XAI-driven adversarial attacks on network intrusion detectors

Satoshi Okada, Houda Jmila, Kunio Akashi, Takuho Mitsunaga, Yuji Sekiya, Hideki Takase, Gregory Blanc, Hiroshi Nakamura

▶ **To cite this version:**

Satoshi Okada, Houda Jmila, Kunio Akashi, Takuho Mitsunaga, Yuji Sekiya, et al.. XAI-driven adversarial attacks on network intrusion detectors. European Interdisciplinary Cybersecurity Conference (EICC), Jun 2024, Xanthi, Greece. pp.65-73, ⟨10.1145/3655693.3655714⟩. ⟨hal-04660625⟩

HAL Id: hal-04660625

https://hal.science/hal-04660625v1

Submitted on 24 Jul 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

XAI-driven Adversarial Attacks on Network Intrusion Detectors

Satoshi Okada
The University of Tokyo
Tokyo, Japan
okada@hal.ipc.i.u-tokyo.ac.jp

Houda Jmila
Université Paris-Saclay, CEA, List
Palaiseau, France
houda.jmila@cea.fr

Kunio Akashi
The University of Tokyo
Tokyo, Japan
k-akashi@si.u-tokyo.ac.jp

Takuho Mitsunaga
INIAD, Toyo University
Tokyo, Japan
takuho.mitsunaga@iniad.org

Yuji Sekiya
The University of Tokyo
Tokyo, Japan
sekiya@nc.u-tokyo.ac.jp

Hideki Takase
The University of Tokyo
Tokyo, Japan
takasehideki@hal.ipc.i.u-tokyo.ac.jp

Gregory Blanc
SAMOVAR, Télécom SudParis,
Institut Polytechnique de Paris
Palaiseau, France
gregory.blanc@telecom-sudparis.eu

Hiroshi Nakamura
The University of Tokyo
Tokyo, Japan
nakamura@hal.ipc.i.u-tokyo.ac.jp

ABSTRACT

Deep Learning (DL) technologies have recently gained significant attention and have been applied to Network Intrusion Detection Systems (NIDS). However, DL is known to be vulnerable to adversarial attacks, which evade detection by introducing perturbations to input data. Meanwhile, eXplainable Artificial Intelligence (XAI) helps us to understand predictions made by DL models and is an essential technology for ensuring accountability. This paper focuses on the relationship between the DL model's decision-making processes and adversarial examples (AEs) and proposes a new AE generation method based on XAI. Our method utilizes XAI to identify important features when making predictions and perturb them in real (traffic) space to evade detection by DL-based NIDS. We implemented our proposed method in a real-world network environment. We confirmed that our AEs completely evade detection without compromising the malicious nature of the attack communications. This experiment reveals that, unlike many existing studies, our proposed method is feasible in the traffic space.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Security and privacy**;

KEYWORDS

Adversarial Example, XAI, NIDS, Cyber Security

ACM Reference Format:
Satoshi Okada, Houda Jmila, Kunio Akashi, Takuho Mitsunaga, Yuji Sekiya, Hideki Takase, Gregory Blanc, and Hiroshi Nakamura. 2024. XAI-driven Adversarial Attacks on Network Intrusion Detectors. In *European Interdisciplinary Cybersecurity Conference (EICC 2024), June 05–06, 2024, Xanthi, Greece.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3655693.3655714

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
EICC 2024, June 05–06, 2024, Xanthi, Greece
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1651-5/24/06
https://doi.org/10.1145/3655693.3655714

1 INTRODUCTION

Machine learning technology has become more and more popular not only in research fields but also in society. It has been introduced in various applications such as image recognition, anomaly detection, text mining, and malware detection [11, 21, 26]. Among these machine learning algorithms, *deep learning* (DL), in particular, has gained significant attention for its astonishing performance, equivalent to or exceeding human capabilities (such as natural language processing and decision-making). The advancements in deep learning technology have been made possible thanks to the availability of large datasets for training neural networks, as well as the remarkable advancements in hardware technology [31].

Recently, DL technologies have been introduced into cyber security products such as Network Intrusion Detection Systems (NIDS). NIDS play an important role in detecting attackers' malicious activities in networks by monitoring network traffic. Formerly, NIDS were signature-based, which could only find known attack patterns. In contrast, by introducing DL techniques, they can now detect new and unknown attacks [17, 27, 38]. Thus, they are now receiving more and more attention.

DL models have been pointed out to be extremely vulnerable to adversarial attacks, in which attackers perturbed the input data to cause a machine learning model to make incorrect predictions [15, 18, 22]. To evaluate and improve the robustness of machine learning models against them, it is common to construct Adversarial examples (AEs) to demonstrate the upper bound of the robustness [4]. This approach has led to a notable increase in studies focused on generating AEs.

Feature-space attacks (such as FGSM [10]) where attackers directly modify the feature vectors input to the model effectively generates AEs for image recognition models. That is because the mapping from the image space (called the problem space in this paper) to the feature space is reversible or differentiable, and it is

easy to find the perturbation in the problem space corresponding to the modification in the feature space. On the other hand, NIDS process flow-based and statistical features extracted from raw network traffic data as input. It is known that this mapping from problem space (raw traffic data) to feature space in NIDS is neither invertible nor differentiable. It is referred to as the *inverse feature mapping problem*. Therefore, the feature-space attack is not directly applicable to DL-based NIDS, and different approaches are necessary to generate AEs for NIDS [29]. In addition to their feasibility in the problem space, it is also required to mutate malicious traffic without compromising its malicious nature.

Recently, the connection between eXplainable Artificial Intelligence (XAI) and AEs has been pointed out [20]. XAI offers us a way to understand the decision-making processes of DL-based models [33]. XAI has attracted more and more attention, and many XAI-related techniques have been proposed to gain insights into ML systems [1, 8]. Interpretations given by XAI can play an important role in improving the adversary's capacity and strategy. For instance, Kumagai et al. [19] propose an XAI-based method for generating AEs in the image domain, which is superior to previous ones. Thus, it is also expected that XAI is utilized to generate AEs for DL-based NIDSs.

1.1 Contribution

In this paper, we propose a new XAI-based method for generating AEs for DL-based NIDS. In our proposed method, we treat the target NIDS model as a white box and analyze False Negative (FN) samples –those classified as benign despite being malicious– by utilizing XAI. Through the analyses, we identify features significantly contributing to detection evasion and determine how they should be perturbed. By focusing on important features and minimizing the number of perturbed features, we address the *inverse feature mapping problem*. Specifically, we find feasible transformations in the problem space that correspond to the perturbations in the feature space. This approach enables us to generate highly evasive AEs by fully utilizing the feature space information. Furthermore, we verified whether attacks perturbed by our proposed method could evade detection by implementing them in a real-world network environment. We perturbed two types of network attacks, and both perturbed attacks bypassed NIDS detection at rates of about 96% and 100%, respectively. These findings clearly proved the feasibility of our proposed method and its high rate of evasion from detection.

1.2 Organization of the Paper

We explain the background of our research, such as adversarial attacks and XAI, in Section 2. Then, we introduce related research in Section 3. Section 4 describes our proposals. We provide the experimental settings and describe the results in Section 5. Section 6 concludes this paper.

2 BACKGROUND

2.1 Network Intrusion Detection Systems

NIDS are designed to monitor network traffic for suspicious activities and potential threats. Unlike Host-based Intrusion Detection Systems (HIDS), which are installed on individual computers to monitor inbound and outbound packets only from that particular host, NIDS are deployed at strategic points within the network to inspect all traffic in the network [3]. This makes NIDS particularly effective in detecting attacks that might not be visible at the host level, such as distributed denial-of-service (DDoS) attacks. Traditional NIDS relied on signature-based detection methods, which compare network traffic against a database of known attack signatures. However, the rise of sophisticated and novel attacks has forced researchers to develop more advanced systems. Deep Learning (DL)-based NIDS have emerged as a potent solution due to their ability to learn from data, recognize complex patterns, and detect anomalies [6].

An important aspect of DL-based NIDS is its classification capability. They can be configured for binary classification, distinguishing between benign and malicious traffic, or for multi-classification, which involves identifying the specific type of attack. Depending on the required capability of NIDS, training methods also differ. For instance, supervised learning methods with mixed datasets containing both benign and malicious traffic are often adopted to train multi-classification models. Meanwhile, if NIDS models conduct anomaly detection, they can be trained by datasets consisting solely of legitimate traffic.

2.2 Adversarial Examples

AEs cause misclassification in a machine learning model by manipulating input data. The attacker successfully manipulates the input data to cross a decision boundary, causing that input data to be misclassified. This attack can be formulated as follows [37].

$$\begin{aligned} \text{minimize} \quad & \|x' - x\| \\ \text{subject to} \quad & f\left(x'\right) = l', \\ & f(x) = l, \\ & l \neq l', \\ & x' \in [0,1]^m, \end{aligned}$$

where $x \in [0,1]^m$ is an input to a classifier f, l is the correctly predicted class for x, and $l' \neq l$ is the target class for $x + r$, with $r \in [0,1]^m$ being a small perturbation to x.

Adversarial attacks are also classified based on the information available to the attacker. If an attacker knows all information, including input and output data, as well as the weights and classification labels of the target model, the attack is deemed a *white-box attack*. On the other hand, an attack conducted under conditions where the attacker only has access to information about the input/output data is called a *black-box attack*. *Gray-box attacks* lie in between white-box and black-box attacks, where the attacker possesses partial knowledge or limited access to the victim model.

2.3 Explainable Artificial Intelligence

In recent years, DL-based systems are increasingly being introduced across several domains. Furthermore, the architecture of DL models has become more complex in order to improve their performance. With the frequent utilization of such complex DL systems, there is an urgent need to understand their decision-making process and to gain insights into the outcomes. That is why XAI has gained much attention these days. XAI provides us with some helpful

information to understand the decision-making process of ML or DL systems.

Integrated Gradients [36], a method introduced for attributing predictions of deep neural networks to their inputs. Formally, let a function $F : \mathbb{R}^n \rightarrow [0, 1]$ representing a deep neural network, an input $x \in \mathbb{R}^n$, and a baseline $x' \in \mathbb{R}^n$. The baseline represents a reference point that satisfies $F(x') \approx 0$. Most deep neural networks have a natural baseline in the input space where the prediction is neutral. For instance, in the object recognition area, it is the black image. Integrated Gradients are calculated by accumulating the gradients along the straight-line path in $\mathbb{R}^n$ from x' to x. The integrated gradient (IG) along the i^{th} dimension for an input x and baseline x' is defined as

$$IG_i(x) = \left(x_i - x_i'\right) \times \int_{\alpha=0}^{1} \frac{\partial F\left(x' + \alpha \times (x - x')\right)}{\partial x_i} d\alpha.$$

The value represents the contribution of x_i. For example, in object recognition, we can see which pixels of the image were responsible for a certain label being predicted. This method also satisfies key axioms such as *completeness*, ensuring that attributions sum up to the difference between the outputs at x and x', and *implementation invariance*, making it a robust tool for interpreting complex DL models. In this paper, we adopt Integrated Gradients as an XAI model because it relies only on calculating gradients, a fundamental aspect of neural networks that is independent of the type of data or the specific architecture of the model. Thus, this can be applied to the analyses of our targeted NIDS model, which utilizes tabular data.

3 RELATED RESEARCH

We categorize and introduce existing research on adversarial examples for DL-based NIDS into feature-space attacks and problem-space attacks. This categorization allows us to clarify the differences between our study and previous works, thereby highlighting our contributions.

In the field of adversarial attacks targeting ML-based NIDS, feature-space attacks assume the ability of attackers to modify feature vectors input to NIDS directly. Starting with white-box attacks, existing gradient-based AE generation algorithms were applied to evade a DL-based NIDS [25, 39]. Techniques for bypassing a particular NIDS model, Kitsune [27], were proposed by Clements et al. [7]. Additionally, strategies for circumventing GAN-based NIDS detection are introduced by Piplai et al. [30]. There also exist gray- and black-box attacks. A boundary-based method designed to produce AEs for DoS attacks was proposed by Peng et al. [28], and a method for generating AEs against botnet detectors by introducing random mutations to features was presented by Apruzzese et al. [2]. Lin et al. [23] developed a GAN-based approach to generate AEs without any knowledge of the NIDS's internal structure or parameters.

Problem-space attacks directly modify or transform network traffic to evade detection. Hashemi et al. [14] proposed a white-box attack method for multiple NIDS models. Their maximum evasion rate for this proposed method in flow-based NIDS is limited to 68%. Regarding gray-box attacks, Stinson et al. [35] proposed techniques that evade botnet detection by introducing random mutations, and Homoliak et al. [16] proposed random obfuscation techniques for evading the detection of various classifiers. Han et al. [13] proposed a black-box attack that preserves the maliciousness of attack communications while being generic and minimally overhead-intensive.

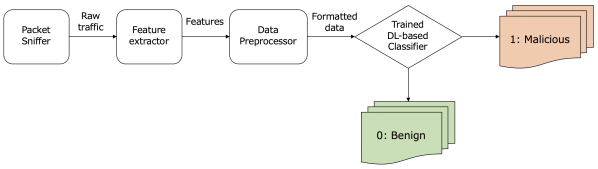

Figure 1: General flow of DL-based NIDS

First of all, feature-space attacks cannot be directly converted to actual network traffic because feature extraction in DL-based NIDSs is not always invertible [13]. Therefore, their feasibility is limited, and they are impractical. On the other hand, while problem-space attacks are more practical than feature-space attacks, they have several drawbacks compared to our proposed method. Firstly, the evasion rate of Hashemi et al.'s method [14], which deals with the same type of attack as our proposed method (problem-space white-box attack), is only 68% at most. Second, since other existing problem-space attacks do not fully utilize the information in the feature space, they have to add a relatively large amount of random perturbations to the attack communication. In contrast, our method uses XAI to select important features and finds modifications in the problem space that perturb them in the feature space. As a result, the modifications in the problem space are so small that they may not compromise the feasibility of AEs or the maliciousness of original attacker traffic.

4 PROPOSED METHOD

We propose a new XAI-based method for generating AEs for DL-based NIDS. In this section, we explain our proposed method. At first, we explain our target machine learning model and define the threat model. We then show how we solve the challenges common to problem-space attacks. We finally describe the overview of our proposed method.

4.1 Targeted NIDS and Threat Model

In this paper, we focus on generating AEs against DL-based NIDS. General DL-based NIDS' detection flow is described in Figure 1. First, using a packet-capturing tool, traffic in the target network is captured. Next, features are extracted from captured raw traffic. If necessary, the extracted features are pre-processed for shaping. Finally, the extracted and shaped data are input to the NIDS model, and the model returns a binary value (0: benign, 1: malicious). We assume that an attacker conducts a *white-box attack* where he or she knows all information about the target NIDS model.

4.2 Challenges and Solutions

Our proposed method solves four main challenges that are common to problem-space attacks [29]. In the following, we enumerate each challenge and describe how we solve it.

(1) **Available transformations:**
Any transformation for generating an AE must be able to be

performed in the problem space. In other words, we must show that our proposed AE for NIDS is realized in the real network. Since it is difficult to clarify this theoretically in our research, we clarify the feasibility of our proposed method by realizing the proposed AE in a real-world environment.

(2) **Preserved semantics:**
In the process of adding perturbations, semantics must be preserved. Semantics in network traffic refer to each feature's link to a host and network attribute, as well as correlations and dependencies between them [25]. If we perturb a feature that has a strong semantic value, we also have to perturb other features that are correlated to it in order to preserve semantics. To avoid this complicated process, we utilize a correlation heatmap [24] to select more independent features and perturb them.

(3) **Plausibility:**
AEs are required to retain their qualitative properties after being perturbed. In our case, this means that the perturbed malicious communication keeps its original maliciousness. In the network domain, the verification of plausibility is difficult to show theoretically or quantitatively. It needs to be analyzed by humans [29]. Therefore, we run our proposed perturbed attacks against a real system and verify that they are successful.

(4) **Robustness to pre-processing:**
In ML-based detection, input data are pre-processed with non-ML techniques. They may disrupt the adversarial attack. In our proposed method, we first select important and independent traffic features to be perturbed, and the perturbation we find in the feature space almost corresponds one-to-one to the modification performed in the problem space. The simplicity of the correspondence between the feature space and the problem space allows us to generate better perturbations that also consider the data pre-processing. As a result, our proposed AEs are robust (unaffected) to data pre-processing.

4.3 Details of Our Proposed Method

Based on the solutions introduced in Section 4.2, we propose a novel method for creating AEs for NIDS. To achieve a high evasion rate of generated AEs, it is important to fully utilize information in the feature space. Therefore, our proposed method identifies effective perturbations in the feature space and then seeks corresponding transformations in the problem space. However, there is an *inverse feature mapping problem* in the network domain: feature extraction functions are irreversible and non-differentiable [29]. Due to this problem, the larger and the more complex the perturbations in the feature space, the more difficult it becomes to find the corresponding problem-space transformations. To address it, we minimize the number of features perturbed in the feature space, aiming to simplify the feature-space perturbations. This approach also makes our AEs more robust to pre-processing. For implementing effective AEs with a minimal number of perturbed features, we use XAI to identify key features that significantly contribute to evade detection. Additionally, to maintain semantics, we focus on perturbing the more independent features among the selected ones.

Our proposed method consists of the following five major steps.

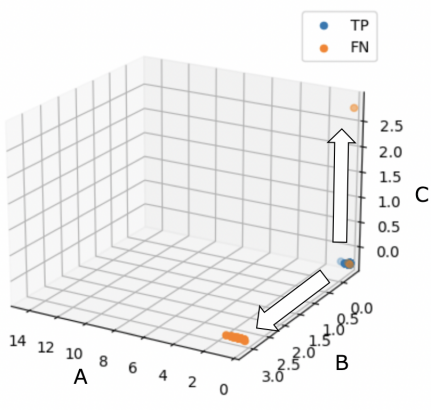

Figure 2: Sample 3D scatter plot

(1) We test the model and analyze False Negative (FN) samples using Integrated Gradients [36] as an XAI model. Then, we select the top k most important features contributing to the targeted model's decision on FN samples. In this research, we deal with the case where $k = 3$.
(2) We plot True Positive (TP) samples and FN samples in the $k(= 3)$-dimensional graph, whose axes are the top k features.
(3) We calculate a correlation heatmap [24] and confirm how independent each important feature is.
(4) From the 3D graphs and heatmap, we select the most suitable feature to be perturbed
(5) We implement the perturbations (AEs) using the real environment and confirm whether they keep their original maliciousness

In Steps (1) and (2), we focus on FN and TP samples. That is because our goal of generating AEs is similar to transforming TP into FN. In Step (2), we plot, for instance, a graph like Figure 2. This figure shows that TP samples are concentrated at the lower end of each axis. Meanwhile, some FN samples are situated at a higher value on both the feature `B` or `C` axes. Through the analyses, we can hypothesize that when generating AEs, we should increase the value of `B` or `C` of the malicious communication (in the direction of the white arrows in Figure 2). For instance, if `B` is '`URG flag Count`,' an attacker might send more packets with the URG flag or set the URG flag on attack packets to increase the feature value. Furthermore, if feature `B` is more independent than feature `C`, we select `B` as the most suitable feature to be perturbed in Step (4).

5 EXPERIMENTAL RESULTS AND DISCUSSION

We implemented our proposed method described in Section 4 and perturbed two types of web attacks, Brute Force attacks and Cross-Site Scripting (XSS), in an actual network environment. We then assessed the extent to which these examples could evade detection of the targeted NIDS model. In this section, we first explain our experimental environment and implementation details of the targeted NIDS model. Then, we show the experimental results of the two attack cases and finally discuss the results.

5.1 Environment Settings

We are required to prepare a real network environment to measure the performance (feasibility and detection evasion rate) of our proposed AE generation, as described in Section 4.3. In the environment, an attacker host (Kali Linux) and a victim server (CentOS) are set up on the same network so that they can communicate with each other. Both machines are virtual machines built on virtualization software, VMware Fusion. All network traffic actually occurred and was captured using Wireshark. CICFlowMeter performs feature extraction. The reason why we chose CICFlowMeter is that it allows us to maintain feature consistency with the CIC-IDS2017 [34] dataset, which is used to build the base model of our targeted NIDS.

5.2 Targeted NIDS Model

Our targeted NIDS model consists of an input layer, two hidden layers (with 256 neurons), and an output layer. During the learning process, we compute the cross-entropy between the labels and predictions as a loss function, and Adam (Adaptive Moment Estimation) with a learning rate of 0.01 is utilized as an optimizer. This architecture is typical for a feedforward neural network and was also adopted in previous works [25]. To construct an NIDS model with sufficient accuracy, we need a sufficiently large and varied set of training data. However, it was difficult for us to generate such training data by using our own environment. Therefore, we first trained the targeted model using the CIC-IDS2017 dataset, which contains a high volume of traffic and a large number of features to be observed for anomaly detection. Subsequently, we fine-tuned it with benign and malicious data generated from our environment to build the final NIDS model.

We apply some pre-processing to the CIC-IDS2017 dataset and the data collected from our network environment:

- Feature removal: We extracted features from the data collected from the real network environment by using CICFlowMeter. The features extracted by CICFlowMeter contains '`Flow ID`', '`Src IP`', '`Src Port`', '`Dst IP`', '`Dst Port`', and '`Timestamp`'. They are not included in the CIC-IDS2017 dataset. We removed them to maintain the consistency of features we deal within our evaluation.
- Min-Max normalization: We normalized the data to ensure that features with larger values do not bias the classification process. This normalization scales the feature values to a [0, 1] range.
- Binary labels: In one attack type, we merge multiple attack categories into a single binary feature. For instance, when dealing with brute force attacks (Section 5.3), we categorize both FTP and SSH brute force attack labels in CIC-IDS2017 under one label, *malicious*.

5.3 Experiment 1: Brute Force Attack

5.3.1 Building the NIDS model. We trained NIDS model using the dataset from CIC-IDS2017, which was collected on Tuesday, July 4, 2017. The model's performance is summarized in Table 1. Subsequently, fine-tuning was performed using benign and malicious data generated in our actual network environment, which was generated as follows:

Table 1: Targeted NIDS model performances

Attack	Model	Precision	Recall
Brute Force	Before Fine-tuning	99.94%	98.67%
	After Fine-tuning	94.12%	94.12%
XSS	Before Fine-tuning	93.18%	94.65%
	After Fine-tuning	93.55%	93.86%

- Benign traffic: Legitimate client logins to an FTP server (vsftpd), along with file uploads and downloads.
- Malicious traffic: FTP Brute Force attacks using FTP-Patator, similar to the one used in CIC-IDS2017.

After fine-tuning, we tested the model with test data from the real environment. As shown in Table 1, it classified benign and malicious traffic with high accuracy.

5.3.2 Generating our proposed AEs. We generated AEs of Brute Force attacks by using our proposed method. The following enumerated items correspond to those in Section 4.3.

(1) Using XAI, we analyzed False Negative (FN) samples. We used the `IntegratedGradients` function from `Xplique` [9] to calculate each feature's mean impact for every FN sample. We selected the top 20 features in order of impact and plotted them in Figure 3. From this figure, we focused on the top three most important features: (`Fwd PSH Flags`, `Down/Up Ratio`, `PSH Flag Count`). The explanations of these three features are as follows [34]:
   - `Fwd PSH Flags`: Number of times the `PSH` flag was set in packets travelling in the forward direction (0 for UDP).
   - `Down/Up Ratio`: Download and upload ratio.
   - `PSH Flag Count`: Number of packets with `PSH` flag.
(2) We created a three-dimensional graph (see Figure 4). From this 3D scatter plot, it was clear that `Fwd PSH Flags` was more suitable to be perturbed than the other two features. Specifically, reducing its value could likely shift TP to FN.
(3) We checked the independence of each feature by creating a heatmap (Figure 5) of their correlations. The figure showed that `Fwd PSH Flags` had a weak correlation with other features. Also, our qualitative analysis revealed that `Fwd PSH Flags`, being a TCP packets' flag count in the forward direction (from a client to a server), had little relation to other flow data features.
(4) Based on the analyses of Figure 4 and Figure 5, we decided to generate adversarial examples by perturbing the `Fwd PSH Flags` to 0.
(5) We implemented `Python` scripts to perform FTP Brute Force attacks without setting a `PSH` flag. In other words, we set the `PSH` flag to 0 for all packets sent from the attacker. We also confirmed the perturbed attacks worked successfully.

5.3.3 Evaluating our proposed AEs. The perturbed FTP brute force attack succeeded. Thus, our proposed AEs did not affect the original maliciousness of attacker traffic at all. To evaluate the impact of these adversarial examples, we measured the evasion rate (i.e., the fraction of malicious communication misclassified as benign).

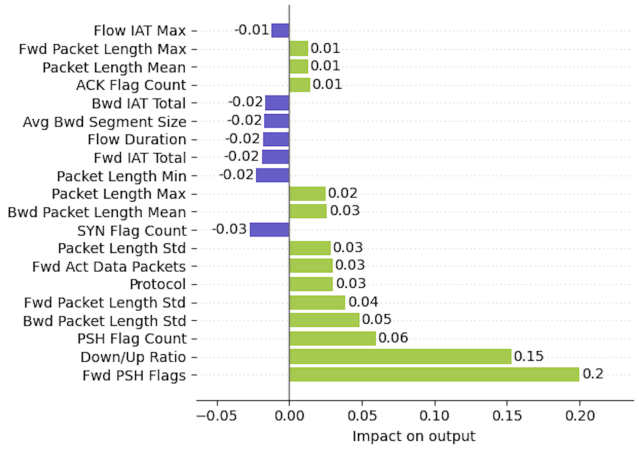

Figure 3: Feature importance for NIDS FN samples in Brute Force attacks

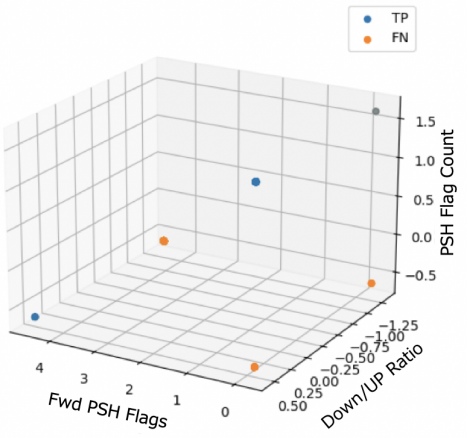

Figure 4: 3D scatter plot of TP and FN samples in Brute Force attacks

The evasion rate was 95.65%, which indicates that our proposed adversarial examples evade detection with a fairly high probability.

5.4 Experiment 2: XSS

5.4.1 Building the NIDS model. Initially, we employed the CIC-IDS2017 dataset containing traffic related to XSS (specifically those conducted on Thursday, July 6, 2017) to train our NIDS model. We excluded data about SQL injection and Brute Force attacks from this dataset. The performance of the model after the training is also shown in Table 1. Subsequently, we constructed a real-world environment for data collection to fine-tune our model. We set up a web server using Apache and prepared a simple e-commerce site, deliberately leaving an XSS vulnerability on the login page. For example, if an attacker entered `<script>alert('xss');</script>` in the username field of the login form, a JavaScript alert, as depicted in Figure 6, would appear on the screen. Data collected in such an environment included:

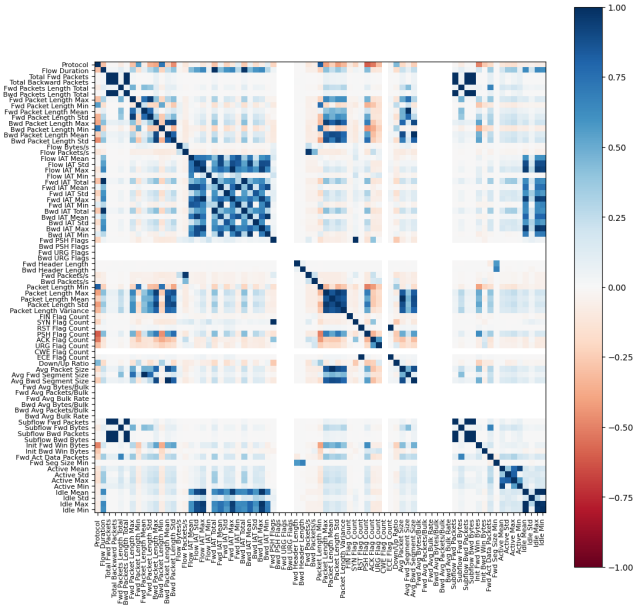

Figure 5: Correlation matrix of features in Brute Force attacks

Figure 6: Login page after XSS

- Benign traffic: Legitimate client logins and subsequent page browsing.
- Malicious traffic: Various inputs of XSS vectors from [32] to the login page.

After the fine-tuning, we evaluated the performance using test data, as illustrated in Table1, confirming the model's high accuracy in classifying communications.

5.4.2 Generating our proposed AEs. We created AEs of XSS. As in Case 1, the following enumerated items correspond to those in Section 4.3.

(1) We analyzed the FN samples using XAI. The results are presented in Figure 7. From this figure, we selected the top three features (`Fwd Seg Size Min`, `URG Flag Count`, and `Bwd Packet Length Min`). The detailed explanations of these three features are as follows:
   - `Fwd Seg Size Min`: Minimum segment size observed in the forward direction.
   - `URG Flag Count`: Number of packets with URG flag.
   - `Bwd Packet Length Min`: Minimum size of packet in backward direction.

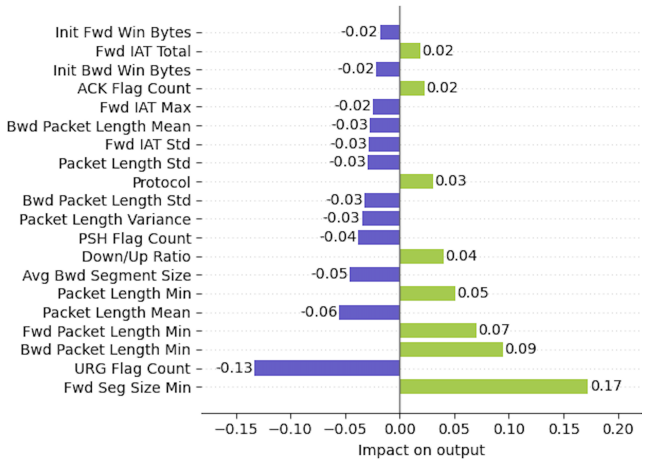

Figure 7: Feature importance for NIDS FN samples in XSS

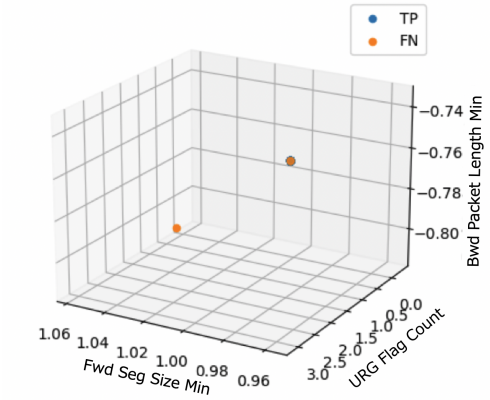

Figure 8: 3D scatter plot of TP and FN samples in XSS

(2) We plotted TP and FN samples in 3D space, as shown in Figure 8. The graph revealed that `Fwd Seg Size Min` was the most critical and easily perturbed feature to generate adversarial examples. Specifically, increasing its value seems to cause the change from TPs to FNs.
(3) We used a heatmap (Figure 9) to verify the independence of each feature's correlation. The figure demonstrated that `Fwd Seg Size Min` had a sufficiently low correlation with other features, indicating it is more independent. Through experimental analyses, under XSS attacks, the attacker's packets whose segment size is minimum were `SYN` or `ACK` packets. We hypothesized perturbing these packets would have minimal impact on other features.
(4) `Fwd Seg Size Min` had the biggest impact on FN samples and was independent enough to be perturbed. Thus, we decided to perturb it.
(5) We implemented the perturbation in the problem space by padding `SYN` and `ACK` packets from the attacker host. Even with such perturbations, all XSS attacks succeeded.

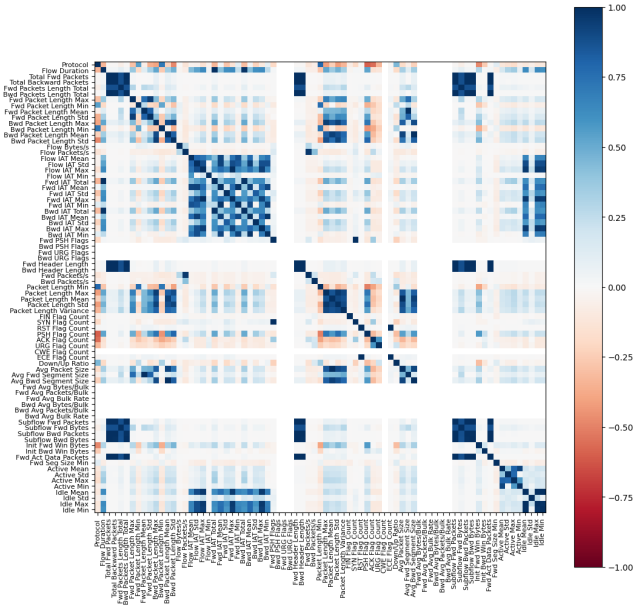

Figure 9: Correlation matrix of features in XSS

5.4.3 Evaluating our proposed AEs. The fact that perturbed XSS attacks succeeded proves that our proposed perturbation did not affect the malicious nature of the attacks. We also evaluated the evasion rate of the adversarial examples. The rate was 100%, which showed that our proposed adversarial examples could completely evade the detection of the NIDS.

5.5 Discussion

Our method attained evasion rates of 95.7% (for Brute Force) and 100.0% (for XSS). Thus, we can consider that our proposed method can generate highly evasive AEs for DL-based NIDS. However, our study has some potential improvements. The first is the inadequate evaluations for the generality of our proposed method. We applied our approach to two types of attacks, but this is somewhat insufficient to assert its general applicability. Additionally, we have used only one type of dataset (CIC-IDS2017) and our targeted NIDS model. Future work will involve validating our method's applicability to various attacks, datasets, and NIDS models. The second is that our method is a white-box attack, where the attacker has full access to the targeted NIDS information. Such a scenario is often pointed out to be impractical in previous studies [5, 12, 13]. We originally assumed that the AEs proposed in this paper could be utilized to evaluate or improve the robustness of DL-based NIDS. In this context, our proposed method does not necessarily have to guarantee complete practicality; that is, it is required to be realized in traffic space but not to be a black-box attack. However, it is also important to make our proposed method more realistic, considering the situation where attackers conduct adversarial attacks against DL-based NIDS. Thus, we plan to explore the transferability of our proposed method. If the adversarial examples we generated demonstrate high transferability, it could extend the utility of our method to more realistic and practical black-box attack scenarios.

6 CONCLUSION

This paper proposes a novel method for generating AEs for DL-based NIDS using XAI. In our proposed approach, we utilized XAI to analyze the decision-making processes of DL-based NIDS models, identifying critical features that should be perturbed to evade NIDS detection. By focusing on these key features and minimizing the number of perturbed features, we also identify the modifications in the problem space (traffic space) that could realize these perturbations in the feature space. In other words, we succeeded in maintaining the feasibility of our method and the maliciousness of the original attack traffic. Furthermore, we implemented our method in a real-world network environment and executed it against two types of network cyber-attacks. We generated AEs with perturbations to only one feature by transforming malicious traffic in the problem space (traffic space). Furthermore, our proposed AEs completely evade the NIDS detection.

ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Number JP23H03361. This work was partly supported by the National Research Agency (ANR) through the GRIFIN project (ANR-20-CE39-0011).

REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018), 52138–52160.

[2] Giovanni Apruzzese, Michele Colajanni, and Mirco Marchetti. 2019. Evaluating the effectiveness of Adversarial Attacks against Botnet Detectors. In *18th IEEE International Symposium on Network Computing and Applications, NCA 2019, Cambridge, MA, USA, September 26-28, 2019*, Aris Gkoulalas-Divanis, Mirco Marchetti, and Dimiter R. Avresky (Eds.). IEEE, 1–8.

[3] Dhanashri Ashok Bhosale and Vanita Manikrao Mane. 2015. Comparative study and analysis of network intrusion detection tools. In *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*. 312–315.

[4] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognit.* 84 (2018), 317–331.

[5] Lei Bu, Zhe Zhao, Yuchao Duan, and Fu Song. 2022. Taking Care of the Discretization Problem: A Comprehensive Study of the Discretization Problem and a Black-Box Adversarial Attack in Discrete Integer Domain. *IEEE Trans. Dependable Secur. Comput.* 19, 5 (2022), 3200–3217.

[6] Anna L. Buczak and Erhan Guven. 2016. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Commun. Surv. Tutorials* 18, 2 (2016), 1153–1176.

[7] Joseph Clements, Yuzhe Yang, Ankur A Sharma, Hongxin Hu, and Yingjie Lao. 2021. Rallying Adversarial Techniques against Deep Learning for Network Security. In *IEEE Symposium Series on Computational Intelligence, SSCI 2021, Orlando, FL, USA, December 5-7, 2021*. IEEE, 1–8.

[8] Filip Karlo Dosilovic, Mario Brcic, and Nikica Hlupic. 2018. Explainable artificial intelligence: A survey. In *41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018, Opatija, Croatia, May 21-25, 2018*, Karolj Skala, Marko Koricic, Tihana Galinac Grbac, Marina Cicin-Sain, Vlado Sruk, Slobodan Ribaric, Stjepan Gros, Boris Vrdoljak, Mladen Mauher, Edvard Tijan, Predrag Pale, and Matej Janjic (Eds.). IEEE, 210–215.

[9] Thomas Fel, Lucas Hervier, David Vigouroux, Antonin Poche, Justin Plakoo, Remi Cadene, Mathieu Chalvidal, Julien Colin, Thibaut Boissin, Louis Bethune, Agustin Picard, Claire Nicodeme, Laurent Gardes, Gregory Flandin, and Thomas Serre. 2022. Xplique: A Deep Learning Explainability Toolbox. *Workshop on Explainable Artificial Intelligence for Computer Vision (CVPR)* (2022).

[10] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[11] M. Gopinath and Sibi Chakkaravarthy Sethuraman. 2023. A comprehensive survey on deep learning based malware detection techniques. *Comput. Sci. Rev.* 47 (2023), 100529.

[12] Chuan Guo, Jacob R. Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Q. Weinberger. 2019. Simple Black-box Adversarial Attacks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 2484–2493.

[13] Dongqi Han, Zhiliang Wang, Ying Zhong, Wenqi Chen, Jiahai Yang, Shuqiang Lu, Xingang Shi, and Xia Yin. 2021. Evaluating and Improving Adversarial Robustness of Machine Learning-Based Network Intrusion Detectors. *IEEE J. Sel. Areas Commun.* 39, 8 (2021), 2632–2647.

[14] Mohammad J. Hashemi, Greg Cusack, and Eric Keller. 2019. Towards Evaluation of NIDSs in Adversarial Setting. In *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks, Big-DAMA@CoNEXT 2019, Orlando, FL, USA, December 9, 2019*. ACM, 14–21.

[15] Mohammad Mehedi Hassan, Md. Rafiul Hassan, Md. Shamsul Huda, and Victor Hugo C. de Albuquerque. 2021. A Robust Deep-Learning-Enabled Trust-Boundary Protection for Adversarial Industrial IoT Environment. *IEEE Internet Things J.* 8, 12 (2021), 9611–9621.

[16] Ivan Homoliak, Martin Teknos, Martín Ochoa, Dominik Breitenbacher, Saeid Hosseini, and Petr Hanácek. 2019. Improving Network Intrusion Detection Classifiers by Non-payload-Based Exploit-Independent Obfuscations: An Adversarial Approach. *EAI Endorsed Trans. Security Safety* 5, 17 (2019), e4.

[17] Ahmad Y. Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. 2016. A Deep Learning Approach for Network Intrusion Detection System. *EAI Endorsed Trans. Security Safety* 3, 9 (2016), e2.

[18] Gour C. Karmakar, Abdullahi Chowdhury, Rajkumar Das, Joarder Kamruzzaman, and Syed Mofizul Islam. 2021. Assessing Trust Level of a Driverless Car Using Deep Learning. *IEEE Trans. Intell. Transp. Syst.* 22, 7 (2021), 4457–4466.

[19] Ryo Kumagai, Shu Takemoto, Yusuke Nozaki, and Masaya Yoshikawa. 2023. Explainable AI based Adversarial Examples and its Evaluation. In *Proceedings of the 6th International Conference on Electronics, Communications and Control Engineering, ICECC 2023, Fukuoka, Japan, March 24-26, 2023*. ACM, 220–225.

[20] Aditya Kuppa and Nhien-An Le-Khac. 2021. Adversarial XAI Methods in Cybersecurity. *IEEE Trans. Inf. Forensics Secur.* 16 (2021), 4924–4938.

[21] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C. Suh, Ikkyun Kim, and Kuinam J. Kim. 2019. A survey of deep learning-based network anomaly detection. *Clust. Comput.* 22, Suppl 1 (2019), 949–961.

[22] Chen Li, Weisi Guo, Schyler Chengyao Sun, Saba Al-Rubaye, and Antonios Tsourdos. 2020. Trustworthy Deep Learning in 6G-Enabled Mass Autonomy: From Concept to Quality-of-Trust Key Performance Indicators. *IEEE Veh. Technol. Mag.* 15, 4 (2020), 112–121.

[23] Zilong Lin, Yong Shi, and Zhi Xue. 2022. IDSGAN: Generative Adversarial Networks for Attack Generation Against Intrusion Detection. In *Advances in Knowledge Discovery and Data Mining - 26th Pacific-Asia Conference, PAKDD 2022, Chengdu, China, May 16-19, 2022, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 13282)*, João Gama, Tianrui Li, Yang Yu, Enhong Chen, Yu Zheng, and Fei Teng (Eds.). Springer, 79–91.

[24] Mohamed Amine Merzouk, Frédéric Cuppens, Nora Boulahia-Cuppens, and Reda Yaich. 2020. A Deeper Analysis of Adversarial Examples in Intrusion Detection. In *Risks and Security of Internet and Systems - 15th International Conference, CRiSIS 2020, Paris, France, November 4-6, 2020, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 12528)*, Joaquín García-Alfaro, Jean Leneutre, Nora Cuppens, and Reda Yaich (Eds.). Springer, 67–84.

[25] Mohamed Amine Merzouk, Frédéric Cuppens, Nora Boulahia-Cuppens, and Reda Yaich. 2022. Investigating the practicality of adversarial evasion attacks on network intrusion detection. *Ann. des Télécommunications* 77, 11-12 (2022), 763–775.

[26] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2022. Image Segmentation Using Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 7 (2022), 3523–3542.

[27] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society.

[28] Xiao Peng, Weiqing Huang, and Zhixin Shi. 2019. Adversarial Attack Against DoS Intrusion Detection: An Improved Boundary-Based Method. In *31st IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2019, Portland, OR, USA, November 4-6, 2019*. IEEE, 1288–1295.

[29] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. 2020. Intriguing Properties of Adversarial ML Attacks in the Problem Space. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 1332–1349.

[30] Aritran Piplai, Sai Sree Laya Chukkapalli, and Anupam Joshi. 2020. NAttack! Adversarial Attacks to bypass a GAN based classifier trained to detect Network intrusion. In *6th IEEE International Conference on Big Data Security on Cloud, IEEE International Conference on High Performance and Smart Computing, and IEEE International Conference on Intelligent Data and Security, BigDataSecurity/HPSC/IDS 2020, Baltimore, MD, USA, May 25-27, 2020*. IEEE, 49–54.

[31] Thomas E. Potok, Catherine D. Schuman, Steven R. Young, Robert M. Patton, Federico M. Spedalieri, Jeremy Liu, Ke-Thia Yao, Garrett S. Rose, and Gangotree Chakma. 2018. A Study of Complex Deep Learning Networks on High-Performance, Neuromorphic, and Quantum Computers. *ACM J. Emerg. Technol. Comput. Syst.* 14, 2 (2018), 19:1–19:21.

[32] PortSwigger Research. [n. d.]. *Cross-site scripting (XSS) cheat sheet.* Retrieved January 2, 2024 from https://portswigger.net/web-security/cross-site-scripting/cheat-sheet

[33] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* 1, 5 (2019), 206–215.

[34] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Madeira - Portugal, January 22-24, 2018*, Paolo Mori, Steven Furnell, and Olivier Camp (Eds.). SciTePress, 108–116.

[35] Elizabeth Stinson and John C. Mitchell. 2008. Towards Systematic Evaluation of the Evadability of Bot/Botnet Detection Methods. In *2nd USENIX Workshop on Offensive Technologies, WOOT'08, San Jose, CA, USA, July 28, 2008, Proceedings*, Dan Boneh, Tal Garfinkel, and Dug Song (Eds.). USENIX Association.

[36] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 3319–3328.

[37] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[38] Tuan A. Tang, Lotfi Mhamdi, Desmond C. McLernon, Syed Ali Raza Zaidi, and Mounir Ghogho. 2016. Deep learning approach for Network Intrusion Detection in Software Defined Networking. In *2016 International Conference on Wireless Networks and Mobile Communications, WINCOM 2016, Fez, Morocco, October 26-29, 2016*. IEEE, 258–263.

[39] Zheng Wang. 2018. Deep Learning-Based Intrusion Detection With Adversaries. *IEEE Access* 6 (2018), 38367–38384.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009