Article

Defending The Defender: Detecting Adversarial Examples For Network Intrusion Detection Systems

Dalila Khettaf [1,*], **Lydia Bouzar-Benlabiod** [1]

[1] LCSI Laboratory, Ecole Nationale Superieure d'Informatique (ESI), Algiers, Algeria
* Correspondence: hd_khettaf@esi.dz

Abstract: The advancement in network security threats led to the development of new Intrusion Detection Systems(IDS) that rely on deep learning algorithms known as deep IDS. Along with other systems based on deep learning, deep IDS suffer from adversarial examples: malicious inputs aiming to change the prediction of a machine learning/deep learning model. Protecting deep learning against adversarial examples remains an open challenge. In this paper, we propose "NIDS-Defend" a framework to enhance the robustness of Network IDS against adversarial attacks. Our framework is composed of two layers: a statistical test and a classifier that together detect adversarial examples in real-time. The detection process consists of two steps: (1) flagging flows that contain adversarial examples with a statistical test, and (2) extracting individual adversarial examples in the previously flagged flows with a classifier. Our approach is evaluated on binary IDS with the NSL-KDD dataset. To generate adversarial examples, the crafting methods used are (1) Boundary attack and (2) HopSkipJumpAttack. We investigate the vulnerabilities of a Network IDS against adversarial examples, then apply our defense. The statistical test can confidently distinguish adversarial flows with more than 95% accuracy, and the classifier detects individual adversarial examples with more than 80% accuracy. We also show that our framework detects adversarial examples crafted by an adversary aware of the defense and confirm the effectiveness of our solution against adversarial attacks.

Keywords: intrusion detection systems; adversarial examples; adversarial attacks; adversarial machine learning; statistics

1. Introduction

Cybercrime is estimated to cost companies worldwide about $10.5 trillion in 2025, up from $3 trillion in 2015[1]. To illustrate this, if cybercrime were a country, it would be ranked as the world's third-largest economy right behind the US and China[2].

To prevent cyberattacks, deep learning(DL) has joined cybersecurity's powerful tools and has been applied to tackle security-related issues including spam detection, malware detection, intrusion detection, and detection of software vulnerabilities [1]. One particular success of deep learning has been in intrusion detection systems(IDS), one of the main tools to protect computer systems or networks against cyberattacks. Two main methodologies are used to detect intrusions: signature detection and anomaly detection, deep learning-based intrusion detection systems are based on the latter. Anomaly detection consists of finding suspicious activities among a network or a host, and the deep version relies on supervised or non-supervised deep-learning algorithms to do so. Different deep learning models have been used to implement intrusion detection systems and achieved satisfactory performance in detecting both known and novel attacks. The authors in [2], [3] developed a Network IDS(NIDS) based on an XGBoost model that achieved 99% in classification accuracy on the NSL-KDD dataset.

[1] https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/
[2] https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

The development of deep IDS is accompanied by parallel progress in another field known as "adversarial machine learning", a domain of intersection between machine learning and cybersecurity. Adversarial machine learning studies the vulnerabilities of ML and DL models against the so-called "adversarial examples" and techniques to defend against them. Adversarial examples are inputs slightly manipulated aiming to fool an ML/DL model [4]. The existence of adversarial examples raises security issues in deep IDS and questions whether adversaries might not disguise potential attacks in clean data using adversarial examples to cause misclassification for an IDS.

Since adversarial examples were discovered for the first time by the authors in [4] and proved to be a security threat against systems based on ML and DL [5], more researchers are proposing methods to detect attacks with adversarial examples. However, most of those works tackle adversarial examples' detection for the image classification problem and very few of them focus on NIDS.

As modern networks grow and become more complex, deploying a NIDS is crucial to keep them free of cyber-attacks. Machine learning has shown great potential in tasks like intrusion detection, however, NIDS based on deep learning despite their outstanding performance in normal settings are vulnerable to adversarial attacks that represent a new security challenge for them and drastically reduce their accuracy [6].

In this research work, we extend results from the statistical properties of adversarial examples introduced by [7] to protect a Network IDS against adversarial attacks after a literature review to explore the potential methods to attack and defend intrusion detection systems. Our main contribution is a two-layer framework that detects decision-based adversarial attacks targeting a NIDS in real-time. Other contributions include implementing a deep IDS trained with the NSL-KDD dataset and then highlighting the vulnerabilities of a NIDS in adversarial settings before demonstrating the efficiency of our approach.

The rest of this paper is organized as follows: **section 2** is a literature review of the existing methods of adversarial attacks and defenses within intrusion detection. **section 3** explores the two methods: the statistical test and the outlier class introduced in [7]. **section 4** introduces our main contribution (the framework) along with other secondary contributions. **section 5** introduces the experiments and results on the NIDS and the framework. **section 6** discusses the future improvement of our approach and compares it to prior work, then **section 7** concludes the paper.

2. Related Work

An intrusion detection system(IDS) is one of the main tools to protect computer systems and networks against cyberattacks. Recent IDS rely on deep learning algorithms to detect potential cyberattacks attacks. Along with these threats, deep learning-based intrusion detection systems suffer from new vulnerabilities related to machine learning's security. Since authors in [4] published their work on adversarial examples, only a few research works studied the vulnerabilities of deep network intrusion detection against adversarial attacks that could be crafted in two settings: white-box where the adversary has full knowledge of the IDS or black-box where the adversary can only query the IDS.

Researchers in [8] investigated the accuracy of a deep learning-based network intrusion detection system(NIDS) against the Fast Gradient Sign Method(FGSM) [9] adversarial attack. The victim model is a neural network including 3 hidden layers of 110 neurons total and trained with NSL-KDD dataset [10]. The binary version of this dataset has two classes: normal and attack. In normal settings, the model has achieved 99% in accuracy for both classes. After generating adversarial examples with the method FGSM, all adversarial instances were misclassified as normal data. Furthermore, in [11], the authors carried out experiments using a deep neural network (DNN) model made of three layers of 512 hidden units total and trained with the NSL-KDD dataset. The authors generated adversarial examples with three black-box methods: (1) based on a substitute model, (2) based on the zeroth order optimization (ZOO), and (3) based on generative adversarial nets (GAN). All adversarial attacks were crafted from the test set. In normal settings, the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

model achieved 89% in accuracy, that has drastically dropped to 53,7% for the zeroth order optimization (ZOO) attack, to 73,8% for the substitute model (transferability attack) and to 56,7% for the generative adversarial nets (GAN) attack. Also, authors in [12] crafted adversarial examples based on Fast Gradient Sign Method(FGSM) [9], Jacobian-Based Saliency Map Attack(JSMA) [13], Carlini & Wagner(C&W) [14], DeepFool [15] and Basic Iterative Method/Projected Gradient Descent(BIM/PGD) [5]. The victim model is a Multi-Layer Perceptron (MLP) composed of two hidden layers of 256 neurons each and a Softmax output layer. In normal settings, the model achieved 75.11% in accuracy, but decreased to about 24% under adversarial attacks. The authors went as far as to investigate how each of the attack methods impacts the accuracy of the model and choose the criteria to validate the adversarial attack instances.

On the other hand, other works have proposed techniques to detect adversarial examples in network intrusion detection. In the work detailed in [16], authors built a NIDS that works with both packet-based and flow-based network data. The NIDS is a denoising autoencoder that uses Reconstruction from Partial Observation (RePO), this approach consists of reconstructing parts of the input only; a mask is used to hide some parts of the input that the model has to rebuild during inference time. This method leads to an increase in the error of reconstruction of malicious data and thus their detection when this error exceeds a certain threshold. The NIDS is trained with the CIC (the Canadian Institute of Cybersecurity) dataset [17]. In adversarial settings, the packet-based NIDS detected 62.07% of the network attacks, whereas the flow-based NIDS' detection rate was 47.02%. Furthermore, authors in [18] proposed MANDA, a MANifold and Decision boundary-based AE (adversarial examples) detection system, their solution is based on the observation that crafted adversarial examples need to preserve their malicious properties and reside inside or close to the malicious manifold. The attacks considered here are white-box attacks that aim to modify the traffic flow. MANDA has two components: Manifold and DB, Manifold detects whether an input belongs to the malicious manifold whereas DB detects if an input is near the decision boundary. The experiments were performed using the two datasets NSL-KDD and CICIDS2017 [17], the target IDS was a multilayer perceptron (MLP) and the adversarial attacks that were used for the test are FGSM, BIM, C&W, and JSMA. The detection rate was 92% for the FGSM attack, 98% for the BIM attack, and 95% for the C&W attack. Also, authors in [19] trained a generative adversarial network(GAN) with the CSE-CIC-IDS2018 dataset [20] to enhance the robustness of a NIDS against the C&W attack. The discriminator is added as an auxiliary model to perform binary classification to distinguish between adversarial and non-adversarial data, the NIDS then performs its regular classification task to detect network attacks. The authors implemented the NIDS with different algorithms and studied their vulnerability, then the performance of their defense. The GAN discriminator improved the robustness of the studied NIDS, 75% of the adversarial examples were detected by the discriminator. In addition, authors in [21] implemented a framework that combines three techniques: model voting ensembling, ensembling adversarial training, and query detection to protect a NIDS against black-box adversarial attacks. The framework was tested with the decision-based attacks: NES [22], Boundary[23], Pointwise, HopSkipJumpAttack [24] and Opt-Attack [25].

Although these works provide a relatively good resistance against adversarial attacks for NIDS, only one proposed a method to tackle decision-based black-box attacks when in real-world settings a NIDS is a black box. To our knowledge, we're the first to study the statistical defenses for NIDS against decision-based black-box adversarial attacks with the NSL-KDD dataset.

3. Statistical Properties of Adversarial Examples

The authors in [7] used hypothesis testing to prove that adversarial examples and clean data are drawn from different probability distributions. The two-sample statistical hypothesis test aims to determine whether two samples are drawn from the same probability distribution. Let X_1 and X_2 two samples such that $X_1 \sim p$, $X_2 \sim q$ drawn from

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

different distributions, $\|X_1\| = n$ and $\|X_1\| = m$ respectively. The statistical test $\tau(X_1, X_2)$ distinguishes between two hypotheses: the null hypothesis H_0 and the alternative hypothesis H_A. The null hypothesis H_0 states that the two samples are drawn from the same distribution, while the alternative hypothesis H_A states that the two samples are drawn from two different distributions. The statistical test takes the two samples as an input and outputs a p-value. The p-value is the probability to obtain the observed results under the assumption that the null hypothesis H_0 is true. A low p-value means that the observed results are very unlikely to happen, and the null hypothesis is rejected.

During the training process, a classifier tries to learn the real distribution $D_{real}^{C_i}$ of features with regard to a partition of data C_i corresponding to a class i. Due to the limited number of data points in the training dataset, a classifier can't learn the real distribution of data, so it learns the distribution of the training data $D_{train}^{C_i}$.

Adversarial examples generated in a class i constitute their own distribution $D_{adv}^{C_i}$. Since any data point belonging to this distribution is still in the class i, it's also in the real distribution of the data $D_{real}^{C_i}$. But this is not the case for the two distributions $D_{adv}^{C_i} \neq D_{train}^{C_i}$ because adversarial examples crafted from data points in the training set cannot have the same class i as the data they originated from, otherwise they wouldn't be adversarial. This offers the possibility to statistically distinguish between adversarial examples and clean data.

3.1. The Statistical Test

To compare two data samples, the authors in [7] used a statistical test based on a distance metric that is suitable for high dimensional data and small sample sizes: the biased estimator of the true Maximum Mean Discrepancy (MMD) introduced by [26]:

$$MMD_b[\mathcal{F}, X_1, X_2] = \sup_{f \in \mathcal{F}} \left(\frac{1}{n} \sum_{i=1}^{n} f(x_{1i}) - \frac{1}{m} \sum_{i=1}^{m} f(x_{2i}) \right) \tag{1}$$

x_{1i}, x_{2j} are the ith and jth data points in the samples X_1, X_2 respectively. $f \in \mathcal{F}$ is the kernel function (Gaussian kernel was used by the authors to do the experiments) that is chosen to maximize the distance between the two samples.

To evaluate the hypothesis that adversarial examples are drawn from a different distribution than clean data, a statistical test is performed. The biased estimate of MMD is computed using the formula in **Equation 1** then 10,000 bootstrapping iterations are performed to estimate the distributions. Finally, the p-value is computed and compared with a threshold (typically 5%). For legitimate data, the p-value is expected to be high, for adversarial examples, however, it's expected to be very low (below the threshold) and thus the hypothesis H_0 is rejected.

3.2. Detecting Individual Adversarial Examples

The statistical test aims to prove the assumptions on the distributions of adversarial examples and legitimate data. The test cannot be used to detect individual adversarial examples, which is mandatory for applications like intrusion detection. To solve this problem, a second model is trained with the same dataset as the NIDS, the dataset is augmented with a new category called the outlier class. The model assigns the new label to all the data points that are not part of the distribution of legitimate data.
A model N is trained on the dataset $D = \{X, Y\}$ augmented with adversarial examples crafted from the original dataset. Adversarial examples crafted using different algorithms belong all to the outlier category. The model is trained with batches of $1/3$ adversarial examples and $2/3$ legitimate data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

4. Contribution

In this paper, we propose "NIDS-DEFEND" a two-layer defense framework added on top of the IDS to detect adversarial examples. The framework works on any intrusion detection system as long as it's trained on the same dataset.

Before diving into the details of our main contribution, we start by defining the threat model, then the victim NIDS, and finally we detail the two layers of the framework.

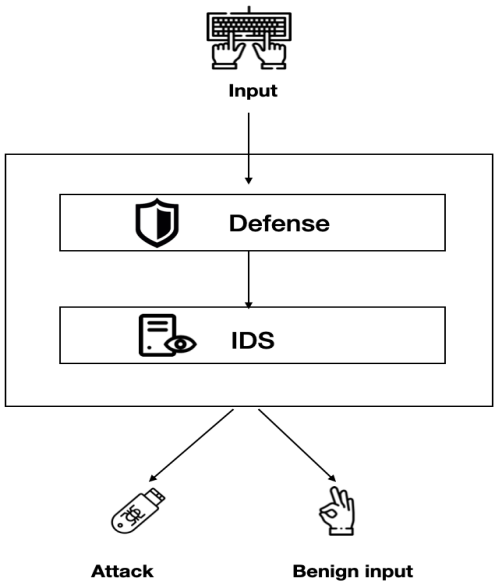

Figure 1. Overview of the framework.

Figure 1 shows an overview of the solution we propose. The detection of adversarial examples happens during inference time, this type of setup is essential for intrusion detection systems. Every input that the IDS receives will go into the defense framework first, if the input is considered adversarial it's rejected immediately. Only clean data is allowed to get into the IDS. The IDS then will detect network attacks(non-adversarial ones) as usual.

4.1. Threat Model

A deep NIDS is a black box in the sense that an adversary cannot get access to the model's architecture or parameters and will just query the NIDS. The adversarial attacks considered here are decision-based and more precisely the two black box attacks: boundary attack and HopSkipJump attack.

4.2. The Victim IDS

The IDS targeted by the adversaries is a deep NIDS. Our NIDS is based on the sequence-to-sequence architecture. The model is based on two main components: an encoder and a decoder.

The encoder: receives the training data as input and transforms it into a context vector that encapsulates all the knowledge from the training dataset into a single vector.

The decoder: receives a signal to begin the training process. At each step, the decoder receives the context vector as input and uses the prediction from the previous iteration to make the new prediction.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

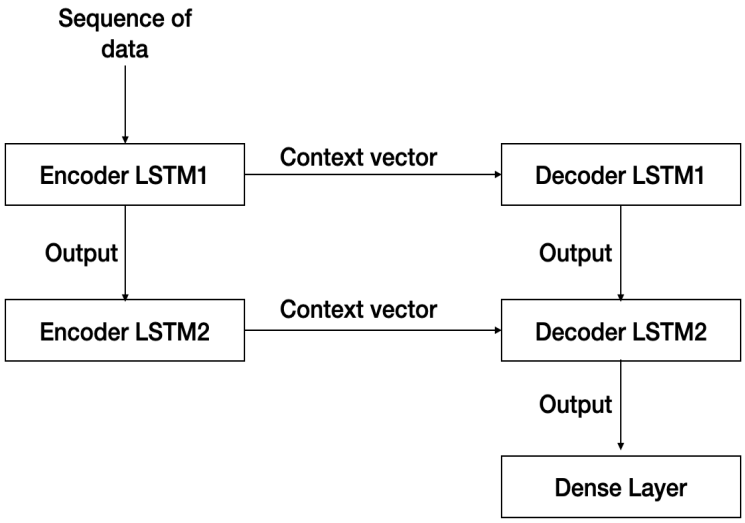

Figure 2. Architecture of the victim NIDS.

Figure 2 shows the architecture of the NIDS. We've used two layers composed of an encoder and a decoder each. The encoder and the decoder are LSTM cells. The two encoders and decoders in the sequence-to-sequence model are connected via their outputs. The dense layer is used for the final prediction.

4.3. NIDS-DEFEND: A Framework To Defend NIDS Against Adversarial Examples

The framework is composed of two layers:

- The first layer is a statistical test (recall **subsection 3.1**) that detects if a network flow contains adversarial examples.
- The second layer is a classifier that detects individual adversarial examples (recall **subsection 3.2**).

4.3.1. Layer 1: The Statistical Test

The goal of this statistical test is to compare two data samples to check whether they are drawn from the same distribution. The objective is to flag any adversarial network data samples. For our case, the two-sample test is performed on the following populations:

- **Population 1:** In a buffer, the NIDS inputs are saved and then arranged into samples of the same size. These samples might contain adversarial examples.
- **Population 2:** This population consists of legitimate data taken from the dataset. The data samples from this population are chosen after a series of tests.

The size of the samples used in the statistical test will be chosen after the experiments.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

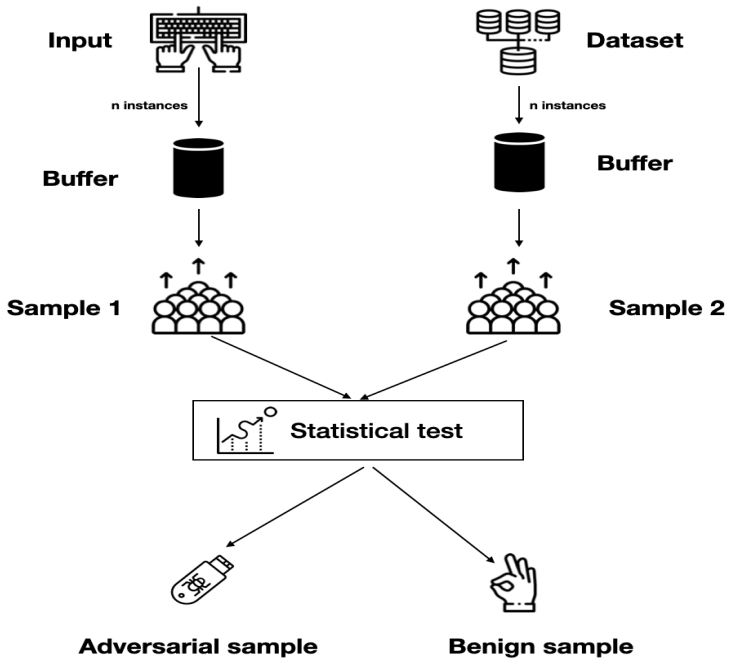

Figure 3. The statistical test for adversarial samples detection.

Figure 3 illustrates the different steps of the statistical test:

1. Choose a sample of legitimate data from the dataset.
2. Extract a sample of data from the NIDS' inputs.
3. Apply the kernel-two-sample statistical test to compare the two samples, the output of this test is the p-value.
4. Compare the p-value with a threshold chosen during the experiments, if the p-value is below the threshold, the null hypothesis is rejected, and thus the input sample contains adversarial examples.

4.3.2. Layer 2: The Outlier Detection

The feasibility of this technique relies on the results of the statistical test. The authors [7] proved that adversarial examples are drawn from a different distribution than legitimate data, which makes this technique possible.

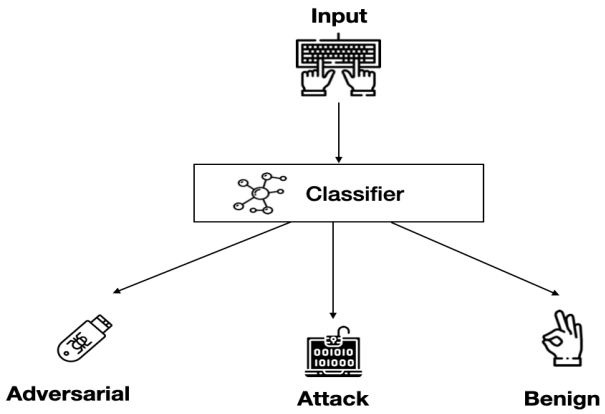

Figure 4. Adversarial examples detection with an outlier class.

Figure 4 shows the three classes used in this technique. To train the classifier, two categories of benign data and network attack data are used, this is typical for a network

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

intrusion dataset. Then the dataset is augmented with a third class called the outlier class with a label "adversarial". This class consists of all adversarial examples crafted from different algorithms.

4.3.3. Interaction Between The Two Layers

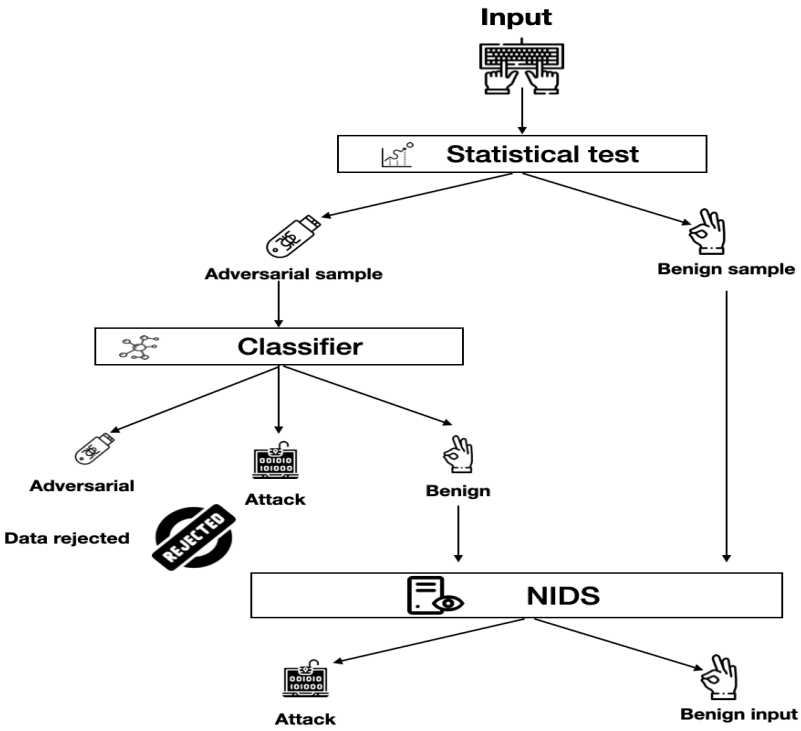

Figure 5. Interactions between the two layers of the framework.

Figure 5 illustrates the adversarial examples' detection process with the framework.

1. After receiving the network data, samples are saved in buffers, then a statistical test is performed. If the data sample is benign, it's sent over to the NIDS, otherwise, the data sample is sent over to the second layer of the framework.
2. In the second layer and for each data point in the sample, the classifier will make a prediction. If a data point is not benign, it's rejected. Otherwise, the data point is sent over to the NIDS.
3. The final prediction is made by the NIDS to whether a network attack is present in the data. That way, our solution minimizes the number of adversarial examples that get the NIDS.

5. Experimental Setup & Results

In this section, we aim to answer the following questions:

1. Is our intrusion detection system vulnerable against adversarial examples?
2. What is the best population and sample size that provide better results for the statistical test?
3. How good is the detection with an outlier class?
4. How good is the framework in protecting the NIDS?
5. Can an adversary aware of the defense attack the NIDS?

We carry out a series of tests to prove the effectiveness of our solution. We start by studying the vulnerabilities of the NIDS in the presence of black-box adversarial attacks. Then, we perform tests to choose the parameters of the statistical test and confirm its ability to flag white-box adversarial examples. After that, we test the resistance of the classifier

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

to white-box adversarial examples. We finally perform two tests in a real-world attack scenario and an adversary aware of the defense with black-box adversarial attacks.

5.1. Dataset

We have chosen to work with is the NSL-KDD dataset [10]. It's an improved version of the KDD cup99 where the redundancies are eliminated. The dataset is composed of 4 files:

- **KDDTrain+:**Training set available in two formats: ARFF with binary labels and with attack type labels for CSV format.
- **KDDTrain+_20Percent:**A subset of the training set, it's also available in two formats: ARFF and CSV.
- **KDDTest+:**The test set that's available in two formats: ARFF with binary labels and with attack type labels for CSV format.
- **KDDTest-21:**A subset of the test set without records of difficulty level 21 out of 21 (the most difficult traffics in the dataset). Available in two formats: ARFF and CSV.

The records in this dataset are made of 41 attributes and a label that specifies whether the record represents an attack or not. 9 of these attributes are basic, 13 are content-related, 9 are time-related and 10 are host-based. The records in the dataset are available in 5 categories: normal and 4 attack classes: DoS, Probe, R2L, and U2R.

5.2. Adversarial Examples Crafting

To evaluate each layer of the proposed framework, we started by crafting white-box adversarial attacks. The victim deep NIDS is a multi-layer perceptron(MLP) composed of 2 hidden layers of 256 neurons each. The RELU activation function was used for the hidden layers and a Softmax for the output layer. The model is implemented using the Pytorch library and trained with 1000 epochs [12].

To craft the adversarial examples, we have used the attack class of the dataset NSL-KDD. The goal is to transform attack records into benign ones. So the target class for all the attacks is the benign class of the NSL-KDD dataset. For this purpose, the following white-box crafting methods have been used:

- **FGSM:**With the maximum perturbation magnitude $\epsilon = 0.1$ and a batch size of 128.
- **BIM/PGD:**The number of iterations for this attack is 100 and the magnitude of the perturbation is 0.001 each iteration and the maximum perturbation magnitude $\epsilon = 0.1$ and a batch size of 128.
- **Deep Fool:**This attack is generated after 100 iterations with a magnitude of 10^{-6}. We used the l_2 version of the attack.
- **C&W:**This attack was crafted with a learning rate of 0.01 and batch size of 128. The l_2 version of this attack was crafted.
- **JSMA:**We used the targeted version of the JSMA attack, the target class is the class normal of NSL-KDD. All the features were perturbed by the algorithm, with a perturbation of 0.1 and a batch size of 128.

We first confirm that the statistical test can distinguish between normal data and white-box adversarial examples. After that, the solution is tested on real scenarios of black-box attacks. For the Boundary attack, we crafted the adversarial examples using the targeted version of the attack with 100 iterations and a learning rate of 0.01. For the HopSkipJump attack, the targeted version was also used. We generated the adversarial examples after 40 iterations and 100 evaluations. The victim model for these two attacks is the NIDS itself (recall **subsection 4.2**).

5.3. Vulnerabilities Of The NIDS

Before tackling our defense strategy, it's important to prove that NIDS needs to be protected.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

Table 1. Performance of the NIDS in normal settings.

Metric	Value
Accuracy	90.68 %
Precision	99.69 %
F1 score	99.74 %
Recall	99.79 %

Table 1 shows that the NIDS detects attacks with an accuracy of 90% and with a precision of 99%. The NIDS shows good state-of-the-art performance in normal settings for test data.

Table 2. Performance of the NIDS in adversarial settings. The success rate is the rate of the adversarial examples classified in their target class (the class attack of NSL-KDD).

Attack	Success Rate(%)
Boundary attack	97.81 %
HopSkipJump attack	100 %

Table 2 shows that the NIDS is vulnerable to adversarial examples despite its performance in normal settings. The success rate is 97.81% for the Boundary attack and 100% for the HopSkipJump attack. This answers our **first question** and proves how essential our defense is.

5.4. Statistical Testing

Before performing the statistical test to detect black-box adversarial attacks, we carry out a series of tests to answer the second and fifth questions.

For this first test, the objective is to choose the best population composition and size that can efficiently flag adversarial examples. We run the kernel-two-sample test using the following populations:

- **Population 1:** Adversarial examples crafted with the algorithms FGSM, JSMA, C&W, BIM and DeepFool.
- **Population 2:** We first run the statistical test with data belonging to the class benign of the NSL-KDD dataset and then with data belonging to the class attack of the same dataset.

We vary the sample size each time to find the optimal value that allows distinguishing between adversarial and normal data.

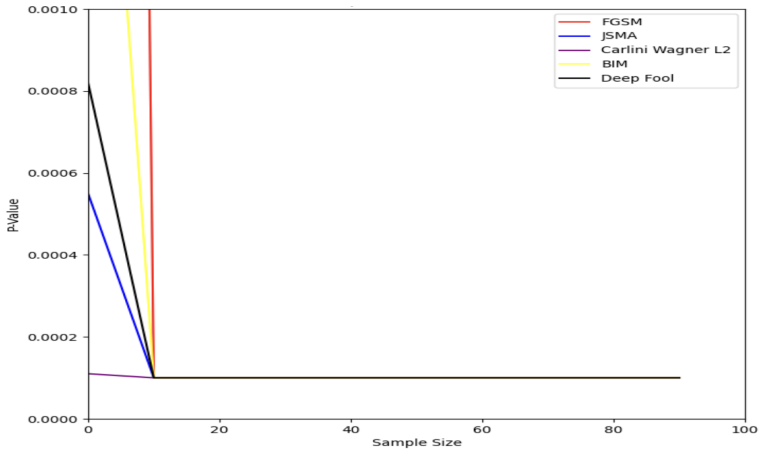

Figure 6. Statistical test: benign data X adversarial data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

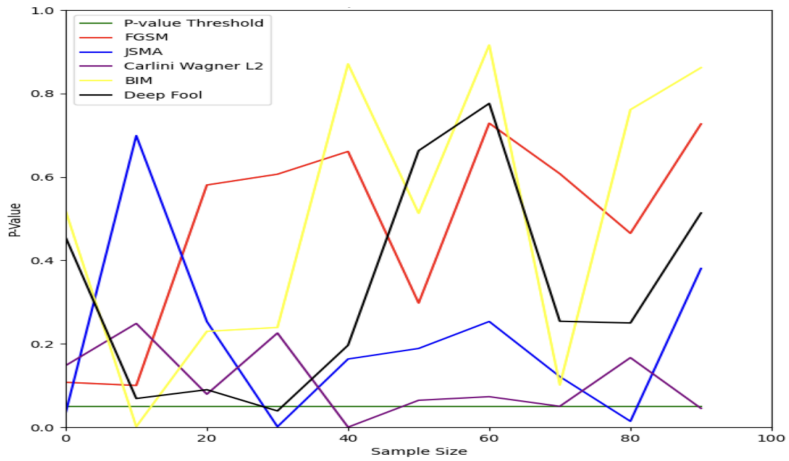

Figure 7. Statistical test: attack data X adversarial data.

The figures show the p-value as a function of the sample size for the statistical test between population 1 and the benign class of the NSL-KDD dataset in **Figure 6** and the attack class in **Figure 7**. The threshold of the p-value used here is 0.005.

Based on the graphs, we can make two conclusions:

- Comparing adversarial data with data in their target class (benign) allows for more accurate results in the statistical test,.i.e a p-value that's below the threshold.
- The sample size necessary to perform the statistical test is 10 inputs per sample.

For the second test, the objective is to verify whether an adversary whose aware of the defense can still attack the NIDS. In such a scenario, the adversary will inject adversarial examples among legitimate data from both the benign and attack classes of NSL-KDD. To run the test, the following populations are used:

- **Population 1:** Benign class data from the NSL-KDD dataset.
- **Population 2:** Mixture of benign class data and variable percentages of attack class (Normal Mix) and variable percentages of adversarial examples.

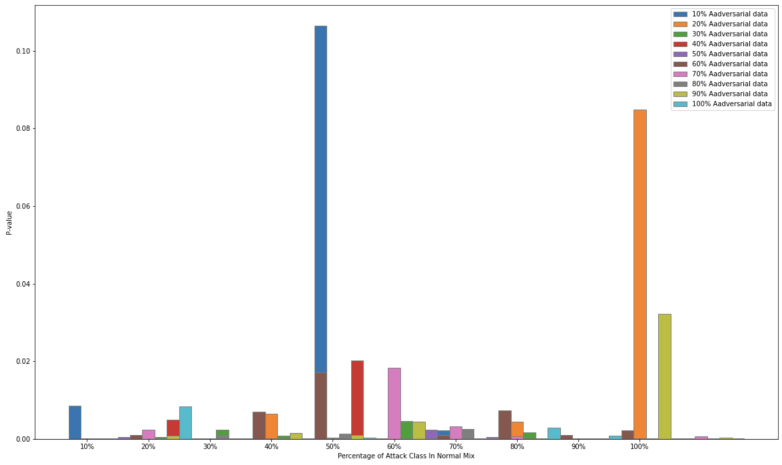

Figure 8. Different percentages of FGSM attack and attack class data X Benign class data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

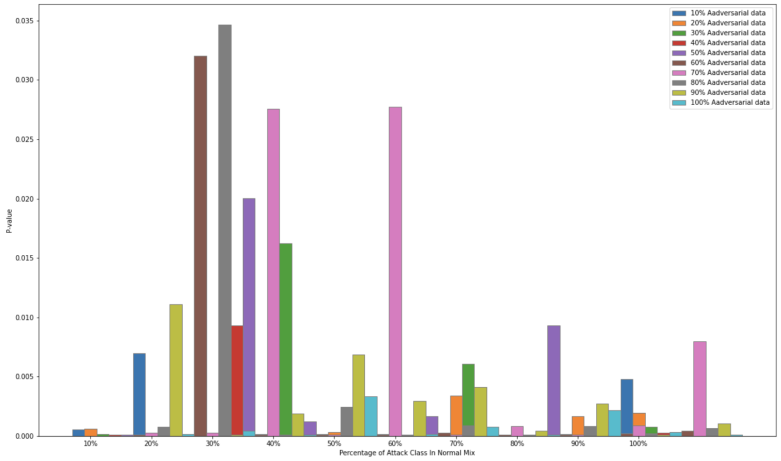

Figure 9. Different percentages of JSMA attack and attack class data X Benign class data.

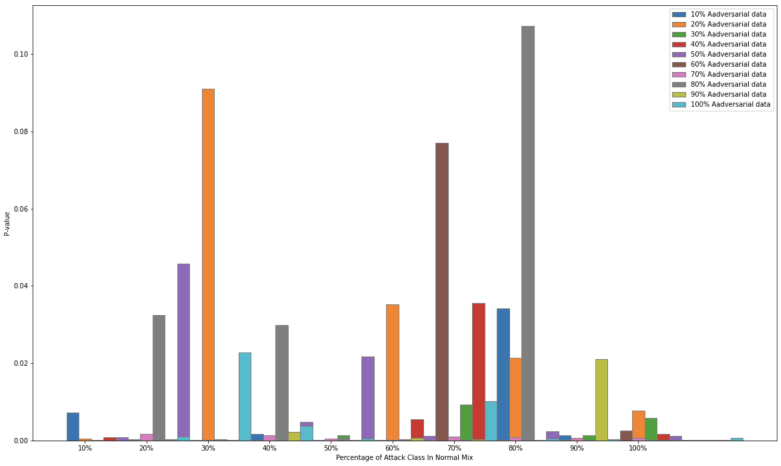

Figure 10. Different percentages of C&W attack and attack class data X Benign class data.

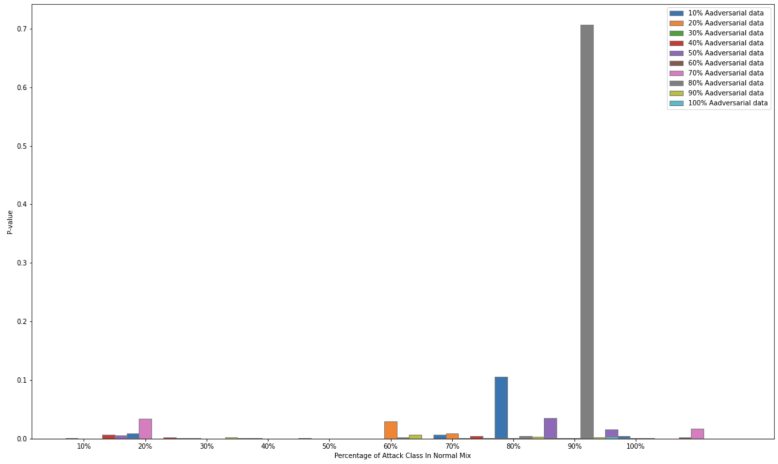

Figure 11. Different percentages of BIM/PGD attack and attack class data X Benign class data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

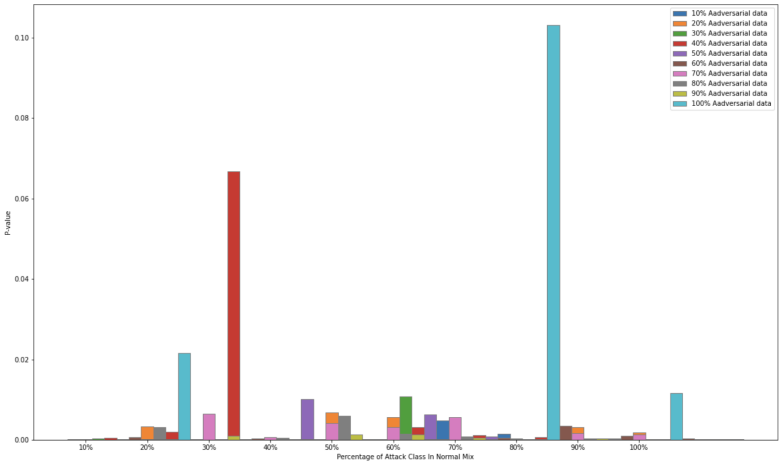

Figure 12. Different percentages of DeepFool attack and attack class data X Benign class data.

The graphs in the above figures represent the p-value as a function of the attack class percentage and the adversarial examples' percentage.

From the graphs, we can make the following observations:

1. For the JSMA attack (**Figure 9**) and for all the percentages of the class attack or adversarial examples, the p-value stays below the threshold.
2. For the C&W attack (**Figure 10**), the p-value exceeds the threshold three times for some percentages of the class attack and adversarial examples.
3. For the two attacks, DeepFool (**Figure 12**) and FGSM (**Figure 8**), the p-value exceeds the threshold about 3 times in the whole graphs.
4. For the attack BIM/PGD (**Figure 11**), when the percentage of the class attack is between 80% and 90% and the adversarial examples' percentage between 10% and 80%, the p-value exceeds the threshold and thus the statistical test fails to detect adversarial samples.

Through these two tests, we answered the **second and fifth question** and confirmed that in most cases, the statistical test is capable of distinguishing between adversarial samples and benign data even when the adversary is aware of the defense.

5.5. The Outlier Detection

To implement this technique, we kept the two classes of the original dataset and added an outlier class with the label "adversarial". The data points in this class are adversarial examples crafted using the training set of NSL-KDD. The algorithms used to generate the adversarial examples are: FGSM, JSMA, Deep Fool, PGD et C&W. We make new training and test sets with equal amounts of the instances for each of the classes: benign, attack and adversarial.

For the classifier, we used Tabnet [27], a deep neural network for tabular data. Tabnet could be used with raw data, without the need to preprocess the dataset. The training of our classifier auto-stopped after 22 epochs with an accuracy of 72% for test data.

Table 3. Performance of Tabnet trained with an outlier class. Detection rate is the rate of adversarial examples classified in the adversarial class. Recovery rate is the rate of adversarial examples classified on their original class (attack). Error rate is the rate of adversarial examples classified as benign.

Attack	Detection Rate(%)	Recovery Rate(%)	Error Rate(%)
JSMA	83.51%	8.46%	8.02%
C&W	52.20%	20.16%	27.63%
Deep Fool	73.43%	9.32%	17.23%

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 December 2022 doi:10.20944/preprints202212.0409.v1

Table 3 shows the results after training, we tested the classifier with the 3 adversarial attacks: JSMA, C&W and DeepFool.

The C&W attack has the highest error rate, followed by DeepFool and finally JSMA attack that has the highest detection rate. Overall, the classifier performs well for all the attacks, which answers our **third question** and confirms that the outlier class technique is efficient in detecting and correcting adversarial examples.

5.6. Real-World Attack Scenario

We run a series of tests using adversarial examples crafted with the algorithms Boundary and HopSkipJump and the NIDS as the victim model.

The first test consists of running a statistical test between the benign data and adversarial examples, the sample size for this test is 10.

Table 4. Results of the statistical test: detection rate is the rate of the samples that contain adversarial examples and detected as adversarial. Error rate is the rate of adversarial samples detected as benign.

Attack	Detection Rate(%)	Error Rate(%)
Boundary attack	100%	0%
HopSkipJump attack	95.94%	4.05%

According to **Table 4**, the NIDS with the statistical test as a defense has a very small error rate of 4.05% for HopSkipJumpAttack and 0% for the Boundary attack.

The second test we run consists of using the classifier trained with an outlier class as a defense.

Table 5. Performance of Tabnet trained with an outlier class for the detection of the two black-box attacks: Boundary and HopSkipJump attacks.

Attack	Detection Rate(%)	Recovery Rate(%)	Error Rate(%)
Boundary attack	72.96%	15.07%	11.96%
HopSkipJump attack	81.52%	12.12%	6.34%

Table 5 shows the test on the black-box attacks that record lower error rates compared to white-box attacks. These results show how efficient the outlier classifier is and thus answers the **third question**. **The third test** consists of a scenario where an adversary adds adversarial examples within normal data (from both classes: attack and benign of the NSL-KDD dataset). To generate this mixture of data, we've randomly taken samples from each class. We followed the steps as in **Figure 5**. We start by sampling the data that the NIDS receives, we run the statistical test for each sample. If the sample is adversarial, it will be sent to the second layer. For each data point in the sample, the classifier will make a prediction. If the data point is adversarial or attack, it will be rejected. Otherwise, it's sent to the NIDS. The NIDS makes the final prediction.

Table 6. Performance of the defense framework in a real-world attack scenario.

Metric	Value
True positive rate (Statistical test)	64.56%
False positive rate (Statistical test)	19.26%
True negative rate (Statistical test)	12.56%
False negative rate (Statistical test)	3.60%
Detection rate (Outlier class)	57.95%
Recovery rate (Outlier class)	14.39%
Error rate (Outlier class)	23.83%

 doi:10.20944/preprints202212.0409.v1

Table 6 shows the results of the both layers as a defense. The first layer detects about 64,56% of the adversarial samples. 57,95% of adversarial examples within these samples have been detected as adversarial by the second layer, 14,39% have been recovered and 23,83% misclassified.

This last test shows that our framework is capable of detecting adversarial attacks even when the adversary is aware of the defense and injects benign data among adversarial examples, this answers both **questions four and five**.

6. Discussion

As to our knowledge, the work by [21] is the only one that proposes a defense against decision-based attacks, but they evaluated their solution with a different dataset CSE-CIC-IDS2018[28]. The lowest success rate of boundary attack was 5.86% obtained with the adversarial query detection method, the lowest success rate of the same attack 0% was obtained by the statistical test. Regarding the HopSkipJump attack, the lowest success rate 5.88% was obtained with the adversarial query detection method, the lowest success rate for our solution was 4.05% again with the statistical test. This shows that our framework performs better for decision-based attacks compared to the work in [21]. Our solution could be further improved by increasing the accuracy of the NIDS and also speeding up the runtime performance of the entire framework. The framework can also be generalized to other decision-based attacks like Pointwise, NES, or OPT.

7. Conclusion

The use of deep learning in cybersecurity has achieved exciting results and might indicate the beginning of a new era of AI-driven cybersecurity. The success of deep learning in domains like intrusion detection opens up the possibility for potential adversaries to carry out attacks with adversarial examples. As prospective bad actors develop their methods to threaten the security of computer networks, it is essential to study defenses against such potential attacks.

In this work, we have studied the vulnerabilities of a deep learning-based network intrusion detection system against the black-box decision-based adversarial attacks Boundary and HopSkipJump. After confirming that the NIDS is not resistant to adversarial examples, we proposed "NIDS-DEFEND" a framework composed of two layers: a statistical test and a multi-classifier trained with the NSL-KDD dataset that we added on top of the NIDS to detect adversarial examples. We started by evaluating the performance of our NIDS in both normal and adversarial settings, after studying the vulnerabilities of the model we evaluated the effectiveness of each layer. As a final step, we evaluated both layers in a scenario where the adversary is aware of the defenses and proved that our defense stays efficient in this case.

To mitigate the impacts of adversarial examples in sensitive applications like cybersecurity and healthcare where deep learning vulnerabilities can put in danger a patient's health, expose personal information or cause catastrophic financial loss to companies, we recommend doing further research to develop defense strategies that give robustness guarantees and don't rely solely on statistics.

Author Contributions: Conceptualization, all authors ; methodology, D. Khettaf; software, D. Khettaf; validation, L. Bouzar-Benlabiod; formal analysis, D. Khettaf; investigation, all authors; resources, all authors; data curation, D. Khettaf; writing—original draft preparation, D. Khettaf; writing—review and editing, all authors; visualization, all authors; supervision, L. Bouzar-Benlabiod; project administration, L. Bouzar-Benlabiod; funding acquisition, all authors.

Funding: This research didn't receive external funding.

Conflicts of Interest: The authors declare no conflict of interest.

 doi:10.20944/preprints202212.0409.v1

References

1. Thomas, T.; P Vijayaraghavan, A.; Emmanuel, S. Machine learning and cybersecurity. In *Machine Learning Approaches in Cyber Security Analytics*; Springer, 2020; pp. 37–47.
2. Liu, H.; Lang, B. Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences* **2019**, *9*, 4396.
3. Zhang, B.; Yu, Y.; Li, J. Network intrusion detection based on stacked sparse autoencoder and binary tree ensemble method. In Proceedings of the 2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2018, pp. 1–6.
4. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* **2013**.
5. Kurakin, A.; Goodfellow, I.J.; Bengio, S. Adversarial examples in the physical world. In *Artificial intelligence safety and security*; Chapman and Hall/CRC, 2018; pp. 99–112.
6. Ibitoye, O.; Shafiq, O.; Matrawy, A. Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks **2019**. pp. 1–6.
7. Grosse, K.; Manoharan, P.; Papernot, N.; Backes, M.; McDaniel, P. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280* **2017**.
8. Warzyński, A.; Kołaczek, G. Intrusion detection systems vulnerability on adversarial examples. In Proceedings of the 2018 Innovations in Intelligent Systems and Applications (INISTA). IEEE, 2018, pp. 1–4.
9. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* **2014**.
10. NSL-KDD. https://www.unb.ca/cic/datasets/nsl.html. [Accessed 09-May-2022].
11. Yang, K.; Liu, J.; Zhang, C.; Fang, Y. Adversarial examples against the deep learning based network intrusion detection systems. In Proceedings of the MILCOM 2018-2018 ieee military communications conference (MILCOM). IEEE, 2018, pp. 559–564.
12. Merzouk, M.A.; Cuppens, F.; Boulahia-Cuppens, N.; Yaich, R. A deeper analysis of adversarial examples in intrusion detection. In Proceedings of the International Conference on Risks and Security of Internet and Systems. Springer, 2020, pp. 67–84.
13. Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European symposium on security and privacy (EuroS&P). IEEE, 2016, pp. 372–387.
14. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 ieee symposium on security and privacy (sp). Ieee, 2017, pp. 39–57.
15. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2574–2582.
16. Hashemi, M.J.; Keller, E. Enhancing robustness against adversarial examples in network intrusion detection systems. In Proceedings of the 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, 2020, pp. 37–43.
17. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* **2018**, *1*, 108–116.
18. Wang, N.; Chen, Y.; Xiao, Y.; Hu, Y.; Lou, W.; Hou, T. Manda: On adversarial example detection for network intrusion detection system. *IEEE Transactions on Dependable and Secure Computing* **2022**.
19. Pujari, M.; Cherukuri, B.P.; Javaid, A.Y.; Sun, W. An approach to improve the robustness of machine learning based intrusion detection system models against the carlini-wagner attack. In Proceedings of the 2022 IEEE International Conference on Cyber Security and Resilience (CSR). IEEE, 2022, pp. 62–67.
20. IDS 2018 | Datasets | Research | Canadian Institute for Cybersecurity | UNB — unb.ca. https://www.unb.ca/cic/datasets/ids-2018.html. [Accessed 02-Dec-2022].
21. Zhang, C.; Costa-Pérez, X.; Patras, P. Tiki-taka: Attacking and defending deep learning-based intrusion detection systems. In Proceedings of the Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop, 2020, pp. 27–39.
22. Ilyas, A.; Engstrom, L.; Athalye, A.; Lin, J. Black-box adversarial attacks with limited queries and information. In Proceedings of the International Conference on Machine Learning. PMLR, 2018, pp. 2137–2146.
23. Brendel, W.; Rauber, J.; Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248* **2017**.
24. Schott, L.; Rauber, J.; Bethge, M.; Brendel, W. Towards the first adversarially robust neural network model on MNIST. *arXiv preprint arXiv:1805.09190* **2018**.
25. Cheng, M.; Le, T.; Chen, P.Y.; Yi, J.; Zhang, H.; Hsieh, C.J. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457* **2018**.
26. Gretton, A.; Borgwardt, K.M.; Rasch, M.J.; Schölkopf, B.; Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research* **2012**, *13*, 723–773.
27. Arik, S.Ö.; Pfister, T. Tabnet: Attentive interpretable tabular learning. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2021, Vol. 35, pp. 6679–6687.
28. Ali, M.H.; Al Mohammed, B.A.D.; Ismail, A.; Zolkipli, M.F. A new intrusion detection system based on fast learning network and particle swarm optimization. *IEEE Access* **2018**, *6*, 20255–20261.