of Modeling and Optimization,* vol. 2, no. 4, August 2012.

[6] B. E. Egbenimi, A. Omar, J. Priya, B. Sandeep and N. Takamasa, "Software Defined Networks' Security: An Analysis of Issues and Solutions," *International Journal of Scientific & Engineering Research ISSN 2229-5518,* vol. 6, no. 5, 2015.

[7] P. N. Garcia, O. C. Garcia and C. Fyfe, Nonlinear Boosting Projections for Ensemble Construction, J Mach Learn Res, 2008, p. 1–33.

[8] C. Chen, "A Hybrid Approach Combining R2L-Based And Anomaly-Based Detection Against DDoS Attacks," *International Journal Of Network Security & Its Applications,* vol. 8, no. 5, September 2016.

[9] J. Ashraf and L. Seemab, "Handling Intrusion and DDoS attack in Software Define Networks Using Machine Learning Technique," *Software Engineering Conference (NSEC), National,* 2014.

[10] F. Zhuang, Y. Xuming and T. Liming, "An Adaptive Boosting Algorithm For Image Denoising," *Hindawi, Mathematical Problems in Engineering,* February 2019.

[11] P. Natesan, P. Balasubramanian and G. Gowrison, "Improving the Attack Detection Rate in Network Intrusion Detection using Adaboost Algorithm," *Journal of Computer Science ISSN:1549-3636,* vol. 8, no. 7, pp. 1041-1048, 2012.

[12] M. Khizer and A. Basit, "Implementation Face Detection System using Adaptive Boosting Algorithm," *International Journal of*

Computer Applications (0975 – 8887), vol. 76, no. 2, pp. 51-57, August 2013.

[13] M. Stuart, S. Joel and K. George, Hacking Exposed, McGraw-Hill, Osborne, 2001, p. 504 – 525.

[14] E. S. Robert and Y. Freund, Adaptive Computation and Machine Learning, London, England: The MIT Press Cambridge, Massachusetts, 2012.

[15] H. Aron, J. F. David and Brubaker, Adaboost Machine Learning and Data Mining Lecturer Notes, Toronto: University of Toronto, 2015.

[16] A. S. Elden, H. Moustafa, H. M. Harb and A. Emara, *Adaboost Ensemble with Simple Genetic Algorithm for Student Prediction Model*. *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 5, no. 2, p. 73 – 85, 2013.

[17] J. Friedman, T. Hastie and R. Tibshirani, "The Annals of Statistics," *Additive Logistic Regression: A Statistical View of Boosting* , vol. 28, no. 2, pp. 337-407, 2000.

[18] S. Kotsiantis, K. Dimitris and E. Pintelas, "Local Boosting of Decision Stumps for Regression and Classification Problems," *Journal Of Computers*, vol. 1, no. 4, 2006.

Anomaly-based Intrusion Detection and Prevention Using Adaptive Boosting in Software-defined Network

ORIGINALITY REPORT

4%	3%	5%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.upnyk.ac.id Internet Source	2%
2	"Information Systems Design and Intelligent Applications", Springer Science and Business Media LLC, 2018 Publication	2%

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	Off		