Poisoning Attack Data Detection with Internal Coefficient Displacement for Machine Learning Based NIDS

Hajime Shimada
Information Technology Center, Nagoya University
Nagoya, Japan
Email: shimada@itc.nagoya-u.ac.jp

Takuya Kuwayama
Grad. Sch. Informatics, Nagoya University
Nagoya, Japan
Email: kuwayama@net.itc.nagoya-u.ac.jp

Hirokazu Hasegawa
Center for Strategic Cyber Resilience R&D, NII
Tokyo, Japan
Email: hasegawa@nii.ac.jp

Yukiko Yamaguchi
Information Technology Center, Nagoya University
Nagoya, Japan
Email: yamaguchi@itc.nagoya-u.ac.jp

***Abstract*—Due to the improvement of Machine Learning (ML) techniques, ML has been used extensively in the cyber security area and Machine Learning based Network-based Intrusion Detection Systems (ML-NIDS) is a one of those examples. However, arising methods to attack ML systems are becoming new threats to them. A poisoning attack is one of those threats and has adverse effects on the classification performance. As a threat on ML-NIDS, we are concerned about a threat where an attacker distributes manipulated traffic session data as a new dataset, aiming at a poisoning attack on ML-NIDS. In this paper, we try to identify whether newly added training data is poisoning attack data or not based on the displacement of an internal coefficient of a classifier. This research utilizes Support Vector Machine (SVM) as a classifier so that the internal coefficient vector is represented as a gradient coefficient vector of hyperplane in SVM classifier. We assumed that manipulated traffic session data for poisoning attack will largely confuse the internal coefficient vector. Thus, if the internal coefficient vector displaces largely after retraining with newly added data, we estimate that the newly added data is a poisoning attack data. We also propose a method to define a threshold value that distinguishes poisoning attack data and clean data. We evaluated our proposal with SVM based NIDS with an open traffic session dataset and poisoning attack with Biggio's SVM poisoning algorithm. We confirmed that our proposal can detect poisoning attack data and achieves 0.9838 F1 score at 8% poisoning rate (ratio of newly added poisoning attack training data to existing clean data), which is better performance compared to the existing poisoning attack data detection method.**

Keywords–poisoning attack detection; machine learning based NIDS.

I. INTRODUCTION

Due to the improvement of Machine Learning (ML) techniques, machine learning has been used extensively in the field of cyber security. Machine Learning based Network-based Intrusion Detection Systems (ML-NIDS) is a one of those examples. However, many people are proposing new attack methods to ML systems and they are becoming new threats in the cyber security area. There is a poisoning attack that has an adverse effect to the classification performance. The poisoning attack distributes malicious data in advance and that malicious data contaminate and pollute training data. As a threat on ML-NIDS area, we are concerned about a threat where an attacker distributes manipulated traffic session data as a new dataset with aiming poisoning attack, and some ML-NIDS system maintainers wrongly include them in training data.

Although several methods have been proposed to detect and exclude poison data from training data, there are few studies that evaluate the effectiveness of defense methods in machine learning-based NIDS. In this paper, we try to identify whether newly added training data is poisoning attack data or not based on the displacement of an internal coefficient of a classifier. This research utilizes Support Vector Machine (SVM) for a classifier so that the internal coefficient vector is represented as a gradient coefficient vector of hyperplane in SVM classifier. We assumed that manipulated traffic session data for poisoning attack will largely confuse the internal coefficient vector. Thus, if the internal coefficient vector displaces largely after retraining with newly added data, we estimate that the newly added data is a poisoning attack data. This method requires a threshold value to distinguish poisoning attack data and clean data. We also proposed how to define the threshold value from existing clean data. Our proposed method creates the threshold value by dividing existing clean data into baseline data, additional clean data, and additional local poisoning attack data which is generated by some poisoning attack algorithm from existing clean data. By comparing internal coefficient vector displacements between additional clean data learning result and additional local poisoning attack data learning result, we can obtain the threshold value. We assume that ML-NIDS system maintainer perform such an evaluation of additional data to exclude poisoning attack data comes from some attackers.

We evaluated our proposal with Kyoto 2016 Dataset [1][2] which is a traffic session dataset with malicious/benign ground truth label. We created SVM classifier and performed the poisoning attack with Biggio's SVM poisoning algorithm [3] for baseline. Then, we evaluated our internal coefficient displacement based detection with Euclidean distance and compared it with existing SVM poisoning attack detection method Curie [4]. We confirmed that our proposal can detect poisoning attack data effectively in many poisoning rate (ratio of newly added poisoning attack training data to existing clean data) and it achieves 0.9838 F1 score at 8% poisoning rate as a best score. On the other hand, Curie gives moderate performance because Curie evaluates in individual traffic session

 ISBN: 978-1-68558-186-2

sample level so that it cannot exclude a poisoning attack traffic session sample that have quite similar characteristic to existing clean data samples.

The rest of the paper is organized as follows. Section II introduces related works about ML-NIDS, traffic datasets, poisoning attacks, and poisoning attack data detection. Section III introduces our proposal that distinguishes poisoning attack data with the displacement of the internal coefficient vector before and after retraining. We also propose a method to define the threshold value to distinguish poisoning attack data and clean data. Section IV shows evaluation setups, evaluation results of our proposal, and evaluation results of Curie as a comparison target. Finally, we conclude and introduce future works in Section V.

II. Related Work

A. Network-based Intrusion Detection Systems and Researches

Network-based Intrusion Detection Systems (NIDS) are widely used for observing malicious network traffic and some of them are working as Network-based Intrusion Protection Systems (NIPS). Nowadays, we equip NIDS not only at a border, like the Internet gateway, but also some observation points in intranet. The detection method is largely separated into signature based detection and behavior based detection and ML (ML-NIDS) is widely used for behavior based detection.

Researches on ML-NIDS start from comparatively ancient age. For example, Mukkamala et al. proposed ML-NIDS using Neural Network and support vector machine in 2002 [5]. Currently, there are too many successor researches in ML-NIDS. Nowadays, ML-NIDS is already used in commercial security appliances and many companies consider that ML technologies are effectively working in their security appliances. For example, Sophos Ltd. is applying deep learning protection in their Sandstorm UTM (Unified Threat Management) appliance or software [6].

B. Traffic Dataset

To promote NIDS research, we have to obtain traffic data including malicious/benign traffic data. However, it is hard for many researchers to create own experimental network which can generate both malicious/benign traffic so that many researcher use traffic dataset to promote their own research.

KDD (Knowledge Discovery and Data mining) Cup 1999 Data is one of the most famous traffic dataset [7]. It summarizes traffic data of 1999 DARPA Intrusion Detection Evaluation Dataset [8] into traffic session level so that it can list many traffic data with small file size. It is also famous for adding statistical data as a feature of a traffic session mainly derived from relationships among sessions.

Kyoto 2006+ dataset is a dataset which is generated by Song et al. [9]. The data source is honeypot at Kyoto University and they generate KDD Cup 1999 Data like traffic session level dataset with malicious/benign ground truth label given by security appliances. It equips not only statistical features existing in KDD Cup 1999 Data but also newly generated statistical features. Kyoto 2016 dataset [1][2] is an extension of Kyoto 2006+ dataset. The duration of Kyoto 2006+ dataset is 3 years, but the duration of Kyoto 2016 dataset is 9 years. Kyoto 2016 dataset also gives much more session data even in Kyoto 2006+ duration because PC and software advancement makes additional interpretation to pcap file which could not be interpreted in Kyoto 2006+ age.

C. Poisoning Attack Data Generation

Several researchers touch poisoning attack data generation and its performance.

Biggio et al. proposed SVM poisoning attack algorithm that generates poisoning attack data [3]. It is one of a gradient ascent methods and similar to a gradient descent method on Neural Network. An outlined algorithm is shown as Algorithm 1. It updates L to maximizing loss function in SVM. In this research, we use this algorithm for poisoning attack data generation.

Apruzzese et al. evaluated a poisoning attack to ML-NIDS in an experimental network with normal traffic and malware originated attack traffic [10]. They generated poisoning data with randomly increasing feature vector values from clean attack traffic data and confirmed dramatic degradation of True Positive Rate (TPR) in Random Forest, Multiple Layer Perception, K-Nearest Neighbor methods.

Algorithm 1 Biggio's SVM poisoning attack[3].

Input: training data $\mathcal{D}_{\mathrm{tr}}$, validation data $\mathcal{D}_{\mathrm{val}}$, feature vector and ground truth label of initial attack point $\{x_c^{(0)}, y_c\}$, step size t

Output: Feature vector of one adversarial training data x_c

1: Train SVM with $\mathcal{D}_{\mathrm{tr}}$
2: Current iterations $k \leftarrow 0$.
3: **repeat**
4: Train SVM again with $\mathcal{D}_{\mathrm{tr}} \cup (x_c^{(k)}, y_c)$
5: Calculate gradient of loss function $\frac{\mathrm{d}L}{\mathrm{d}u}$ with $\mathcal{D}_{\mathrm{val}}$
6: Let u to parallel vector to $\frac{\mathrm{d}L}{\mathrm{d}u}$
7: Update adversarial training data by $k \leftarrow k+1$, $x_c^{(k)} \leftarrow x_c^{(k-1)} + tu$
8: **until** $L\left(x_c^{(k)}\right) - L\left(x_c^{(k-1)}\right) < \epsilon$
9: **return:** $x_c = x_c^{(k)}$

D. Countermeasure to Poisoning Attack Data

There are two directions on countermeasures to poisoning attack data. The one is a hardening a classifier training algorithm not to be affected by poisoning attack data and the other one is a method to detect and exclude poisoning attack data.

Zhou et al. have promoted research in the hardening the classifier training algorithm [11]. They proposed AD-SVM (ADversarial Support Vector Machine) which has additional constraints that is designed for considering poisoning attack data may try to maximize hinge loss. However, it gives some adversarial affect to classification performance because a typical SVM tries to minimize hinge loss.

There are several researches in the method to detect and exclude poisoning attack data. Steinhardt et al. have proposed a method to detect poisoning attack data by combining estimating a barycenter of data class and outlier detection algorithm

[12]. They also confirmed resistance to poisoning attack data. They find that MNIST-1-7 dataset and Dogfish data have high resistance to poisoning attack data but IMDB Sentiment dataset has low resistance. Taheri et al. have tried to estimate an original ground truth label which is flipped when generating poisoning attack data [13]. They utilized Neural Network for estimation and rated data as poisoning attack data if ground truth label is differed from an estimated label.

Laishram et al. proposed Curie that is an algorithm to exclude poisoning attack data generated by SVM poisoning attack [4]. An outlined algorithm of Curie is shown as Algorithm 2. Curie also exploits flipped label similar to Taheri's method. Curie firstly compresses training data to two dimensions and performs clustering with DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Then, Curie adds class label that is weighted with constant and treated as 3rd dimension. Fianlly, Cuirie calculates an average distance between samples in a same cluster. If a sample is a clean data, the distance tend to become small. If a sample is a poisoning attack data, distance tend to become large. Curie detects and excludes poisoning attack data with an above criterion.

Algorithm 2 Algorithm of Curie [4].

Input: $Data = (F, C)$ for inspection. (F is a set of feature vector, C is a set of class label)
Output: Set of vector M which has excluded poisoning attack data

1: $PcaData \leftarrow$ PCA($Data.F$) {Compress data to two dimension}
2: $Clusters \leftarrow$ DBSCAN($PcaData$) {Clustering data by DBSCAN}
3: **for** $point \in Data$ **do**
4: $point.F \leftarrow$ Append($point.F, point.C \times \omega$) {Add weighted ground truth label as 3rd dimension of feature}
5: $cls \leftarrow$ GetCluster($point, Clusters$)
6: $sample \leftarrow$ Sample($cls, count$)
7: **for** $s \in sample$ **do**
8: $s.F \leftarrow$ Append($s.F, s.C \times weight$)
9: $d \leftarrow$ EucledianDistance($point.F, s.F$)
10: $Dist.point \leftarrow Dist.point + d$
11: **end for**
12: $Dist.point \leftarrow Dist.point/$Size($cls$) {Calculate average distance from randomly selected 10 samples in same cluster}
13: $Dist \leftarrow$ ZScore($Dist$) {Normalization with Z value}
14: **end for**
15: **for** $point \in Data$ **do**
16: **if** $Dist.point \leq \theta$ **then**
17: $Result \leftarrow$ Append($Result, point$)
18: **end if**
19: **end for**{Choose samples that have over θ reliability}
20: **return:** $Result$

III. Proposal of Internal Coefficient Displacement Based Detection

A. Assuming Poisoning Attack Scenario

Figure 1 shows an assumed scenario of poisoning attack threat. A company working on security measures is working for updating a classifier of ML-NIDS to catch up with latest cyber attacks. To update classifier, the company gathers traffic data including malicious and benign traffic data from published traffic dataset. Some attacker considers attacking some organization that is using ML-NIDS of the company and the attacker try to weaken ML-NIDS to pass through attack traffic (considering backdoor attack). The attacker tries to generate poisoning attack data from an existing traffic dataset and publish it as a new traffic dataset. If the company includes the poisoning attack data to ones training dataset, the company creates classifier from polluted dataset and it may generate a polluted classifier that have a backdoor. If the polluted classifier has been published, an intention of the attacker has succeeded.

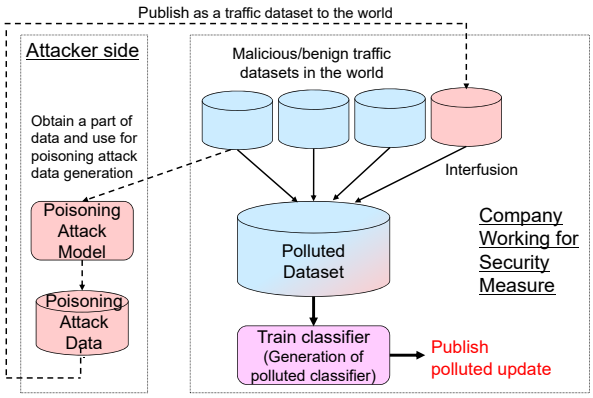

Figure 1. Assuming scenario: perform poisoning attack by distributing poisoning attack data.

B. Idea of Internal Coefficient Displacement Based Detection

To detect a block of poisoning attack data, we assumed "Poisoning attack data are designed to confuse classification criterion (e.g., hyperplane) so that it may largely displaces an internal coefficient of a classifier" so that we considered to detect the poisoning attack data from the internal coefficient distance between before and after retraining with additional data. Based on above assumption, we also assumed "The internal coefficient distance between before and after retraining may become large value if the additional data contain poisoning attack data. On the other hand, if the additional data is only clean data, internal coefficient distance becomes moderate value." so that we created a following procedure to detect poisoning attack data.

Figure 2 represents a poisoning attack data detection method based on an above assumption. Firstly, we create a classifier from reliable datasets O. Then, we add some additional training data A to O. We obtain classifier C_O and C_{OA} from both data blocks and obtain internal coefficient vectors v_O and v_{OA} from them. Then, we calculate Euclidean distance D between v_O and v_{OA} and compare with threshold value D_{th}. If D is larger than D_{th} that means classifier has largely displaced, we judge the data block A as poisoning attack data or the data block A contains some poisoning attack data. Otherwise, the data block A becomes clean data.

As shown in Section IV, we choose SVM for a classifier so that the internal coefficient becomes a gradient coefficient vector of SVM classifier. So, we evaluated the distance of the

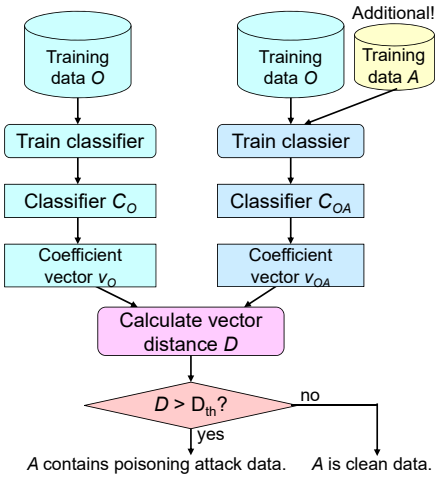

Figure 2. Proposal of internal coefficient distance based detection.

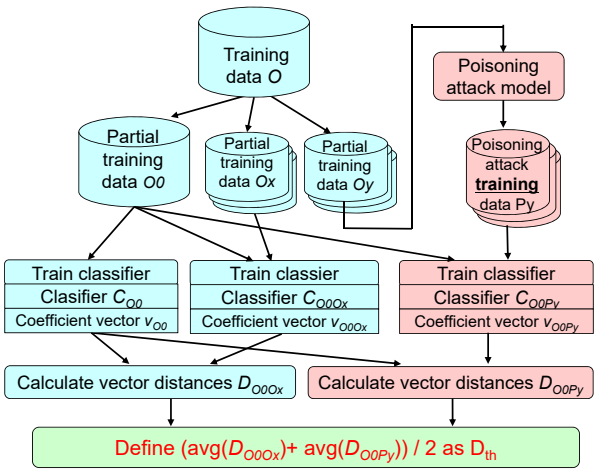

Figure 3. How to generate threshold value.

gradient coefficient vector in Section IV. But we think that many of ML algorithms have internal coefficient vectors so that this method can easily to be adopted into versatile ML algorithms.

C. Method to Define Threshold from Existing Data

To define threshold value D_{th} in Figure 2, we propose a method that generates D_{th} from existing clean data. To obtain displacement when we retrain the classifier with poisoning attack data, we generate poisoning attack training data from a part of the existing clean data.

Figure 3 shows an outlined flow of the proposal. Firstly, we divide existing clean training data O to following data blocks.

- Large size data block O_0 which is used for generating baseline classifier C_{O0}.
- Small size data blocks O_x which is used for generating classifier with additional clean data $C_{O0Ox}(x = 0, ..., n - 1)$.
- Small size data blocks O_y which is used for generating classifier with additional poisoning attack training data $C_{O0Py}(y = 0, ..., n - 1)$.

To generate poisoning attack training data, O_y is converted to P_y with an existing poisoning attack model in a poisoning attack detecting organization (e.g., security measure company).

The classifier generation part is similar to Figure 2. The baseline classifier C_{O0} is generated by training with training data O_0 and obtain coefficient vector v_{O0}. The classifier C_{O0Ox} is generated from merged data of training data O_0 and training data O_x and obtain coefficient vector v_{O0Ox}. The classifier C_{O0Py} is generated from merged data of training data O_0 and training data P_y and obtain coefficient vector v_{O0Py}. Then, we calculate distances between v_{O0} and v_{O0Ox}. We repeatedly calculate distances with different data blocks with varying x and y for n times and get an average. This average becomes an average coefficient perturbation when the classifier retrains with additional clean data. Similarly, we calculate distances between v_{O0} and v_{O0Py} and get average. This average becomes an average coefficient perturbation when the classifier retrains with poisoning attack data as additional data. We defined that the D_{th} is a middle of both averages.

In next section, we evaluate an adequacy of D_{th} definition with cross validation and compare it with an existing poisoning attack data exclusion method.

The method is partially introduced in domestic conference with malware binary feature classification [14]. This proposal extends the method with extending the method to NIDS including a time series update operation of the classifier.

IV. Evaluation

A. Experimental Setup

Before presenting the experimental results, we introduce an experimental setup. As a classifier to realize NIDS that classifies traffic session data into malicious or benign, we created a classifier with SVM. To generate poisoning attack data, we used Biggio's SVM poisoning algorithm [3] which is introduced Section II-C.

We used Kyoto 2016 [1][2] traffic dataset which is introduced Section II-B as a session level traffic dataset with malicious/benign ground truth label. We randomly picked up malicious/benign traffic sessions from November 2015 month of Kyoto 2016 dataset with keeping malicious:benign ratio to 1:1. We used 12 numeric parameters (duration, source bytes, destination bytes, same destination count, same service rate, SYN error rate, same service SYN error rate, same destination host count, same destination host and service count, same source port rate in same destination host count, SYN error rate in same destination host count, SYN error rate in same destination host and service count) of session data to generate SVM classifier.

To generate poisoning attack data, we choose 10,000 traffic session samples and generated poisoning attack data samples with Biggio's SVM poisoning algorithm [3] using Adversarial Robustness Toolbox library [15]. We confirmed that the generated poisoning attack data degrades SVM based ML-NIDS classification accuracy and classification accuracy degrades dramatically if poisoning attack data occupies more

TABLE I. HYPER-PARAMETER OF CURIE OBTAINED WITH BAYESIAN OPTIMIZATION.

Parameter	Value	Definition
$omega$	1030.6	Coefficient when adding ground truth to 3rd dimension (in step 4 of Algorithm 2)
$theta$	0.581	Threshold to distinct poison/clean based on average distance from cluster member (in step 16 of Algorithm 2)
$sample\_size$	36	Sampling amount from the same cluster (in step 6 of Algorithm 2)
eps	0.732	Distance to define the same cluster (in step 2 of Algorithm 2)
$min\_samples$	15	Distinct as noise if a number of samples in eps distance is smaller than this value (in step 2 of Algorithm 2)

TABLE II. RESULTS OF INTERNAL COEFFICIENT DISPLACEMENT METHOD.

Poisoning rate (num of data blocks)	Accuracy	Precision	Recall	F1 score
1% (303 blocks)	0.8861	0.8104	0.9551	0.8768
2% (147 blocks)	0.9592	0.9592	0.9592	0.9592
3% (96 blocks)	0.9430	0.9641	0.9250	0.9441
4% (72 blocks)	0.9510	0.9583	0.9446	0.9514
5% (57 blocks)	0.9421	0.9553	0.9308	0.9429
6% (45 blocks)	0.9500	0.9667	0.9355	0.9508
7% (39 blocks)	0.9558	0.9500	0.9611	0.9555
8% (33 blocks)	0.9841	0.9682	1.0000	0.9838
9% (30 blocks)	0.9825	1.0000	0.9662	0.9828
10% (27 blocks)	0.9639	0.9833	0.9465	0.9646
15% (15 blocks)	0.9500	0.9700	0.9327	0.9510
20% (12 blocks)	0.9625	0.9625	0.9625	0.9625

TABLE III. RESULTS OF CURIE METHOD.

Poisoning rate	Accuracy	Precision	Recall	F1 score
0.0%	0.7830	NaN	0.0000	NaN
2.5%	0.7902	0.8400	0.0905	0.1634
5.0%	0.7909	0.7692	0.1613	0.2667
7.5%	0.8039	0.8272	0.2528	0.3873
10.0%	0.8047	0.7838	0.3107	0.4450
12.5%	0.8074	0.7676	0.3682	0.4977
15.0%	0.8121	0.7614	0.4281	0.5481
17.5%	0.8152	0.7311	0.4814	0.5805
20.0%	0.8176	0.7400	0.5316	0.6187

than 20% of training data. These 10,000 poisoning attack data samples are treated as P_y. We choose different 12,960 traffic session samples as clean training data O. Data O is separated into 3,240 O_0 samples and 9,720 O_x samples. Data O_x and P_y are divided into 3 groups (around 3,000 samples per each group) to perform cross validation in our proposal. So, when we define threshold value D_{th}, from 1st group of O_x and P_y, we evaluate D_{th} with 2nd and 3rd group of O_x and P_y. Similarly, when we define D_{th} from 2nd and 3rd groups of O_x and P_y, we evaluate D_{th} with another groups. In this way, we achieve cross validation of D_{th} and obtain classification performance metric values.

As a comparison target to our proposal, we implemented Curie [4] from scratch. To tune hyper-parameter of Curie for traffic session samples, we used Bayesian optimization in Scikit-Optimization library [16]. Table I shows obtained hyper-parameters of Curie.

B. Evaluation results

Table II shows detection performance of our proposed internal coefficient displacement method under different poisoning rate. Poisoning rate means a rate of additional (poisoning) data block amount compared to original training data. If a number of original training data is 3,000 and additional (poisoning) data is 30, the poisoning rate becomes 1%. If additional training data is poisoning attack data and D_{th} distinguishes it as poisoning attack data, the result becomes True Positive. If additional training data is clean data and D_{th} distinguishes it as clean data, the result becomes True Negative. False Positive and False Negative classes are defined as similar. We can obtain multiple additional training data blocks from O_x and P_y so that we evaluate with many additional training data blocks as possible. For example, in 1% poisoning rate, we can create 101 additional data blocks from each group of O_x and P_y so that we evaluated with 303 times trial (101 additional data blocks per group times 3 groups) in 1% poisoning rate. The number of data blocks are noted beside individual poisoning rates in Table II.

As shown from Table II, our proposal achieves good performance because it achieves 88% accuracy even in 1% poisoning rate and achieves more than 94% accuracy at more than 2% poisoning rate. The performance of the proposal increases in proportion to the poisoning rate and it achieve quite high distinguish performance at 8% and 9% poisoning rate. At greater than 10% poisoning rate, one outlier dominated data block has generated and the outlier dominated data block degrades performance at greater than 10% poisoning rate area. We think that an advantage of our proposal comes from evaluating with data block level. In practical additional training, we add data at block level and not at individual sample level so that distinguishing at data block level may become a moderate assumption in practical viewpoint.

Table III shows detection performance of Curie under different poisoning rates. Poisoning rate 0% means "all data are clean data" so that precision becomes not a number due to both no True Positive and False Positive samples. Compared to our proposal shown in Table II, Curie increases its performance in proportion to the poisoning rate especially in a precision viewpoint, but an increment of accuracy is comparatively slow. This characteristic comes from that Curie perform detection in each sample granularity but our proposal performs detection in block of data granularity. So, there is a possibility that Curie increases performance if we treat each block of data as one sample (e.g., set a virtual averaged sample that represents the block of data). We also confirmed that Curie cannot distinguish a poisoning attack traffic session sample that have quite similar characteristic to existing clean data samples.

V. CONCLUSION AND FUTURE WORK

This paper proposes a method to identify whether newly added training data is poisoning attack data or not based on the displacement of the internal coefficient vector before and after retraining. We assumed that manipulated traffic session data for poisoning attack will largely confuse the internal coefficient vector which is an internal state of ML classifier. Thus, if the internal coefficient vector displaces largely after retraining with newly added data, we estimate that the newly added data is the poisoning attack data. We also proposed how to define the threshold value from the existing clean data.

We evaluated our proposal with SVM classifier, Biggio's SVM poisoning algorithm, and Kyoto 2016 Dataset and

compered with the existing method Curie. By utilizing SVM for classifier, the internal coefficient vector is represented as the gradient coefficient vector of hyperplane in SVM classifier. We confirmed that our proposal can detect poisoning attack data effectively and achieves 0.9838 F1 score at 8% poisoning rate in best case. This performance may come from our method treat and evaluate additional training data at data block level. On the other hand, Curie gives moderate performance because Curie evaluates at individual traffic session sample level so that it cannot distinguish the poisoning attack traffic session sample that have quite similar characteristic to an existing clean data samples.

For the future extension, our proposal may give good performance in the other ML algorithms so that we want to evaluate this method with different ML-NIDS algorithms. Furthermore, we want to apply this method to some other cybersecurity area such as malware detection/classification, spam detection/classification, process activity detection/classification, and so on.

ACKNOWLEDGMENT

This work was partially supported by JSPS KAKENHI Grant Numbers 23K28086 and 24K14959.

REFERENCES

[1] R. Tada, R. Kobayashi, H. Shimada, and H. Takakura, "Generating Kyoto 2016 Dataset for NIDS Evaluation (in Japanese)", *Journal of Information Processing*, vol. 58, no. 9, pp. 1450–1463, Sep. 2017.

[2] "Traffic Data from Kyoto University's Honeypots", [Online]. Available: https://www.takakura.com/Kyoto_data/.

[3] B. Biggio, B. Nelson, and P. Laskov, "Poisoning Attacks against Support Vector Machines", in *Proceedings of the 29th International Conference on Machine Learning (ICML '12)*, Jul. 2012, pp. 1807–1814.

[4] R. Laishram and V. V. Phoha, "Curie: A method for protecting SVM Classifier from Poisoning Attack", in *arXiv e-prints, arXiv:1606.01584*, Jun. 2016.

[5] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion Detection Using Neural Networks and Support Vector Machines", in *Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02)*, vol. 2, May 2002, pp. 1702–1707.

[6] Sophos Ltd., "Sophos UTM / Deep Learning Protection", [Online]. Available: https://www.sophos.com/en-us/products/unified-threat-management.

[7] The UCI KDD Archive, "KDD Cup 1999 Data", [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[8] MIT Lincoln Laboratory, "1999 DARPA Intrusion Detection Evaluation Dataset", [Online]. Available: https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset.

[9] J. Song *et al.*, "Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation", in *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS '11)*, Apr. 2011, pp. 29–36.

[10] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Addressing Adversarial Attacks Against Security Systems Based on Machine Learning", in *Proceedings of 11th International Conference on Cyber Conflict (CyCon 2019)*, May 2019, pp. 1–18.

[11] Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and B. Xi, "Adversarial Support Vector Machine Learning", in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2012, pp. 1059–1067.

[12] J. Steinhardt, P. W. Koh, and P. Liang, "Certified Defenses for Data Poisoning Attacks", in *Proceedings of 31st Neural Information and Processing Systems (NIPS '17)*, Dec. 2017, pp. 3520–3532.

[13] R. Taheri *et al.*, "On Defending Against Label Flipping Attacks on Malware Detection Systems", *Neural Computing and Applications*, vol. 32, pp. 14 781–14 800, Jul. 2020.

[14] H. Shimada, S. Su, H. Hasegawa, and Y. Yamaguchi, "Gradient Variation based Poisoning Attack Data Detection for Poisoning Attacks Targeting SVM based Malware Detection (in Japanese)", Information Processing Society Japan, Technical Reports 2022-CSEC-98, Jul. 2022, pp. 1–8.

[15] M.-I. Nicolae *et al.*, "Adversarial Robustness Toolbox v1.0.0", Nov. 2019, [Online]. Available: https://github.com/Trusted-AI/adversarial-robustness-toolbox.

[16] T. Head, K. Kelly, and A. C. Mayes, "scikit-optimize: v0.5.1.", Mar. 2018, [Online]. Available: https://github.com/scikit-optimize/scikit-optimize.