2018 IEEE Fourth International Conference on Big Data Computing Service and Applications

Classification of Attack Types for Intrusion Detection Systems using a Machine Learning Algorithm

Kinam Park
School of Software
Sungkyunkwan University
Suwon-si, South Korea
Knpark2008@skku.edu

Youngrok Song
School of Software
Sungkyunkwan University
Suwon-si, South Korea
Id4thomas@skku.edu

Yun-Gyung Cheong
School of Software
Sungkyunkwan University
Suwon-si, South Korea
Aimecca@skku.edu

***Abstract*— In this paper, we present the results of our experiments to evaluate the performance of detecting different types of attacks (e.g., IDS, Malware, and Shellcode). We analyze the recognition performance by applying the Random Forest algorithm to the various datasets that are constructed from the Kyoto 2006+ dataset, which is the latest network packet data collected for developing Intrusion Detection Systems. We conclude with discussions and future research projects.**

***Keywords*— Supervised Machine Learning; Kyoto2006+; Labeling; Intrusion Detection System; Classification**

I. INTRODUCTION

The IDS (Intrusion Detection System) is a defense against attacks attempting to steal information stored on various platforms such as servers and personal computers. In the case of widely known attacks, it is easy for the administrator to judge and process it immediately, but it is unclear to judge unknown abnormal data, and the cost of restoration increases as the handling is delayed [8].

Machine Learning techniques are widely used in IDS due to its ability to classify normal/attack network packets by learning patterns based on the collected data. There are many results for classification of normal/attack, however, there is little work on classifying different attack types.

Therefore, an IDS that can detect a particular attack type may not be able to respond properly to arracks that are developing in various forms. The purpose of classifying network threats is to quickly respond to attacks according to attack types, and to respond to urgent attacks first when the importance is different. We can solve this problem through the advantages of artificial intelligence mentioned above.

We selected Kyoto 2006+ as a learning dataset. Kyoto 2006+ dataset [1] contains network traffic data collected from November, 2006 to December, 2015. The dataset is also available for big data analysis, of which size is 19.683 gigabytes. In addition, the most commonly used dataset for IDS research is KDD Cup99 [7]. However, the dataset was collected in 1999, and may not contain the latest network intrusion patterns. In addition, this dataset is collected from a virtual network environment, which makes it different from the patterns observed in real network systems.

In this paper, we describe our work. The section 2 describes the related work and background knowledge, and the section 3 details the evaluation process that includes the construction of various datasets for testing the algorithm and the evaluation configuration. The section 4 presents the results of our experiments to evaluate the performance of detecting IDS, Malware and Shellcode. Finally, we conclude with discussions and future work.

II. BACKGROUNRD AND RELATED WORK

A. Maching Learning for IDS and Classification

There are a number of works of intrusion detection systems that applied machine learning algorithms to Kyoto 2006+ dataset [1]. Song et al. proposed an intrusion detection approach based on correlation of the results obtained from the two one-class SVM models, one model trained with raw traffic data and the other model trained with Snort alerts, respectively [2]. In [3], Sallay et al. proposed a real time intrusion detection alert classifier based on online self-trained support vector machines in order to identify real attacks correctly even when a high ratio of false positives exists in the intrusion alerts. In [4], Chitrakar et al. developed an intrusion detection system based on a candidate support vector based incremental SVM algorithm, and evaluated the IDS using the Kyoto 2006+ dataset. In [5], Ishida et al. proposed an anomaly based intrusion detection approach that combines OptiGrid clustering and a cluster labelling algorithm using grids to extract the feature of traffic data and identify the attack traffic accurately. In [6], Ambusaidi et al. proposed a mutual information based algorithm to select the optimal set of features for classification from high dimensional

DOI 10.1109/BigDataService.2018.00050

traffic data and a least square support vector machine based IDS integrated with the optimal feature selection algorithm.

B. Evaluation Metrics

Selecting evaluation indicator that can evaluate performance objectively is important for comparing performance between various methods or different datasets. IDS performance evaluation usually uses accuracy mainly. Accuracy is computed as the number of correctly classified data over the total number of data (Equation 1). True Positive (TP) and True Negative (TN) means the number of correctly classified as positive or negative. False Positive (FP) means that a negative instance is predicted as positive, and False Negative (FN) means the opposite (predicted negative when the instance is positive).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Although accuracy is an intuitive measurement, it can give when the data is imbalanced. For example, a dataset of 1000 instances contain 990 positive and 10 negative instances. For a high accuracy measurement, one can ignore all the negative cases and predict an input as positive. This results in a high accuracy of 0.99.

Precision indicates the portion of data correctly predicted positive over the number of data predicted as positive.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall (i.e., sensitivity, detection rate) indicates the percentage of correctly predicted positive instances out of the number of actual positive instances. Recall serves as a main performance indicator when it is important to detect all positive data, such as IDS.

$$Recall = \frac{TP}{FN+TP} \quad (3)$$

To increase the precision of a certain class, you only need to predict that a data instance belongs to a certain class only when the probability of belonging to the class is very high. At the same time, the performance of the recall becomes lowered because there are instances belonging to the class are likely to be excluded due to the high threshold.

Precision and Recall are both important indicators and using only one of them is not sufficient to evaluate the IDS performance. F1-score is the harmonic mean of the two, which considers Precision and Recall together. With unbalanced binary classification datasets, F1-score can be a better indicator than accuracy. F1-score can be obtained using the following equation that computes Fβ-score, by substituting β with 1.

$$F_\beta = \frac{(1+\beta^2)(PXR)}{(\beta^2 P+R)} \quad (4)$$

F1 is a metric that considers precision and recall equally, while F2 considers recall two times more important than precision. F0.5 considers precision twice as important as recall. Since IDS needs to detect all the attacks as much as possible, using F2 as the primary metric is appropriate. In this study, all the major metrics (accuracy, precision, recall, and F-score) are presented for comparison in the evaluation section.

III. EVALUATION

This section describes the experimental procedure and the evaluation results that we obtained.

A. Data Preparation

The Kyoto 2006+ dataset is a massive 19.78GB of data collected over the past nine years and two months from November 2006 to December 2015. The data used in the experiments were selected according to the following criteria. We initially constructed a dataset between 3 and 7 days as recommended by Dr. Song, one of the researchers who collected the Kyoto 2006+ dataset and an author of [1]. The larger the data set, the more noise is likely to be present, which will be learned by machine learning algorithms. However, we found that the number of data sets per class in the dataset is reduced substantially when the attack class is subdivided into different attack types (e.g., IDS, Malware, and Shellcode). Therefore, we set the duration of data collected a month, so that the dataset contains all different attack types and has enough data per attack type.

Figure 1 shows the process of constructing various datasets. First, the monthly statistics for the six labels are obtained and select the appropriate month for the purpose of the study. Finally, subdivide labels in the process of creating training and test datasets.

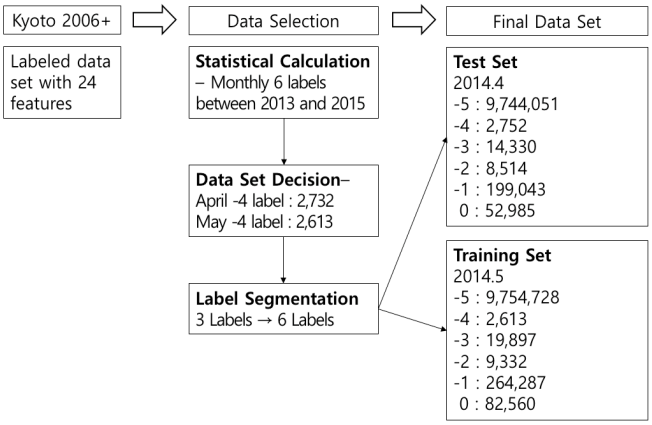

Figure 1 The process of building training and test datasets for evaluation

Table 1 Label Segmentation

IDS	Malware	Shellcode	Label	New Class	New Label
0	0	0	1	Normal	0
0	0	0	-1	Unknown	-5
0	0	1	-2	Shellcode	-4
1	0	1	-1	IDS+Shellcode	-3
1	1	1	-1	IDS+Shellcode	
0	1	0	-1	Malware	-2
0	1	1	-1	Malware	
1	1	0	-1	Malware	
1	0	0	-1	IDS	-1

Table 2 Statistic of training and test data. The number in parentheses denotes the percentage of the label in the set.

Label	Training (2014.5)	Test (2014.4)
-5	9,754,728 (96.26%)	9,744,051 (97.23%)
-4	2,613 (0.03%)	2,752 (0.03%)
-3	19,897 (0.2%)	14,330 (0.14%)
-2	9,332 (0.09%)	8,514 (0.08%)
-1	264,287 (2.61%)	199,043 (1.99%)
0	82,560 (0.81%)	52,985 (0.53%)

The Kyoto 2006+ dataset contains 3 class types: -1 (attack), -2 (shellcode), and 1 (normal). For the -1 and -2 labels, there are three attack types (e.g., IDS, Malware, Shellcode). As shown in Table 1, a single packet can be detected by several IDSs. For instance, the packet in the fourth row was detected by IDS and Shellcode at the same time, and its original label is -1. However, we discovered that some packets that were detected by IDSs are labeled as normal (e.g., 1). We removed these data, considering them as errors. More details about the data collection process is referred to [1].

Table 1. shows the new label assignment criteria. We set the class name as the detected attack field name, and in the case of detecting malware along with other attack types, we use malware. Because the detection of malware is more important than the detection of the other two threats.

Table 2. shows the number of instances for each class. It can be seen that the unknown attack occupies the largest number of data, and the sizes of the remaining classes are less than 1% except IDS class.

B. Selection of Evaluation Metrics and Machine Learning Algorithm

In network intrusion detection systems, it is important to know how well the intrusion is detected, so the recall is more important than the precision. Therefore, both F1-score and F2-score are presented in this study.

The Random Forest algorithm was implemented using the Scikit-learn 0.19.0 version [], a machine learning library written in Python 3.6 version. The PC specification used in the evaluation was Intel Core i5 3.1GHz, 8GB memory, and Mac OS. The parameter values of the algorithm are set to the default values.

C. Learning Features

We used 17 features for our experiment, which consists of 14 items that are selected as important for intrusion detection in KDD Cup 99 and 3 of 10 newly created items in Kyoto 2006+. Since the 10 newly created features in Kyoto 2006+ (see Table 3) were created for verification purposes and contain critical information for attack detection, we excluded the three detection fields (IDS, Malware, Shellcode) during the training session that may enhance the evaluation performance. Then, all the data were normalized.

Table 3 New Features of Kyoto2006+

Feature	Meaning	Note
Duration	The length of the connection.	Used
Service	The connection's service type, e.q., http, telnet, etc.	Used
Source Bytes	The number of data bytes sent by the source IP address.	Used
Destination Bytes	The number of data bytes sent by the destination IP address.	Used
Count	The number of connections whose source IP address and destination IP address are the same to those of the current connection in the past two seconds.	Used
Same_srv _rate	Percentage of connections to the same service in Count feature.	Used
Serror_rate	Percentage of connections that have "SYN" errors in Count feature.	Used

Srv_serror _rate	Percentage of connections that have "SYN" errors in Srv_count feature.	Used
Dst_host _count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose source IP address is also the same to that of the current connection.	Used
Dst_host_srv _count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose service type is also the same to that of the current connection.	Used
Dst_host_same _src_port_rate	Percentage of connections whose source port is the same to that of the current connections in Dst_host_count feature.	Used
Dst_host _serror_rate	Percentage of connections that have "SYN" errors in Dst_host_srv_count feature.	Used
Dst_host_srv _serror_rate	Percentage of connections that "SYN" errors in Dst_host_srv_count feature.	Used
Flag	The state of the connection at the time the summary was written.	Used
IDS	It indicates whether IDS(Intrusion Detection System) triggered an alert for the connection; '0' means any alerts were not triggered, and an arabic numeral(except '0') means the different kinds of the alerts. Parenthesis indicates the number of the same alert observed during the connection.	Unused
Malware	It indicates whether malware, also known as malicious software, was observed in the connection; '0' means no malware was ob-served, and a string indicates the corresponding malware observed at the connection.	Unused
Shellcode	It indicates whether shellcode; '0' means no shellcodes and exploit codes were observed, and an arabic numeral (except '0') means the different kinds of the shellcodes or exploit codes. Parenthesis indicates the number of the same shellcode or exploit code observed during the connection.	Unused
Label	It indicates whether the session was attack or not; '1' means the session was normal, '-1' means known attack was observed in the session, and '-2' means unknown attack was observed in the session.	Used (Training)
Source IP Address	It indicates the source IP address used in the session. Also, the same private IP addresses are only valid in the same month.	Unused
Source Port Number	It indicates the source port number used in the session.	Used
Destination IP Address	indicates the source IP address used in the session. Also, the same private IP addresses are only valid in the same month.	Unused
Destination Port Number	It indicates the destination port number used in the session.	Used
Start Time	It indicates when the session was started.	Unused
Protocol	It indicates the protocol type.	Used

IV. RESULTS

This section describes the results of applying the Random Forest algorithm to the selected training and test datasets we constructed. First, we experimented on predicting six classes using a dataset that were collected between April and May, 2014. Next, we built an additional dataset to tackle the data imbalance problem, and then we carried out the experiments on the new set again.

A. Prediction Results

Table 4 shows the performance of six classes predicted through Random Forest algorithm in the order of precision, recall, F_1-score, F_2-score, and accuracy, when the data collected in May, 2014 is used for training and the data collected in April, 2014 was used for testing. The result value shown at the end is the weighted average, averaging the performance values weighted by its class size.

The overall performance (i.e., weighted average) is good enough to reach 0.99 for all evaluation metrics. Yet, the performance of predicting each attack type differs greatly. While the detection rate for unknown attack is high (F_1 score of 0.99), the detection of Shellcode attack (-4 label) shows a poor performance as low as 0.16 of F_1 score. The IDS attack and normal classes (-1 and 0 labels) did not show good results either. Since the data is highly unbalanced, the weighted average performance can be high when the major class detection shows a good result, neglecting the performance of classes containing small much less data.

Table 4 Performance comparison of the Random Forest algorithm for Dataset A

	Precision	Recall	F_1 Score	F_2 score	Accuracy	Number of instances
-5	0.99	1.00	0.99	1.00		9744051
-4	0.31	0.11	0.16	0.13		2752
-3	0.82	0.91	0.86	0.89		14330
-2	0.98	0.99	0.98	0.99		8514
-1	0.89	0.70	0.78	0.73		181125
0	0.69	0.74	0.72	0.73		52985
Result	0.99	0.99	0.99	0.99	0.99	10003757

B. Prediction Results of Under-Sampled Datasets

In the previous dataset, the label -4 (Shellcode) contained the smallest number of instances. To tackle the data imbalance problem, we randomly under-sampled the training data, setting the number of all the classes to the number of class -4, which is 2,613. Therefore, the number of instances of all the classes was arbitrarily adjusted to

2,613. The same test set (April, 2014) was again used for comparison.

Table 5 shows the results of this experimentation. The overall performance has dropped significantly and the performance for each class has also decreased. We believe that the 2,613 instances per class were not sufficient for the machine learning algorithm. The performance of the class -4 (Shellcode) was even lower than that of the previous data set.

Table 5 Performance comparison of the Random Forest algorithm for under-sampled dataset

	Precision	Recall	F_1 Score	F_2 score	Accuracy	Number of instances
-5	0.99	0.85	0.92	0.87		9744051
-4	0.02	0.09	0.03	0.05		2752
-3	0.11	0.41	0.18	0.27		14330
-2	0.20	0.81	0.32	0.50		8514
-1	0.13	0.76	0.22	0.39		181125
0	0.06	0.55	0.10	0.21		52985
Result	0.97	0.84	0.90	0.86	0.84	10003757

V. CONCLUSION AND FUTURE WORK

In this study, we analyzed class-specific detection of Kyoto 2006+ datasets, using the Random Forest algorithm, an efficient supervised machine learning algorithm for IDS. We first refined the original 3 classes (i.e., normal, known attack unknown attack) into 6 classes (i.e., normal, unknown, shellcode, IDS+shellcode, malware, IDS). Next, we constructed one test dataset and the two training datasets that vary in the size among classes for evaluating the performance of detecting the different attack types.

Although we obtained a high overall detection performance when trained with the first training set (0.99 of precision, recall, F_1-score, and F_2-score), we found that the performance for each class differs greatly (as low as 0.16 of F_1-score for shellcode attack). For this reason, we built the second training set via random under-sampling to set the size of all the class equal to the number of instances of the smallest class (i.e., shellcode). The evaluation resulted in much lower performances for all the classes, which was disappointing. We believe that the size of data was not sufficient, and training with the same size class may not be ideal for the machine learning approaches. We also note that the unknown attack class still shows a good performance, F_1-score of 0.90, which suggests that there is a distinct pattern for the unknown attack.

In the future, we will further investigate the performance of detecting different attack types using the Kyoto 2006+ dataset varying the training conditions.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2016R1D1A1B03933002). This research was supported by the MISP(Ministry of Science, ICT & Future Planning), Korea, under the National Program for Excellence in SW(R2215-16-1005) supervised by the IITP(Institute for Information & communications Technology Promotion). This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2017-0-01642) supervised by the IITP(Institute for Information & communications Technology Promotion)

REFERENCES

[1] Song, Jungsuk, et al. "Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation." Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security. ACM, 2011.

[2] Song, Jungsuk, et al. "Correlation analysis between honeypot data and IDS alerts using one-class SVM." Intrusion Detection Systems. InTech, 2011.

[3] Sallay, Hassen, and Sami Bourouis. "Intrusion detection alert management for high-speed networks: current researches and applications." Security and Communication Networks 8.18 (2015): 4362-4372.

[4] Chitrakar, Roshan, and Chuanhe Huang. "Selection of Candidate Support Vectors in incremental SVM for network intrusion detection." computers & security 45 (2014): 231-241.

[5] Ishida, Moriteru, Hiroki Takakura, and Yasuo Okabe. "High-performance intrusion detection using optigrid clustering and grid-based labelling." Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on. IEEE, 2011.

[6] Ambusaidi, Mohammed A., et al. "Building an intrusion detection system using a filter-based feature selection algorithm." IEEE transactions on computers 65.10 (2016): 2986-2998.

[7] KDD Cup 1999. Available on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, Ocotber 2007.

[8] http://www.ddaily.co.kr/news/article.html?no=157416)