Proc. of International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA 2024)
1-2 February 2024, Victoria-Seychelles

2024 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA) | 979-8-3503-9452-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ACDSA59508.2024.10467929

A Heuristics and Machine Learning Hybrid Approach to Adaptive Cyberattack Detection

Makoto Iwabuchi
NITORI Co., Ltd.
Kita-ku, Tokyo, Japan
makoto01401@gmail.com

Akihito Nakamura
Computer Science Division University of Aizu
Aizu-Wakamatsu, Fukushima, Japan
nakamura@u-aizu.ac.jp

Abstract—Cybersecurity is more significant now than ever, and the severity of the threat has escalated. One possible countermeasure is the Intrusion Detection and Prevention System (IDPS), which enables the detection of malicious activities in the network based on signature-matching and other detection methods. A signature represents the specific pattern of an attack. However, it occasionally misses malicious traffic or raises false alerts when the detection method is not carefully configured with the latest information. That is, it is susceptible to false positives or false negatives. This paper presents a highly accurate cyberattack detection method with the automatic generation of tailored signatures for a rapid response to emerging threats. We combine heuristics for known attacks and machine learning (ML) techniques to detect unforeseen attack patterns in traffic, i.e. a hybrid method. Rule-based judgment for heuristics and anomaly detection for ML are used, respectively. This study introduces a novel approach by employing machine learning with a packet-to-image conversion technique. We convert network packet data into images and utilize the image data for training and classifying attack patterns. By transforming the problem to anomaly detection in image data, the evaluation results revealed that the method has high accuracy.

Index Terms—cybersecurity, IDPS, honeypot, machine learning, GAN, LSTM, PNG images

I. INTRODUCTION

Cybersecurity is more significant now than ever, and the severity of the threat has escalated. One possible countermeasure is the Intrusion Detection and Prevention System (IDPS), which enables the detection of malicious activities in the network based on signature-matching and other detection methods. A signature represents the specific pattern of an attack. However, it occasionally misses malicious traffic or raises false alerts when the detection method is not carefully configured with the latest information. That is, it is susceptible to false positives or false negatives. Also, the creation of signatures requires in-depth knowledge of network and security.

A signature-matching method is one of the IDPS anomaly detection methods, which detects attacks based on whether they match a predefined pattern. This method has the advantage of being able to detect anomalies with a very high accuracy rate against known attacks. On the other hand, it has the disadvantage that it is difficult to detect malicious traffic whose characteristics are unknown.

This paper presents a highly accurate cyberattack detection method with the automatic generation of tailored signatures for a rapid response to emerging threats. We combine heuristics for known attacks and machine learning (ML) techniques to detect unforeseen attack patterns in traffic, i.e. a hybrid method. Rule-based judgment for heuristics and anomaly detection for ML are used, respectively. This study introduces a novel approach by employing ML with a packet-to-image conversion technique. We convert network packet data into images and utilize the image data for training and classifying attack patterns. By transforming the problem to anomaly detection in image data, the evaluation results revealed that the method has high accuracy.

Several studies utilize machine learning techniques for detecting unknown attacks [1]–[5]. Compared to them, our method and system offer superior detection accuracy. In addition, the proposed system design enables rapid response to unknown attacks. When a new attack pattern is detected, the system generates specific signatures on demand to block subsequent activities. As a result, the system enables real-time adaptive security.

The remainder of this paper is organized as follows. Section II introduces the IDPS system design. Section II describes the proposed cyberattack detection method. In section IV, we evaluate the performance of the method. Section V reviews the related work and section VI concludes the paper.

II. IDPS DESIGN FEATURES

In this section, we discuss the design features of our IDPS.

A. System Architecture

Fig. 1, shows an overview of the system architecture. On the edge of a network, an IDPS analyzes incoming network traffic by signature-matching. The IDPS compares incoming packets with predefined attack patterns, known as signatures. The defined prevention method in the

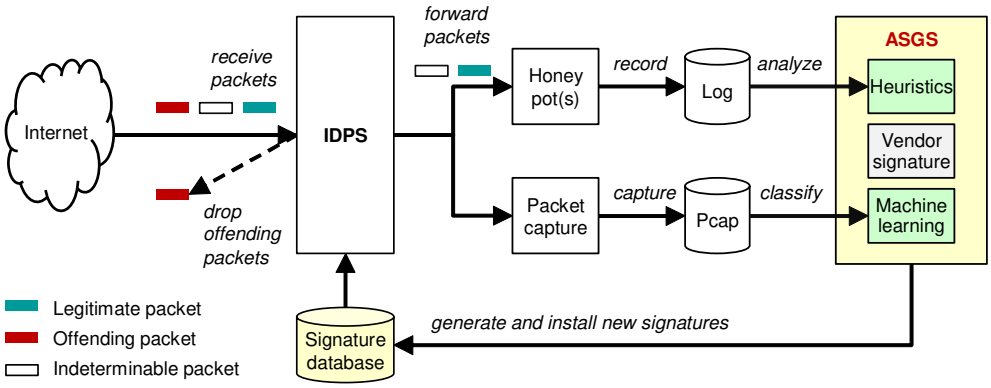

Fig. 1. System Architecture

signatures involves dropping matched packets and raising alerts.

In the second tier, honeypot(s) capture the packets evading the IDPS and log the information. A honeypot is a software that dares to put on the software that remains vulnerable and receives attacks to observe and record logs. It enables the discovery of new attack patterns and the latest trends in attacks. The log is analyzed by the heuristics-based classifier. The packet capture component records the whole packets for later reference by the ML-based classifier. Here, we assume the use of the de facto standard PCAP format for recording, commonly employed in network packet capture and analysis [6]. In the later sections, we discuss the details of these classifiers.

B. Adaptive Signature Generation

The core of the system, at the third tier, is the Automatic Signature Generation System (ASGS). Naturally, we leverage the signatures provided by the IDPS vendor. However, signature updates provided by IDPS vendors are relatively infrequent and lack immediacy. On the other hand, the ASGS generates signatures promptly by analyzing the honeypot log and the captured packets constantly. When a new attack pattern is detected, the ASGS generates specific signatures on demand and immediately installs them on the database to block subsequent activities. As a result, the system enables real-time adaptive security.

III. CYBERATTACK DETECTION METHODS

This section discusses the specific methods to detect cyberattacks in the network.

A. Heuristics Method

The signature-matching method is currently the most popular for detecting cyberattacks. A signature represents the specific pattern of an attack. The creation of signatures requires in-depth knowledge of network and security. Therefore, each IDPS provider creates signatures and periodically adds them to the signature database of the IDPS software.

In our current design, a few simple conditions and baselines are used based on heuristics. The ASGS takes two types of conditions: IP address and protocol as shown in Fig. 2. First, the IP address is obtained from the honeypot log and determine if it is suspicious. For example, if the originator host is located overseas. In addition, protocol-specific conditions are deployed. For example, the number of login trials and errors per unit of time is used to identify brute-force attacks on SSH or Telnet (process A). Regarding HTTP, the payload of access is investigated, including path-traversal, admin login trial, SQL injection (process B). If one of these conditions is newly met, the ASGS will generate a new signature based on the identified criteria, enhancing its ability to detect specific cyber threats.

B. Machine Learning Method

In addition to heuristics, we employ a machine learning (ML) based method to enhance attack detection accuracy, creating a hybrid approach. The shape of the idea is packet visualization. Since ML algorithms are very good at image classification, we transform the problem into anomaly detection in image data. That is, image data are used for training and classification of cyberattack patterns.

1) Packet-to-Image Conversion: Here, we discuss how to generate image data that represent network traffic. As previously stated, network packets are captured and stored in the PCAP format (Fig. 1). Therefore, the PCAP files are used as the input to the packet visualization. For output, the standard Portable Network Graphics (PNG) format is used as the raster image file. PNG format is the standard format for raster image files [7].

The steps in the conversion process are shown below and in Fig. 3. Let integer N be the length, in pixels, of one side of the square image to be generated.

- Step 1 (vertical packet extraction): The system reads the first N packets $P_1, P_2, ..., P_N$ from the input PCAP file.

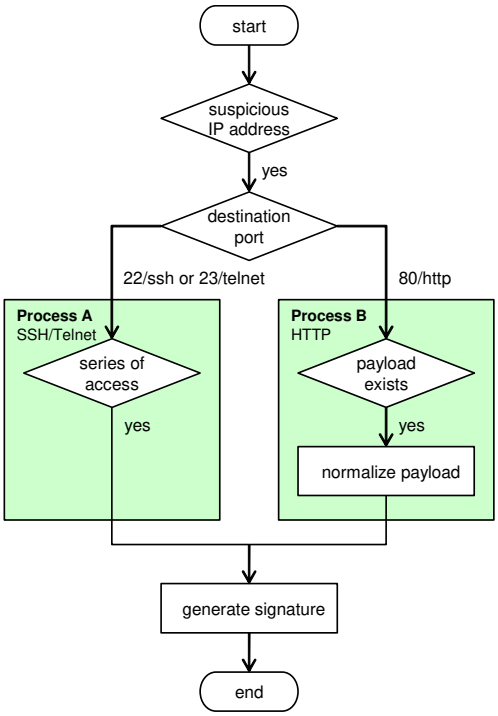

Fig. 2. Heuristics-based Signature Generation Flow (IP Address and Protocol)

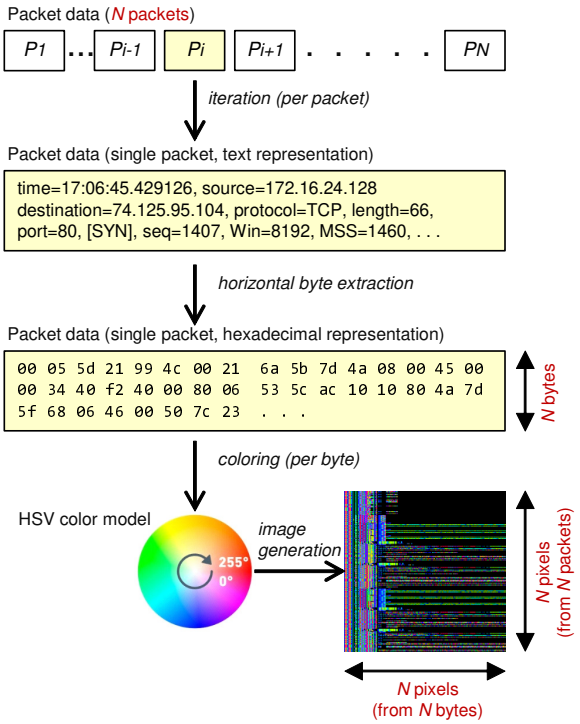

Fig. 3. Packet-to-Image Conversion

- Step 2 (horizontal byte extraction): For each packet P_i, the first N bytes $B_1, B_2, ..., B_N$ are obtained. If the length of P_i is shorter than N bytes, a padding of zeros is appended at the end.
- Step 3 (coloring): For each byte B_j of P_i, it is interpreted as an integer between 0 and 255 and the value is mapped to a color C_{ij} in the hue ring of HSV (hue, saturation, value) color model [8]. The value is compressed from 360 degrees to 255 degrees in the ring and assigned to a circular coordinate. That is, B_j of P_i is converted to one pixel of color.
- Step 4 (image generation): A colored square of $N \times N$ pixels is obtained with the horizontal direction consisting of a sequence of N packet bytes and the vertical direction consisting of N packets.
- Step 5 (iteration): The system reads the next N packets and repeats the above steps to create a colored square image. Once all packet data have been processed, this process is complete.

In our method, we set N to 256 for two reasons. First, the size covers significant information about each network packet. The size of the normal IPv4 header is 20 bytes and the maximum is 60 bytes and the size of the IPv6 fixed header is 20 bytes. The data part typically contains upper-layer protocol data, such as TCP or UDP. TCP headers range from 20 to 60 bytes, while UDP headers are 8 bytes. These headers include source and destination IP addresses, source and destination port numbers which mean application layer protocols, and control information. In addition, at least 136 bytes of application data are included. The second reason is that the size affects the performance of machine learning algorithms. The size 256 was the best in the preliminary experiments.

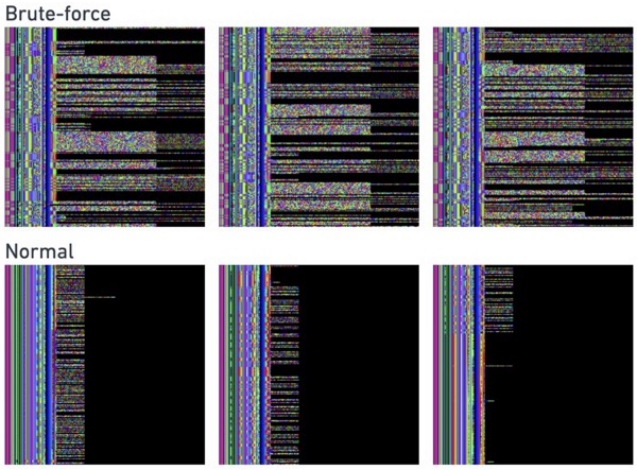

Fig. 4. Examples of Generated Images (SSH Brute-Force Attacks)

Fig. 4 displays example images generated using the conversion method. All these images were generated from SSH traffic data. At the top of the figure, three images depict SSH brute-force attacks characterized by successive login attempts. In contrast, the images at the bottom represent normal traffic patterns.

2) Training Efficient GAN: This study employs Efficient GAN, a type of generative adversarial network (GAN) [3], [9]. GANs have been successfully used to model complex and high-dimensional data in ML algorithms. Efficient GAN improves the performance of conventional GANs and is suitable for solving anomaly detection problems with large datasets and real-time applications.

For training an Efficient GAN, the traffic data of normal communication is imaged using the aforementioned method. In training, the following steps are repeated in one epoch.

- Step 1: Create a false image from latent variables in the generator.
- Step 2: Create latent variables from real images with the encoder.
- Step 3: Training the discriminator to classify latent variables and images as real or fake.
- Step 4: Create fake images from latent variables with the generator.
- Step 5: Training the generator so that the discriminator's classification results are wrong.
- Step 6: The encoder creates latent variables from the real images.
- Step 7: Training the encoder so that the discriminator's classification results are wrong.

Fig. 5 displays the overall ML-based anomaly detection flow with the signature generation step.

C. Implementation

To implement the system, open-source software Suricata, T-Pot, and tcpdump were utilized for IDPS, honeypot, and packet capture, respectively [10]–[12]. The ASGS software is written in Python language and Google Colaboratory is used for Efficient GAN training.

Fig. 6 shows example signatures generated by the ASGS. Signature S1 describes a rule to drop packets if the source IP address is the specified one and the destination port is 22, i.e., SSH. Similarly, signature S2 defines a rule for Telnet. Signature S3 describes a protocol-based rule, dropping HTTP requests to the specified content.

IV. EVALUATION

In this section, we show the evaluation results of the proposed method.

A. Datasets

We used the following three datasets to verify the performance of the method. Table I presents the summary statistics to provide a comprehensive overview.

1) BOS Dataset from MWS 2019 [13]: This dataset is provided by the research community. It includes packet data on observed malware breach activities to an organization's internal network. We used a part of this dataset which was observed from 2017 to 2018. Of the approximately 700 thousand packets, 550 thousand were used for training non-malicious traffic.

2) Stratosphere IPS Dataset [14]: This dataset is composed of normal traffic in HTTP and anomalous traffic from malware. The normal data is collected by accessing the Alexa rank TOP 1000 sites in 2017. The anomalous data are recorded traffic from malware-infected PCs connecting for short periods to several IP addresses, with a large number of requests being sent.

3) Original Dataset: In addition to the above open datasets, we created an original dataset. We generated the anomalous data by executing simulated SSH and HTTP attacks using remote login cracker Hydra on a virtual machine. 100 IP addresses for brute-force attacks against an SSH server and 50 IP addresses for normal communication were used with general commands and processing such as ping and file operations. On HTTP, 50 IP addresses were used for brute-force attacks against a web login form. In addition, 50 IP addresses were used for normal communication through manual login, page transitions, and other normal activities.

TABLE I.
Datasets Used for Training and Testing: Number of Packets and Number of Images Generated

Dataset	Task	Class	Packets	Images
BOS	train	normal	550,000	21,000
	test	normal	25,600	100
		anomaly	25,600	100
Stratosphere IPS	train	normal	2,616,990	10,222
	test	normal	33,280	130
		anomaly	18,523	72
Original (SSH)	train	normal	76,800	300
	test	normal	23,040	90
		anomaly	32,500	126
Original (HTTP)	train	normal	476,160	1,860
	test	normal	25,600	100
		anomaly	25,600	100

B. Evaluation Metrics

Common metrics that can be used to measure the performance of a binary classifier are shown in Table II. They can be calculated from the number of test results in the following four categories.

- TP (True Positive): A correct classification where the test result accurately indicates the presence of a condition. In other words, malicious activity is correctly classified as malicious.
- FP (False Positive): A classification error where the test result inaccurately indicates the presence of a condition. In other words, benign activity is mistakenly considered malicious.
- TN (True Negative): A correct classification where the test result accurately indicates the absence of a

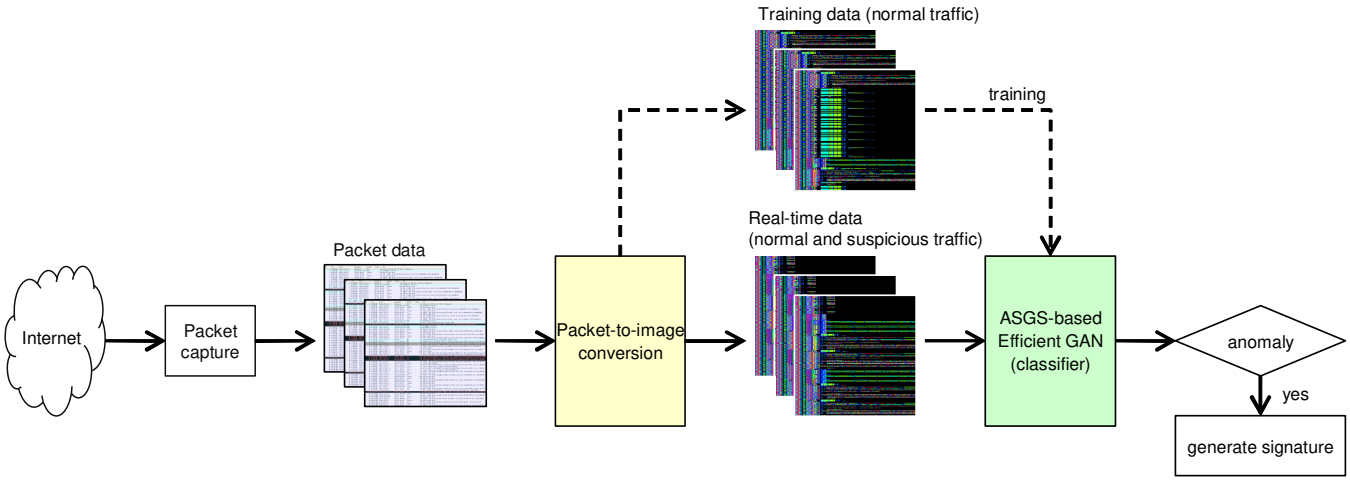

Fig. 5. ML-based Signature Generation Flow (GAN-Based Deep Learning)

```
# Signature S1
drop tcp 118.XXX.YYY.ZZZ any -> $HOME_NET 22
(msg:"SSH/22 Alert"; sid:100832; rev:1;)

# Signature S2
drop tcp 61.XXX.YYY.ZZZ any -> $HOME_NET 23
(msg:"Telnet/23 Alert"; sid:100833; rev:1;)

# Signature S3
drop tcp $EXTERNAL_NET any -> $HOME_NET 80
(msg:"  TP/80 Alert";
conten  "/wp-content/plugins/wp-file-manager/readme.txt";
nocase  sid:100854; rev:1;)
```

Fig. 6. Generated IDPS Signatures (Examples)

condition. In other words, benign activity is correctly classified as benign.

- FN (False Negative): A classification error where the test result inaccurately indicates the absence of a condition. In other words, malicious activity is mistakenly considered benign.

TABLE II
Evaluation Metrics

Metrics	Formula and Meaning
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$ The proportion of correct predictions among the total number of cases examined.
Recall	$\frac{TP}{TP+FN}$ The proportion of cases that are predicted as positive and are positive of all the positive cases.
Precision	$\frac{TP}{TP+FP}$ The proportion of cases that are predicted as positive and are positive out of all the positive results.
F-measure	$2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$ The harmonic mean of precision and recall.

C. Experimental Results

In this section, we show the evaluation results of the method.

1) Experiment A: First, the performance of the Efficient GAN was evaluated using the BOS Dataset from MWS 2019. The graph in Fig. 7 shows the Receiver Operating Characteristic (ROC) curve, depicting the true positive rate (TPR) against the false positive rate (FPR) at various discrimination threshold settings. It illustrates the diagnostic ability of a classifier. The measurement result revealed an Area Under the Curve (AUC) of approximately 0.772. AUC values range from 0.0 to 1.0; the closer the AUC is to 1.0, the better the classifier performs. That is, the result shows relatively good performance.

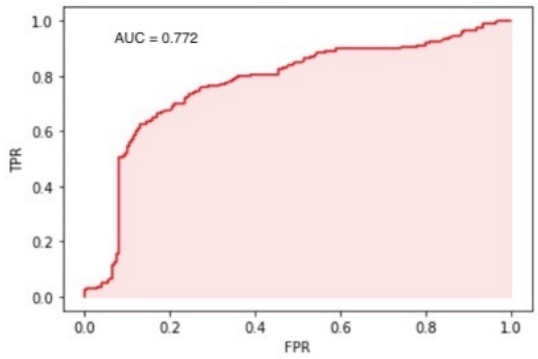

Fig. 7. Area Under Curve (AUC) of the Efficient GAN: True Positive Rate (TPR) against False Positive Rate (FPR)

Next, we compared the results with other methods using the four metrics in Table III. As a result, the detection performance was found to be inferior to that of Long Short-Term Memory (LSTM) [16] and anomaly detection using a combination of Denoising Autoencoder (DAE) and Double Deep Q Network (DDQN) [5]. There are two potential reasons for this. First, the proposed method captures

TABLE III.
Experiment Result A: BOS Dataset

Method	Accuracy	Precision	Recall	F-measure
DAE+DDQN [5]	0.855	0.465	0.812	0.789
DCGAN [15]	-	-	-	-
LSTM [16]	0.930	0.800	0.840	0.810
Efficient GAN	0.745	0.735	0.750	0.742

the entire traffic, and thus the features may not be well represented through imaging. Second, the BOS Dataset lacks distinctive features in normal and abnormal traffic, resulting in lower detection performance compared to the existing methods. The authors of the reference paper [15] concluded that Deep Convolutional GAN (DCGAN) could not capture malware-specific features in the BOS Dataset because the attacks are not characterized in this dataset. Given that the AUC of this method using DCGAN ranged between 0.54 and 0.60, the Efficient GAN demonstrated superior performance.

2) Experiment B: Next, we observed the performance of the Efficient GAN using the Stratosphere IPS Dataset. Table IV provides a comparison of the methods based on the reference paper [17], which utilized a similar Stratosphere IPS for evaluation.

TABLE IV.
Experimental Result B: Stratosphere IPS Dataset (adapted [4] Table 2)

Model	Precision	Recall	F-measure
DSEBM-r	0.425	0.425	0.425
DSEBM-e	0.437	0.437	0.437
DAGMM	0.652	0.652	0.652
AE	0.654	0.654	0.654
AnoGAN	0.468	0.468	0.654
ALAD	0.699	0.700	0.700
Efficient GAN	0.783	0.771	0.777

The Efficient GAN was 8–9 percent more accurate than Adversarially Learned Anomaly Detection (ALAD) [3], the best one in the existing methods. This difference in accuracy can be attributed to how the score threshold is determined. ALAD incorporates abnormal samples into the normal training dataset and sets the maximum anomaly score as the threshold. In this scenario, the threshold value depends on the abnormal samples, highlighting the significant impact of the threshold calculation method on accuracy.

3) Experiment C: Finally, we evaluated the performance of the combined detection methods. For SSH attacks, both the heuristics and Efficient GAN achieved approximately 80% accuracy individually, while the combined method exhibited a remarkable 95% accuracy. Regarding HTTP attacks, the heuristics method demonstrated high accuracy independently. The F-measure scores of the combined method were around 95%. These results indicate that the combined method maintains a stable balance between accuracy and false positives, showcasing its robust detection capability.

TABLE V.
Experimental Result C: Original Dataset

Protocol Method	Accuracy	Precision	Recall	F-measure
SSH				
Heuristics	0.83	1.00	0.66	0.80
Efficient GAN	0.87	0.89	0.84	0.86
Combined	0.95	0.97	0.92	0.94
HTTP				
Heuristics	0.99	0.98	1.00	0.99
Efficient GAN	0.94	0.96	0.92	0.94
Combined	0.96	1.00	0.92	0.96

V. RELATED WORK

There are various approaches to detecting anomalies in network traffic using machine learning [1]. Lopez-Martin et al. [2] used an integrated method of a recurrent neural network (RNN) and a convolutional neural network (CNN) for anomaly detection in IoT traffic. Their best model achieved an accuracy of 0.9632. Urakawa et al. [16] proposed Long Short-Term Memory (LSTM), a method for detecting network anomalies by training feature vectors extracted from packet headers. The method showed a high accuracy rate of 0.930 on the MWS dataset [13]. Hommaru and Terada [5] used an autoencoder, Denoising Autoencoder (DAE) for feature extraction. Also, the Double Deep Q Network (DDQN) approach of deep reinforcement learning was used for training and predicting the transformed features to detect unknown attacks. The evaluation used the NSL-KDD dataset. The results show that the method has an average accuracy rate of 0.657, which is 6-12% higher than the existing method.

Zenati et al. [3] utilized GAN for anomaly detection, especially for a network intrusion dataset, and revealed a good performance at test time on the KDD99 dataset. Hioki et al. [15] proposed a method utilizing DCGAN trained on normal traffic images to classify normal and anomalous communications. In the experiment using the BOS dataset, only the attacks with clear differences between normal and anomalous cases could be well classified: F5 attack, host scan, and SYN flood.

VI. CONCLUSION

The objective of this study was to propose a cyberattack detection method with high accuracy and adaptability. To enhance accuracy, a combined approach of heuristics and machine learning (ML) was adopted, where the rule-based judgment was employed for heuristics and anomaly

detection for ML. In the realm of ML, a novel method was introduced, involving the conversion of packets into images. Network packet data undergoes this transformation, and the resulting image data is utilized for the training and classification of attack patterns. By transforming the problem of cyberattack detection into anomaly detection in image data, the proposed method demonstrated high accuracy. Further research is needed to establish better detection performance against emerging unknown cyberattacks.

Additionally, this study presents the design of an Intrusion Detection and Prevention System (IDPS) with real-time adaptability. When a new attack pattern is detected, the system generates specific signatures as needed and promptly installs them to block subsequent activities. As a result, the system achieves real-time adaptive security. Beyond the improvement in detection accuracy, the system's features can reduce the likelihood of detection omissions.

References

[1] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," IEEE Communications Surveys & Tutorials, vol. 10, no. 4, pp. 56–76, 2008.

[2] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," IEEE Access, vol. 5, pp. 18 042–18 050, September 2017. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2747560

[3] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially learned anomaly detection," in Proceedings of IEEE International Conference on Data Mining (ICDM). IEEE, 2018, pp. 727–736.

[4] T. Truong-Huu, N. Dheenadhayalan, P. Pratim Kundu, V. Ramnath, J. Liao, S. G. Teo, and S. Praveen Kadiyala, "An empirical study on unsupervised network anomaly detection using generative adversarial networks," in Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence (SPAI'20). ACM, 2020, pp. 20–29. [Online]. Available: https://doi.org/10.1145/3385003.3410924

[5] M. Hommaru and M. Terada, "Proposing a method for detecting unknown attacks in nids using machine learning," IPSJ Journal, vol. 62, no. 12, pp. 1915–1925, December 2021.

[6] M. C. Richardson, "Pcap capture file format," December 2020. [Online]. Available: https://pcapng.github.io/pcapng/draft-gharris-opsawg-pcap.html

[7] T. Boutell, "PNG (Portable Network Graphics) Specification Version 1.0," RFC 2083, March 1997. [Online]. Available: https://www.rfc-editor.org/info/rfc2083

[8] F. Ganovelli, M. Corsini, S. Pattanaik, and M. D. Benedetto, Introduction to Computer Graphics: A Practical Learning Approach. CRC Press, October 2014.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014), 2014, pp. 2672–2680.

[10] The Open Information Security Foundation (OISF), "Suricata," 2023. [Online]. Available: https://suricata.io/

[11] "T-pot," 2023. [Online]. Available: https://github.com/telekom-security/tpotce

[12] The Tcpdump Group, "tcpdump," 2023. [Online]. Available: https://www.tcpdump.org/

[13] M. Hatada, M. Akiyama, T. Matsuki, and K. Takahiro, "Empowering anti-malware research in japan by sharing the mws datasets," Journal of Information Processing, vol. 23, no. 5, pp. 579–588, September 2015.

[14] Stratosphere, "Stratosphere laboratory datasets," 2015. [Online]. Available: https://www.stratosphereips.org/datasets-overview

[15] Y. Hioki, S. Aoki, and T. Miyamoto, "Anomaly detection of network traffic using dcgan," in Computer Security Symposium. Information Processing Society of Japan, 2018, pp. 341–347, written in Japanese.

[16] Y. Urakawa, S. Aoki, and T. Miyamoto, "Anomaly detection of network using lstm," in Computer Security Symposium. Information Processing Society of Japan, 2019, pp. 422–429, written in Japanese.

[17] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," in Proceedings of The 33rd International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, June 2016, pp. 1100–1109. [Online]. Available: https://proceedings.mlr.press/v48/zhai16.html