Evading Deep Reinforcement Learning-based Network Intrusion Detection with Adversarial Attacks

Mohamed Amine Merzouk, Joséphine Delas, Christopher Neal, Frédéric Cuppens, Nora Boulahia-Cuppens, Reda Yaich

Evading Deep Reinforcement Learning-based Network Intrusion Detection with Adversarial Attacks

Mohamed Amine Merzouk*†
Polytechnique Montréal
Montreal, Canada
IRT SystemX
Palaiseau, France

Joséphine Delas*†
Polytechnique Montréal
Montreal, Canada
IRT SystemX
Palaiseau, France

Christopher Neal
Polytechnique Montréal
Montreal, Canada
IRT SystemX
Palaiseau, France

Frédéric Cuppens
Polytechnique Montréal
Montreal, Canada

Nora Boulahia-Cuppens
Polytechnique Montréal
Montreal, Canada

Reda Yaich
IRT SystemX
Palaiseau, France

ABSTRACT

An Intrusion Detection System (IDS) aims to detect attacks conducted over computer networks by analyzing traffic data. Deep Reinforcement Learning (Deep-RL) is a promising lead in IDS research, due to its lightness and adaptability. However, the neural networks on which Deep-RL is based can be vulnerable to adversarial attacks. By applying a well-computed modification to malicious traffic, adversarial examples can evade detection. In this paper, we test the performance of a state-of-the-art Deep-RL IDS agent against the Fast Gradient Sign Method (FGSM) and Basic Iterative Method (BIM) adversarial attacks. We demonstrate that the performance of the Deep-RL detection agent is compromised in the face of adversarial examples and highlight the need for future Deep-RL IDS work to consider mechanisms for coping with adversarial examples.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Computing methodologies → Machine learning**.

KEYWORDS

adversarial machine learning, adversarial examples, intrusion detection, reinforcement learning, evasion attacks

ACM Reference Format:
Mohamed Amine Merzouk, Joséphine Delas, Christopher Neal, Frédéric Cuppens, Nora Boulahia-Cuppens, and Reda Yaich. 2022. Evading Deep Reinforcement Learning-based Network Intrusion Detection with Adversarial Attacks. In *The 17th International Conference on Availability, Reliability and Security (ARES 2022), August 23–26, 2022, Vienna, Austria.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3538969.3539006

*Both authors contributed equally to this research.
†Corresponding authors {mohamed-amine.merzouk,josephine.delas}@polymtl.ca

ARES 2022, August 23–26, 2022, Vienna, Austria

ACM ISBN 978-1-4503-9670-7/22/08…$15.00
https://doi.org/10.1145/3538969.3539006

1 INTRODUCTION

Concern about security attacks on modern connected systems such as internet-connected devices or critical data servers has been growing for the past two decades. Intrusion Detection Systems (IDSs) are thus widely used as an automatic way of detecting potential threats within network connections, and their performances are constantly challenged to cope with the development of increasingly sophisticated cyberattacks.

Supervised Learning (SL) has introduced a whole new set of capabilities into IDS technology, leading to spectacular progress in intrusion detection tasks [2]. Still, a particularly difficult task for an IDS remains the detection of previously unseen anomalies (i.e. zero-day attacks). Reinforcement learning (RL) is a promising lead in IDS research, as it constitutes an adaptive and responsive environment suitable for online training, resulting in simple and fast prediction agents [5, 10]. However, the most efficient RL-based IDS implementations use Deep Neural Networks (DNNs) at their core, which have been shown to be vulnerable to adversarial examples [6, 9]. These attacks involve slightly modifying data samples in order to mislead a classification model. Previous work has evaluated the effects of adversarial examples on DNN-based IDSs [11], yet little is known about the vulnerability of RL-based detection methods to adversarial examples.

In this paper, we investigate the performance of a state-of-the-art Deep-RL intrusion detection agent when exposed to adversarial attacks. Caminero *et al.* [3] present a novel approach that has been shown to outperform other RL and SL-based detection models. They trained a Deep-RL agent in an adversarial environment using the NSL-KDD dataset [19]. In this paper, we show how adversarial examples generated using two methods [4, 8] can evade the detection of the agent. In keeping the consistency with initial studies in this domain, we consider white-box individual attacks where the intruder has access to the parameters of the model [9, 16].

The remainder of the paper is organized as follows. Section 2 provides an overview of influential works applying RL to IDSs and a review of adversarial example generation methods for Deep-RL agents. The methodology for this paper is provided in Section 3, where we describe the dataset, the RL detection agent, and the adversarial attacks used in our experiments. In Section 4, we present the results achieved by adversarial examples on the performance of the agent. A discussion of the immediate practicality of these

attacks and an outline for future work is provided in Section 5. Lastly, Section 6 provides some concluding remarks.

2 RELATED WORK

RL techniques have an extensive range of applications in cybersecurity due to their adaptive nature and the rapidity of their predictive models. The first works concerning RL and intrusion detection were published in the early 2000s and present mostly innovative works using tabular methods. Servin *et al.* [17] use a Q-learning algorithm based on a look-up table to detect network intrusions, whereas Xu *et al.* [20] introduce Temporal Difference (TD) learning algorithms for live detection.

More recently, the development of Deep-RL algorithms has further improved the performances of IDS models [10]. In particular, Caminero *et al.* [3] present an innovative multi-agent deep reinforcement learning model that outperforms previous tabular methods, as well as several other DNN models. Their algorithm is based on the concurrency of two different agents to improve the predictions.

Despite the remarkable performance shown by RL agents in intrusion detection, there is a concern about their reliability in the presence of adversarial attacks. Since Deep-RL agents rely on DNNs, they could be vulnerable to malicious inputs, chiefly adversarial examples. Behzadan *et al.* [1] first explored the effect of adversarial examples on Deep Q-Networks (DQNs). The authors use two well-known attacks, namely, Fast Gradient Sign Method (FGSM) [4] and Jacobian-based Saliency Map Attack (JSMA) [16], to perturb the training of a game-learning agent. They also demonstrate the transferability of adversarial examples between agents. Huang *et al.* [6] show how an adversary could interfere with the operations of a trained RL agent. The authors use FGSM to generate adversarial examples in both white-box and black-box settings by utilizing the transferability property [15]. In their study, Kos *et al.* [7] compare the effectiveness of adversarial examples with random noise. They show how the value function can indicate opportune moments to inject perturbations and how adversarial re-training can enhance the resilience of RL agents [4]. Lin *et al.* [9] introduce two novel methods to attack Deep-RL agents using adversarial examples. These are referred to as the *strategically-timed attack*, which aims to introduce perturbations at critical moments, and the *enchanting attack*, which aims to lure an agent to a certain state maliciously. Using these methods, the authors demonstrate they are able to significantly decrease the accumulated rewards collected by a DQN and an Asynchronous Advantage Actor-Critic (A3C) agent on five different Atari games.

These previous studies demonstrate that Deep-RL agents are vulnerable to well-crafted adversarial examples. They propose different methods for attacking Deep-RL agents before and after training, as well as, in white-box and black-box settings. It has even been suggested to remediate the effect of adversarial examples against Deep-RL agents using adversarial re-training [7]. While there is a rich body of work studying how adversarial examples can degrade the performance of Deep-RL models, these previous works investigate attacks against agents used in control problems, particularly the playing of Atari video games. Such models are significantly different from the agent presented in this paper, as we will develop later in Section 3.2, since the successive states are independent of the action taken in the previous step, thus affecting the learning process. In addition, evading an intrusion detection model involves targeting a specific class (labeling malicious connections as normal behavior); while working most of the time with imbalanced datasets [21]. For these reasons, we notice a gap in the literature concerning the understanding of adversarial attacks against Deep-RL-based intrusion detection agents and present this work as an initial building block toward filling this gap.

3 METHODOLOGY

First, we present the dataset that we use for the training and the validation of the the detection agent. Then, we describe the Deep-RL detection agent used in our experiments, as proposed by Caminero *et al.* [3]. Finally, we outline the adversarial attacks we use against the agent.

3.1 Dataset

For comparative studies of our results, we opt for the commonly used NSL-KDD dataset [19]. This dataset is widely used in similar research papers, and particularly in Caminero *et al.* [3] to validate the agent.

Each record is composed of 41 network features: 38 continuous (such as the duration of the connection) and 3 categorical. A record is labeled as either normal or an attack. There are 22 different attack types in the training set (therefore, 23 different label outcomes) but 38 in the testing set: an efficient detection model will have to detect anomalies it has not encountered during training. From this basis, a few preprocessing steps were applied: categorical features were one-hot encoded, and non-binary features were normalized (i.e. zero mean, standard deviation equal to one).

Finally, in this work, we aim to mislead the model into classifying an attack record as a normal one (i.e. a false negative classification). Therefore, we do not need to be able to differentiate the 23 anomaly type. Instead, we group them into 4 classes of attacks. The approximately 120,000 samples are thus distributed into the following classes: Normal (53.46%), Denial-of-Service (DoS) (36.46%), Probing (PROBE) (9.25%), Remote-to-Local (R2L) (0.79%), and User-to-Root (U2R) (0.04%).

3.2 Detection Model

We work with a state-of-the-art Deep-RL intrusion detection agent that has been shown to outperform other DNN and Deep-RL methods on the KDD-NSL dataset [3]. The agent is referred to as Adversarial Environment using Reinforcement Learning (AE-RL); since it enhances its learning phase by using an adversarial environment to select training samples. It is composed of two concurrent agents: the first agent is the classifier that predicts the labels for each sample, whereas the second agent is a selector that acts as a simulated environment and feeds sample records to the classifier. Therefore, the second agent is only used during training to obtain a more robust model and is not involved in the attack detection.

The classifier is a Deep Q-Network (DQN) agent [13], described in Figure 1. With the states (record features) as input, its goal is to choose the best action according to its Q-function [18]. The Q-function is simulated by a fully-connected, 3-layer neural network with 100 units per layer, that is trained to approximate the optimal

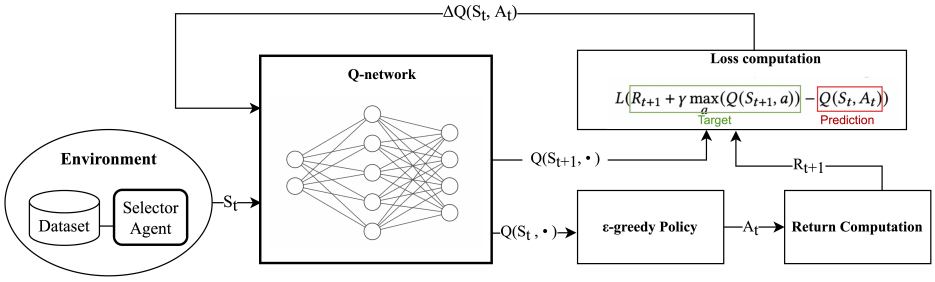


Figure 1: Details of the classifier DQN agent for the training phase

Q-function defined by the Bellman equation in Equation 1:

$$Q^*(S_t, A_t) = R_{t+1} + \gamma \max_a(Q(S_{t+1}, a)), \tag{1}$$

where Q^* is the optimal Q-function, S, A, and R are the states, actions, and rewards, γ is the discount factor, and t and $t+1$ are the timesteps.

During training, the error between the target and the predicted Q-value is back-propagated through the model's parameters. It is calculated with Huber loss, which is quadratic if the absolute difference falls below 1 and linear otherwise. This loss function provides smoothness near zero while being less sensitive to outliers than the squared error loss. After predicting the state's Q-value, the agent chooses the action according to an ϵ-greedy policy [18]: randomly with a probability of ϵ, else the one that maximizes the Q-value. The ϵ value is high at the beginning and reduces over the course of the training process. When the training is done, ϵ is set to zero in order to optimize the prediction.

In this setup, the states are data records issued from the dataset and the actions are the different possible label outputs. The reward is set to 1 if the classifier is correct and 0 otherwise. Finally, the discount factor γ is set to a value close to zero, since the states do not influence one another and each state is independent of the precedent.

When transitioning from one step to the next, selecting random samples from the dataset is not the most efficient solution due to the unbalanced nature of the dataset. The selector agent instead chooses which anomaly category to pull the next state and attempts to find the most difficult records for the classifier. The selector's algorithm is DQN with Huber loss and epsilon-greedy policy, similar to the first agent, but the rewards are opposite. That is, -1 if the classifier chooses correctly and 0 otherwise. The two agents are considered concurrent because of this method for providing rewards.

Once the training is complete, the prediction phase only consists of passing a record through a small fully-connected neural network and choosing the maximum output. This simple architecture allows for efficient classification, which is critical in intrusion detection tasks.

3.3 Adversarial Attacks

When the training is complete, we use adversarial examples to mislead the agent on test data. A perturbation is computed using the following generation methods and added to the original test data. This perturbation will corrupt the prediction of the Q-values and influence the decision of the agent. These attacks were implemented using the Adversarial Robustness Toolbox (ART) library [14].

3.3.1 Fast Gradient Sign Method. The first attack we use is the Fast Gradient Sign Method (FGSM) introduced by Goodfellow *et al.* [4]. This method exploits the gradient of the loss function, which usually serves to update the parameters of the model. Instead, the gradient is propagated back to the inputs and its sign guides the perturbation. An adversarial example x' is formed by adding the perturbation amplitude ϵ with the sign of the gradient to an original example x. Equation 2 describes this perturbation, where ∇ is the gradient function, J_θ is the loss function with regards to the parameters θ, and l is the true label of the example.

$$x' = x + \epsilon \cdot sign(\nabla J_\theta(x, l)) \tag{2}$$

3.3.2 Targeted Fast Gradient Sign Method. FGSM is an untargeted attack by definition, as it does not aim to misclassify the adversarial example towards a specific class. However, it would not be in the interest of attackers to misclassify an attack as another type of attack since evading detection implies classifying the attacks as normal traffic. To targeted a specific class using FGSM, we perform the update in Equation 3 where l' is the target class.

$$x' = x - \epsilon \cdot sign(\nabla J_\theta(x, l')) \tag{3}$$

3.3.3 Basic Iterative Method. Kurakin *et al.* [8] introduce the Basic Iterative Method (BIM) as an extension of FGSM. The idea is to apply small perturbations over several steps to create more precise adversarial examples. Additionally, a clipping method is used at each step to prevent features from exceeding valid intervals. Generally, increasing the number of iterations will produce finer perturbations and can lead to more subtle adversarial examples. However, there is a trade-off, as computing these small steps is typically slower to produce adversarial examples than non-iterative methods.

3.3.4 Targeted Basic Iterative Method. Applying BIM involves using FGSM, as outlined in Equation 2, to generate an adversarial example for some unspecified class and may not necessarily serve the goal of the attacker. Using the BIM process with targeted FGSM, as outlined in Equation 3, produces an adversarial example for a particular class.

4 RESULTS

In this section, we present the results of our experiments. We evaluate the performance of the trained agent using the test set, of approximately 30,000 samples, with and without adversarial perturbation. In all adversarial attacks, we set the maximum amount of perturbation to $\epsilon = 0.1$.

4.1 Two-class Attack Detection Facing Adversarial Examples

This section involves experiments using two-class detection, where the detection agent assigns a label of `Normal` or `Anomaly` to each sample. We only consider the generic FGSM and BIM attacks; since the attacker intends to make anomalous packets appear legitimate. The performance of the detection agent is shown in Figure 2, the accuracy and F1-scores are shown in Table 1, and the confusion matrices of the detection agent for the label decisions are shown in Figure 3.

	No Attack		FGSM		BIM	
Label	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
Normal	84.81	84.09	66.87	61.16	75.84	66.71
Anomaly	84.81	85.47	66.87	71.12	75.84	81.04

Table 1: Accuracy and F1-score of AE-RL two-class detection model facing adversarial attacks

No Attack. In this case, the training set is balanced (i.e. 53% normal and 47% anomalies), which allows the agent to learn the two classes accurately. In Figure 3, we can see that 79% of the anomalies are detected by this model, with 84.81% accuracy and 84.09% F1-score. These results correspond to state-of-the-art performance on NSL-KDD, even though Figure 2 shows that about 2800 anomalies are classified as normal traffic.

Fast Gradient Sign Method. We observe, in Figure 3, a significant drop in the number of true positives from both classes, but what is particularly interesting is the false positive rate from the `Normal` class. Indeed, it rose from 0.21 in the baseline model to 0.28 with the FGSM examples, which means that this attack increased the number of suspicious connections undetected by the model.

Basic Iterative Method. BIM is supposed to generate more precise perturbations, finding new paths to escape the prediction. In Table 1, we notice a drop in the performance of the model compared to the baseline; the accuracy drops from 84.81% to 75.84%, and the F1-score also drops from 84.09% to 66.71% for the anomaly class. However, we notice in Figure 2 that most of the misclassified items were initially labeled as `Normal`, and the model was lured into labeling them as `Anomaly`. A targeted attack, in a multi-class context, can improve the deception by aiming toward the `Normal` class for all examples.

4.2 Multi-class Attack Detection Facing Adversarial Examples

This section involves multi-class detection, where the agent must choose a label of `Normal` or one of the attack categories of `DoS`, `PROBE`, `R2L`, and `U2R`. The performance of the detection agent is shown in Figure 4, the accuracy and F1-scores are shown in Table 2, and the confusion matrices of the detection agent are shown in Figure 5.

No Attack. Without the presence of adversarial examples, we see that the agent has an overall good performance with an accuracy of 83.70%. The F1-scores for the `Normal` and `DoS` categories are 83.33% and 91.32% respectively. However, the agent shows weak performance on `R2L` and `U2R` attacks, where the F1-score is below 40%. This is common in many classifiers because these attack types are hard to detect and are underrepresented in the dataset. Figure 5 shows a noticeable intensity on the diagonal, especially for the `Normal`, `DoS`, and `Probe` categories. The `R2L` examples are often classified as `Normal` while the `U2R` are spread across different categories.

Untargeted Fast Gradient Sign Method. Applying adversarial perturbations using FGSM shows an important drop in the performance of the agent. We see a large number of misclassifications for all classes in Figure 4. The accuracy of the agent drops to 75.58%, while the F1-Score of all classes is substantially lower. The false positive rate of the class `Normal` shows that most of the examples are misclassified in this category. The same results are shown in Figure 5, as the diagonal is less intense and the `Normal` column (corresponding to the examples classified as `Normal`) is more intense.

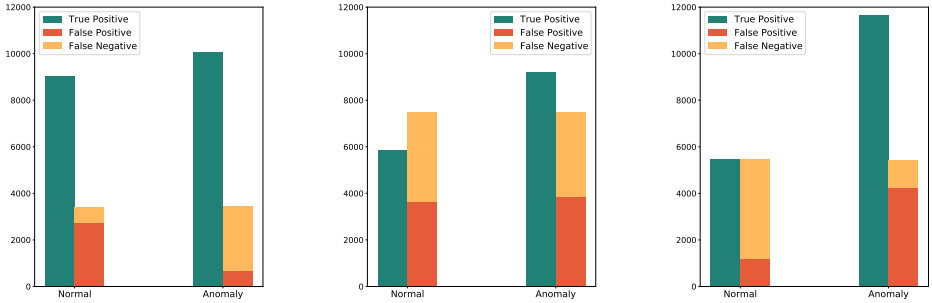
Targeted Fast Gradient Sign Method. With targeted FGSM, adversarial examples are pushed toward the `Normal` target class. We see a noticeable impact in Figure 4 with an even higher number of false positives in the `Normal` class. The accuracy and F1-score for the `Normal` label drop to 56.37% and 65.89% respectively. The confusion matrix in Figure 5 shows a very intense concentration in the `Normal` column. This indicates that targeted attacks can be more interesting for attackers who want to evade an RL-based IDS.

Untargeted Basic Iterative Method. By applying the untargeted BIM method, we find no real change to the performance of the detection agent compared to when no attack is present. This can be explained by the perturbation steps limit (set to 100). With no target class for this attack, the perturbations do not go far enough in a particular direction to modify the class of the sample from the viewpoint of the detection agent. Without any computation limitations, we would expect this method to cause a more severe impact on the detection performance.

Targeted Basic Iterative Method. With targeted BIM, we see the most drastic degradation in the performance of the detection agent. This method can perturb more precisely anomalous samples to the `Normal` class. The accuracy on `Normal` samples drops to 28.47% and the F1-score for all labels drops below 20%. The substantial amount of misclassifications of anomalous packets as `Normal` is demonstrated in Figures 4 and 5.

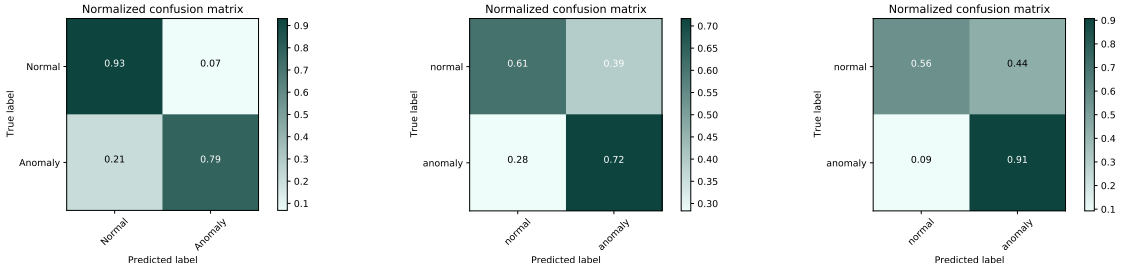
5 DISCUSSION AND FUTURE WORK

The results of our experiment show how adversarial examples can degrade the performance of Deep-RL IDSs. However, more work needs to be done to prove the vulnerability of Deep-RL IDSs to adversarial examples. In reality, an attacker would likely need



(a) No Attack **(b)** FGSM **(c)** BIM

Figure 2: Performance of AE-RL two-class detection model facing adversarial attacks

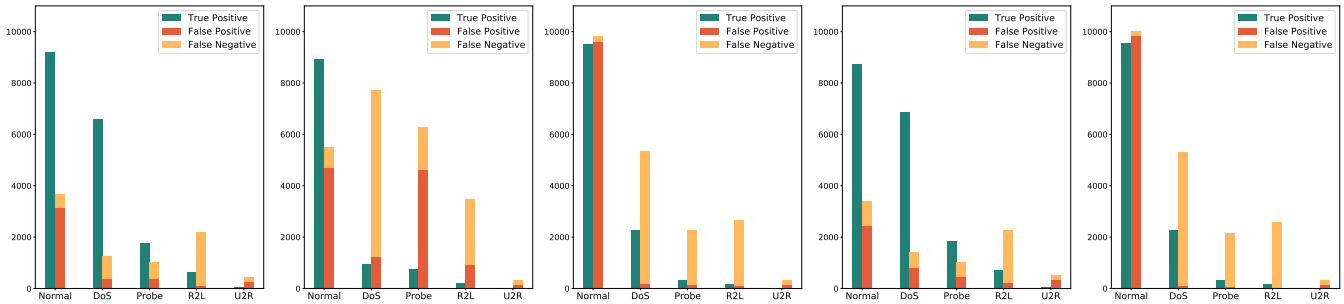


(a) No Attack **(b)** FGSM **(c)** BIM

Figure 3: Confusion matrices of AE-RL two-class detection model facing adversarial attacks

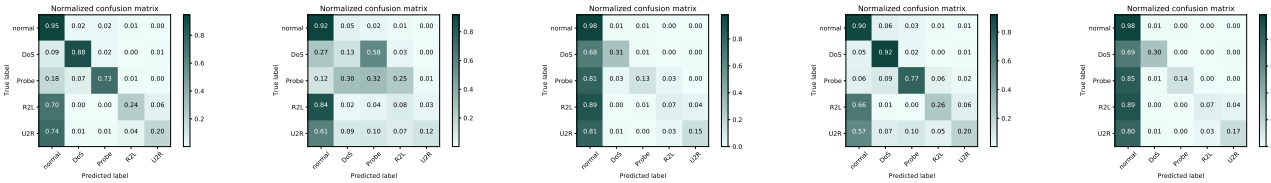
	No Attack		FGSM		Targ. FGSM		BIM		Targ. BIM	
Label	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
Normal	83.70	83.33	75.58	76.40	56.37	65.89	84.90	83.71	28.47	2.52
DoS	94.44	91.32	65.73	120.04	76.36	46.23	93.76	90.71	24.81	6.38
PROBE	95.40	77.36	72.09	119.68	89.93	21.83	95.41	78.27	79.62	13.87
R2L	90.25	37.47	84.68	111.25	88.16	12.17	89.91	39.03	84.19	18.93
U2R	98.10	15.44	98.55	112.83	98.52	15.73	97.76	13.99	97.09	8.13

Table 2: Accuracy and F1-score of AE-RL mutli-class detection model facing adversarial attacks



(a) No Attack **(b)** Untargeted FGSM **(c)** Targeted FGSM **(d)** Untargeted BIM **(e)** Targeted BIM

Figure 4: Performance of AE-RL multi-class detection model facing adversarial attacks



(a) No Attack **(b)** Untargeted FGSM Targeted FGSM **(d)** Untargeted BIM **(e)** Targeted BIM

Figure 5: Confusion matrices of AE-RL detection model facing adversarial attacks

to adversarially modify a stream of malicious packets to evade the IDS. Our experiments are limited to the data instance level and would need to be implemented on a real network. This raises concerns regarding the practicality of these adversarial examples. Recently study [12] has identified several invalidation properties in adversarial examples generated on 3 intrusion detection datasets that prevent their implementation. These properties include out-of-range values, corrupt binary values, multiple categories belonging, and corrupt semantic relations. For the purpose of this work, we demonstrate that current adversarial attacks can bypass Deep-RL detection agents at the data level, but we do not solve the practicality issues at the network level.

An avenue for future work is to investigate the impact of other adversarial attacks on Deep-RL IDSs, including in the more realistic black-box setting. The detection agents should also be trained on recent datasets that are more representative of modern network traffic and attacks. The practicality concerns should be addressed by applying clipping functions and penalties to restrict the perturbation. Semantic relations should be extracted from the data and integrated into the generation methods to produce consistent adversarial examples.

6 CONCLUSION

Recent research in cybersecurity has used Deep-RL in multiple functions, especially intrusion detection. This approach is promising as it allows more adaptability and faster processing. However, using Deep-RL detection methods opens the door to the threat of adversarial attacks.

In this work, we study the vulnerability of a Deep-RL IDS detection agent when faced with adversarial examples. We train a state-of-the-art Deep-RL detection agent using the NSL-KDD dataset and evaluate its performance with several adversarial attack methods. We demonstrate a substantial deterioration in detection performance when adversarial attacks are used to perturb malicious packets towards being classified as benign. Finally, we discuss the practicality of these adversarial examples and suggest research directions to implement adversarial attacks on real networks.

REFERENCES

[1] Vahid Behzadan and Arslan Munir. 2017. Vulnerability of Deep Reinforcement Learning to Policy Induction Attacks. In *Machine Learning and Data Mining in Pattern Recognition*.

[2] Anna L. Buczak and Erhan Guven. 2016. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys Tutorials* (2016).

[3] Guillermo Caminero, Manuel Lopez-Martin, and Belen Carro. 2019. Adversarial environment reinforcement learning algorithm for intrusion detection. *Computer Networks* (2019).

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations*.

[5] Zhisheng Hu, Ping Chen, Minghui Zhu, and Peng Liu. 2019. *Reinforcement Learning for Adaptive Cyber Defense Against Zero-Day Attacks*.

[6] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *5th International Conference on Learning Representations*.

[7] Jernej Kos and Dawn Song. 2019. Delving into adversarial attacks on deep policies. *5th International Conference on Learning Representations*.

[8] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *5th International Conference on Learning Representations*.

[9] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. 2017. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*.

[10] Manuel Lopez-Martin, Belen Carro, and Antonio Sanchez-Esguevillas. 2020. Application of deep reinforcement learning to intrusion detection for supervised problems. *Expert Systems with Applications* (2020).

[11] Mohamed Amine Merzouk, Frédéric Cuppens, Nora Boulahia-Cuppens, and Reda Yaich. 2021. A Deeper Analysis of Adversarial Examples in Intrusion Detection. In *15th International Conference on Risks and Security of Internet and Systems*.

[12] Mohamed Amine Merzouk, Frédéric Cuppens, Nora Boulahia-Cuppens, and Reda Yaich. 2022. Investigating the practicality of adversarial evasion attacks on network intrusion detection. *Annals of Telecommunications* (2022).

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* (2015).

[14] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian M. Molloy, and Ben Edwards. 2019. Adversarial Robustness Toolbox v1.0.0. *arXiv:1807.01069* (2019).

[15] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks against Machine Learning. In *Proceedings of the ACM on Asia Conference on Computer and Communications Security*.

[16] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy*.

[17] Arturo Servin and Daniel Kudenko. 2008. Multi-agent Reinforcement Learning for Intrusion Detection. In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*.

[18] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[19] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *IEEE symposium on computational intelligence for security and defense applications*.

[20] Xin Xu and Tao Xie. 2005. A Reinforcement Learning Approach for Host-Based Intrusion Detection Using Sequences of System Calls. In *Advances in Intelligent Computing*.

[21] Ibrahim Yilmaz, Rahat Masum, and Ambareen Siraj. 2020. Addressing Imbalanced Data Problem with Generative Adversarial Network For Intrusion Detection. In *IEEE 21st International Conference on Information Reuse and Integration for Data Science*.