Generative Adversarial Attacks Against Intrusion Detection Systems Using Active Learning

Dule Shu
Carnegie Mellon University
Pittsburgh, PA
dules@andrew.cmu.edu

Nandi O. Leslie
U.S. Army Research Laboratory
Adelphi, MD
nandi.o.leslie.ctr@mail.mil

Charles A. Kamhoua
U.S. Army Research Laboratory
Adelphi, MD
charles.a.kamhoua.civ@mail.mil

Conrad S. Tucker
Carnegie Mellon University
Pittsburgh, PA
conradt@andrew.cmu.edu

ABSTRACT

Intrusion Detection Systems (IDS) are increasingly adopting machine learning (ML)-based approaches to detect threats in computer networks due to their ability to learn underlying threat patterns/features. However, ML-based models are susceptible to adversarial attacks, attacks wherein slight perturbations of the input features, cause misclassifications. We propose a method that uses active learning and generative adversarial networks to evaluate the threat of adversarial attacks on ML-based IDS. Existing adversarial attack methods require a large amount of training data or assume knowledge of the IDS model itself (e.g., loss function), which may not be possible in real-world settings. Our method overcomes these limitations by demonstrating the ability to compromise an IDS using limited training data and assuming no prior knowledge of the IDS model other than its binary classification (i.e., benign or malicious). Experimental results demonstrate the ability of our proposed model to achieve a 98.86% success rate in bypassing the IDS model using only 25 labeled data points during model training. The knowledge gained by compromising the ML-based IDS, can be integrated into the IDS in order to enhance its robustness against similar ML-based adversarial attacks.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**;

KEYWORDS

intrusion detection, neural networks, active learning

ACM Reference Format:
Dule Shu, Nandi O. Leslie, Charles A. Kamhoua, and Conrad S. Tucker. 2020. Generative Adversarial Attacks Against Intrusion Detection Systems Using Active Learning. In *ACM Workshop on Wireless Security and Machine Learning (WiseML '20), July 13, 2020, Linz (Virtual Event), Austria.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3395352.3402626

WiseML '20, July 13, 2020, Linz (Virtual Event), Austria

ACM ISBN 978-1-4503-8007-2/20/07.
https://doi.org/10.1145/3395352.3402626

1 INTRODUCTION

Intrusion Detection Systems (IDS) play an essential role in defending computer networks against malware attacks. Among the various approaches for anomaly-based intrusion detection, Machine Learning(ML) algorithms have gained increasing attention due to their advantage in detecting zero-day attacks [10]. Moreover, the fast-developing software-defined networks with programmable controllers have provided a convenient platform to implement the ML-based IDS [23]. Various ML models, including support vector machine [25], decision tree [6], k-NN [11], naive Bayes [8] and artificial neural networks [9] have been applied to intrusion detection. In general, these ML methods can be considered as a classifier for anomaly detection.

The development of IDS is accompanied by the evolution of malware and intrusion strategies. In particular, an alarming direction appears in malware evolution which is the development of self-adaptive malware that is capable of adjusting its behavior to avoid detection by a security mechanism. Wu *et al.* [26] proposes a deep Q-learning method to bypass botnet detection models by controlling the network traffic flow generated by the botnets. Shi *et al.* [22] use Generative Adversarial Networks (GAN) to synthesize training data for exploratory attacks and causative attacks on a real online classifer for text subjectivity analysis. Erpek *et al.* [3] propose a method for efficient jamming attacks on wireless communication channels which uses a deep neural network model to predict the channel status and a GAN model to accelerate the prediction model training. GANs are used in [20] to modify the network traffic in the Command and Control (C2) channel of a Remote Access Trojan malware such that the modified traffic mimics the traffic profile of Facebook chat. Lin *et al.* [13] also proposes a GAN-based method where the generator computes the adversarial network traffic features to attack a Black-box IDS model. The different methods proposed in [26], [20] and [13] can be grouped to a special class of attack named the *adversarial attack* which targets the ML-based IDS models.

Adversarial attack refers to an action of (intentionally) incurring classification error of a ML algorithm by perturbing the value of its input feature point. The perturbed feature, named the adversarial feature or the adversarial example, is close to the original feature in terms of considered distance metrics (*e.g.*, Euclidean distance). An adversarial attack algorithm can be used to compute the adversarial feature of a malicious network traffic flow. Once the value of an

adversarial feature is found, the malware can adjust the way it generates the network traffic flow to meet the computed value, such that the IDS will classify the adjusted traffic flow as benign traffic. The small magnitude of the feature value perturbation allows the malware to have a high chance in retaining its intrusive functionality. A detailed review of adversarial attacks against intrusion detection can be found in [14].

Although the methods proposed in [26], [20] and [13] have shown effectiveness in compromising a ML-based IDS model, their threat in actual network environments are limited due to the ideal assumptions that a large amount of data labels for training is available, and that the prior knowledge about the IDS model loss function is known. To assess whether an adversarial attack against IDS can pose a larger threat to network security if these ideal assumptions are removed, we propose a method, named the Generative Adversarial Active Learning (Gen-AAL) algorithm, to compute the adversarial feature of network traffic flow. Our proposed method makes the following contributions.

- We propose a GAN model which constrains the perturbation to the original feature.
- Our proposed method does not require the knowledge of the internal structure of the IDS model or the loss function for IDS model training, and hence is a more practical method for adversarial attack.
- Our method reduces the required number of times to query the black-box IDS model for labeled data points to train the GAN model, which increases the efficiency of GAN training and the generation of adversarial feature points.

The rest this paper is organized as follows. Section 2 provides a review of the literature mostly related to this work. Section 3 introduces the procedure of adversarial attack using the proposed method. Section 4 shows the experimental result. Conclusion and future work are discussed in Section 5.

2 BACKGROUND

2.1 Malware Threats to Computer Networks

In computer networks, malware (*e.g.*, a botnet) can establish a Command and Control (C2) channel between the attacker and the victim computers and use it to acquire remote access to the victim computer [19]. Malware using the C2 channel mostly target personal computers, but has started to attack mobile devices as well since 2010 [15]. A typical example of such malware for mobile devices is Plankton [28], which is included in host apps via an added background service. Once the infected app runs, the background service will start to collect information, such as the device ID and the list of granted permissions, which will be used by the attacker [7]. Other examples of malware spreading through wireless communication channels include Cabir, a worm malware for Symbian operating system that spreads via bluetooth, and Spitmo, a Trojan horse malware running on Andriod operating system which steals information from the infected smartphones and uploads the information to a remote server [15].

To hide the malware network traffic in C2 channel from an IDS, the methods proposed in [26], [20] and [13] train a ML model to learn the pattern of benign network traffic features and make the malware network traffic features mimic the benign features. The

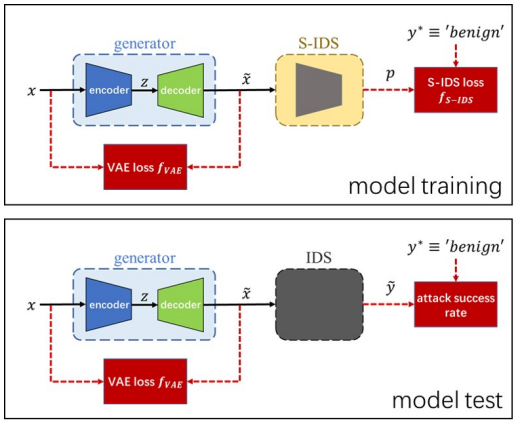

Figure 1: The overall structure of the GAN model.

reinforcement learning agent in [26] is trained on 40,000 flows, where each flow yields an adversarial example to be sent to the IDS model for labeling ('benign' or 'malicious'). Similarly, the GAN model from [13] is trained for 100 epochs on the NSL-KDD dataset which consists of 1,074,992 distinct data samples. Such training process requires the IDS model to produce a large amount of labels for the generated adversarial features. In practice, the action of sending a large amount of adversarial traffic flows to an IDS for labeling could be identified as a suspicious activity and trigger blocking or other countermeasures from security systems which will interrupt the GAN training [1]. The work in [20] addresses this issue by setting up a five-minute time window for the actual IDS to collect generated traffic flow. However, this time window is used as a known parameter in the malware configuration, which is a condition that does not generally hold in practice since the time window is a part of the internal mechanism of an IDS. Another ideal condition required by [13] is that the loss function of the IDS model is known, since the IDS loss function value is used in calculating the GAN model loss. In a practical situation, however, it is not realistic to assume that the attacker has knowledge of the loss function used to train the IDS model. To remove the aforementioned ideal assumptions, we introduce a substitute IDS model to approximate the original IDS model, and propose an active learning algorithm to reduce the number of labels required for ML model training.

2.2 Active Learning

In supervised learning for classification, a ML model is trained with a labeled training dataset to estimate a mathematical mapping from a feature point to a target label. In cases where it is expensive or difficult to obtain the data labels, only a small number of labeled data points are available for model training, and the model performance is likely to suffer from the deficiency of training data [27]. One possible solution to address the issue of limited training data, as shown in [22], is to augment the original training dataset with synthesized data points produced by a generative neural network model. Another solution to maintain model performance under limited training data is active learning. Active learning is a family of methods which optimizes the process of training data collection

in order to build a training dataset with minimal size which still yields an adequate performance [15]. An active learning method is typically an iterative process that alternates between training the ML model (*e.g.*, a classifier) and augmenting the current training dataset with new labeled data points from the oracle (*e.g.*, a human annotator or a ML model). The process is typically initialized with a small training dataset and a large pool of unlabeled feature points. In each iteration, a set of feature points are selected from the unlabeled pool and sent to the oracle for labeling. The strategy to select these unlabeled feature points from the unlabeled pool is an important problem to solve in active learning.

Uncertainty selection [12] is a strategy to select feature points with the lowest confidence of the classifier. The level of uncertainty can be represented by the distance of a feature point to the decision boundary of the classifier. The active learning methods that rely on uncertainty selection measured by the distance to the decision boundary is named the margin-based active learning. Ducoffe *et al.* [2] propose a margin-based method named the DeepFool Adversarial Learning (DFAL) which selects unlabeled samples with the smallest adversarial perturbation. The adversarial perturbation is computed by an iterative process named DeepFool [17]. DFAL achieves the state-of-the-art performance in terms of fast convergence in convolutional neural network models and is used as a main reference to develop our proposed method.

3 METHOD

The adversarial attack problem presented in this work is formally described as follows. Given a black-box IDS model with unknown loss function which is able to classify a feature point x into the correct category ('benign' or 'malicious'), develop a ML model G that takes x as input and returns an adversarial feature point $\tilde{x}$ that alters the classification result of the IDS. A solution to this problem should satisfy the following two practical constraints: (1) the perturbation, defined as $\eta := \|x - \tilde{x}\|_2$, should be minimized, (2) the size of the labeled dataset $\mathcal{L}$ to train the ML model G should be minimized. The motivation for the first constraint is that by minimizing the distance between the original feature point and the adversarial feature points, we can increase the challenge for classification and the probability of retaining the malware's functionality. The motivation for the second constraint is that reducing the required labels for training will reduce the number of adversarial traffic flows sent to the black-box IDS model and thus will reduce the probability that the malware is detected.

To solve this problem, we propose the Gen-AAL algoirthm, which uses a GAN model and an active learning algorithm to compute adversarial attacks against an IDS model. The upper subplot of Fig. 1 shows the overall structure of the GAN model during training in Gen-AAL algorithm. The GAN model consists of a generator network and a discriminator network. The generator network is a Variational AutoEcoder (VAE) model comprising an encoder module and a decoder module. It takes a feature point x as input and outputs an adversarial feature point $\tilde{x}$. The reason we choose an VAE model for the generator is that the symmetric architecture of the encoder and the decoder provides a dimension-preserving mapping from x to $\tilde{x}$. The discriminator network is a Multi-Layer Perceptron (MLP) model that simulates the black-box IDS and is therefore named the Substitute-IDS (S-IDS) model. The S-IDS takes $\tilde{x}$ as input and outputs a probability p which indicates how likely the input feature point is benign. In model test, the S-IDS is replaced by the original black-box IDS model, which outputs a binary label $\tilde{y}$ indicating whether $\tilde{x}$ is benign or malicious, as shown in the lower subplot of Fig. 1. Since the objective of adversarial attack is to make the IDS mistakenly label the malicious feature as benign, the target label, denoted as y^*, is always set as 'benign'. The motivation for introducing the S-IDS model is to avoid a large number of times the IDS model is used during GAN training. Instead of using the IDS model to classify an adversarial feature, the S-IDS model is used to label an adversarial feature. The iterative process of active learning ensures that the S-IDS model approximates the true IDS model with a sufficient accuracy.

The objective of GAN training is to minimize the difference between $\tilde{y}$ and y^* as well as the difference between x and $\tilde{x}$. To achieve this objective, we define the following loss function to train the GAN model.

$$\begin{aligned} f_{\text{GAN}} =& \lambda_1 D_{KL}(p(z), q(z|x)) + \lambda_2 \|x - \tilde{x}\|_2 + \lambda_3 E(x, \tilde{x}) \\ &+ \lambda_4 E(\tilde{y}, y^*) \end{aligned} \tag{1}$$

where $D_{KL}(\cdot)$ is the Kullback–Leibler divergence, z is the output of the VAE encoder (the latent variable), p, q are the distributions of the prior and the encoder's output respectively, $E(\cdot)$ is the binary cross entropy function, $\tilde{y}$ is the label of $\tilde{x}$ assigned by S-IDS, and y^* is the target label for adversarial attack ('benign'). The coefficients λ_1 to λ_4 are tunable hyper-parameters for model training.

Prior to the GAN model training, the VAE model goes through a pretraining process in which it is trained to make its output $\tilde{x}$ identical to its input x. This step is introduced such that an adversarial feature point can be obtained faster in GAN training. The loss function for VAE pretraining is designed as follows.

$$f_{\text{VAE}} = \lambda_1 D_{KL}(p(z), q(z|x)) + \lambda_2 \|x - \tilde{x}\|_2 + \lambda_3 E(x, \tilde{x}) \tag{2}$$

The active learning algorithm controls the iterative retraining of the VAE model and the S-IDS model. At each iteration, it sends to the IDS model a set of unlabeled feature points generated by the VAE and collects binary labels of the feature points from the output of the IDS. The input feature points along with their labels form a labeled dataset. This dataset is added to the existing labeled dataset of network traffic features to form an augmented dataset for the next iteration of GAN model retraining. This iterative process of training data update and model retraining is shown in Fig. 2. A detailed description of Gen-AAL algorithm is shown in Algorithm Description 1. The loss function to train the S-IDS model is designed as the binary cross entropy function shown in the following equation.

$$f_{\text{S-IDS}} = E(\tilde{y}, y^*) = -\frac{1}{M} \sum_{i=1}^{M} [y_i^* log(\tilde{y}_i) + (1 - y_i^*) log(1 - \tilde{y}_i)], \tag{3}$$

where M denotes the dimension of the feature space.

The proposed Gen-AAL algorithm is derived based on the DFAL algorithm. Similar to DFAL, the Gen-AAL algoirthm selects unlabeled feature points to query which are closest to the decision boundary of the S-IDS, as these points contain higher level of uncertainty in approximating the IDS model, and clarification of the

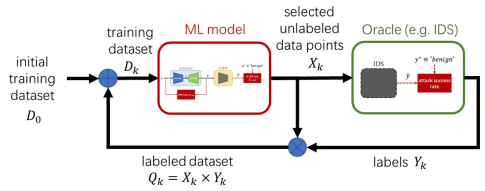

Figure 2: The block diagram of active learning.

Algorithm 1 Generative Adversarial Active Learning

Require: $\mathcal{L}$, set of labeled training data
Require: $\mathcal{U}$, set of unlabeled training data
Require: $\mathcal{H}$, set of hyper-parameters to train the networks
Require: K, the number of candidates
Require: n_{query}, the number of adversarial examples for query
Require: N, the maximal iteration of model retraining

1: #initialization
2: $k = 0$
3: $\mathcal{L}_k = \mathcal{L}$
4: $\mathcal{U}_k = \mathcal{U}$
5: **while** $k < N$ **do**
6: # train the network models G_k, D_k using $\mathcal{L}_k$, $\mathcal{U}_k$
7: D_k = train($\mathcal{H}$, $\mathcal{L}_k$)
8: **if** $k = 0$ **then**
9: G_k = train(D_k, $\mathcal{H}$, $\mathcal{U}_k$)
10: **else**
11: G_k = train(D_k, $\mathcal{H}$, $\mathcal{L}_k$)
12: **end if**
13: # Randomly select a pool of candidate data points $\mathcal{S}_k$ of
14: # size K
15: $\mathcal{S}_k \subseteq \mathcal{U}_k$; $|\mathcal{S}_k| = K$
16: **for** $x_i \in \mathcal{S}_k$ **do**
17: # compute adversarial attacks with network G_k
18: $\tilde{x}_i = G_k(x_i)$
19: **end for**
20: # query the labels of the top n_{query} samples with the smallest
21: # ℓ_2 norm perturbation
22: $\text{Index}_{(k)} \leftarrow argsort(\|x_i - \tilde{x}_i\|_2 | i = 1, ..., K)$
23: $Q_k \leftarrow \{(x_j, y_j), (\tilde{x}_j, \tilde{y}_j) | j \in \text{Index}_{(k)}[0 : n_{\text{query}}]\}$
24: $\mathcal{L}_{k+1} \leftarrow \mathcal{L}_k \bigcup Q_k$
25: $\mathcal{U}_{k+1} \leftarrow \mathcal{U}_k \setminus Q_k$
26: **end while**

uncertainty via labeling provides more knowledge about the decision boundary of the IDS model. As a result, the required inputs in Algorithm 1, the initialization, the general structure of the while-loop, and the step of query (lines 20 - 25) are the same as in DFAL. The main difference in Gen-AAL is the introduction of the network models G_k, D_k and the computation of adversarial attacks using G_k. Unlike the DFAL algorithm which uses a gradient-based DeepFool algorithm to compute the adversarial feature points against the S-IDS, our proposed algorithm uses the VAE model to generate the adversarial feature points. This is because a potential issue of gradient-based adversarial attack methods is that the perturbation vector is derived from a linearized model of the classifier at a particular feature point in the feature space [17] . In general, the decision boundary of a classifier for an IDS is nonlinear. Therefore, for a classifier with a highly nonlinear decision boundary, such way of calculating the perturbation may be subject to high model estimation error.

4 EXPERIMENTS

4.1 Dataset and Data Preprocessing

The network traffic flow features used in the experiment come from the CIC IDS 2017 Dataset [21]. This dataset contains benign traffic feature points and malicious traffic feature points from different types of malware attacks. We choose the malicious feature points from a particular type of attack named the botnet attack. The original data points in CIC IDS 2017 are represented by a mixture of numerical values and categorical values. Some examples of these features are: *standard deviation time between two packets sent in the flow*, *variance of packet length*, and *average number of bulk rate in the backward direction*. Using the preprocessing method presented in [4], we convert the original representation to a 78-dimensional feature vector. The values of the feature vectors are normalized such that each entry in a vector ranges from 0 to 1. Since the 78-dimensional vector representation is used by both the adversarial feature points from the GAN model and the input to the IDS model, we have by default, assumed that the attacker knows the features used in the IDS. In practice, such assumption may not hold for a black-box IDS. Therefore, as part of the future work, we plan to limit the adversarial perturbation to only a small subset of the 78-dimensional features in order to have a more realistic experiment setting.

4.2 Model Design and Training

4.2.1 VAE model. The VAE model consists of an encoder network and a decoder network, both of which are designed as a 5-layer fully connected network. For pretraining, we use the Adam optimizer to train the VAE model for 500 epochs with a learning rate of 0.0001 and a batch size of 128. In each iteration of GAN model retraining, the learning rate of the VAE model is changed to 0.0002 to cope with the reduced size of training dataset. The values of hyperparameters in VAE pretraining such as the learning rate and the number of epochs are empirically chosen to maximize the model performance, as is the case in choosing the hyperparameter values in GAN training for both the VAE and the S-IDS models. After pretraining, the mean-square reconstruction error after pretraining is reduced to less than 0.1.

4.2.2 IDS model. To show that our proposed method is not limited to attacking DNN-based IDS, the IDS model used in the experiment is chosen as a gradient boosted decision tree model. This IDS model is trained with the same training dataset used for VAE pretraining and achieves a classification accuracy of 99.7%. Instead of using the original labels in the CIC IDS 2017 dataset, the prediction result of the IDS model is used as the ground truth labels for all the feature points. This is because the objective of adversarial attack is to compromise the IDS model. Hence, the classification result of

the IDS model is used as the metric for performance evaluation of the GAN model.

4.2.3 S-IDS model. The S-IDS model is designed as a 4-layer fully-connected network. It is trained for 20 epochs using an Adam optimizer with a learning rate of 0.005 and a batch size of 32. The loss function is chosen as the cross-entropy loss. The S-IDS model is used in both the Gen-AAL attack model and the DFAL attack model. The structure of the DFAL attack model is similar to the one shown in Fig. 1, except that the VAE generator is replaced by the DeepFool algorithm to compute the adversarial feature point $\tilde{x}$. The two attack models use the same hyperparameter values for S-IDS training.

4.3 Generative Adversarial Active Learning

Out of the entire pool of training data, 15 data points are randomly selected to form the initial labeled dataset $\mathcal{L}_0$, while the remaining data points are used to create the initial unlabeled dataset $\mathcal{U}_0$. At each iteration of model retraining, the top 5 adversarial feature points with the smallest perturbation are selected for query, which makes the size of Q_k equal to 10 for $k \geqslant 1$ (5 adversarial feature points plus 5 original feature points). The adversarial attack success rate is used as the main metric for performance evaluation, while the ℓ_2-norm of the perturbation is also considered as an auxiliary metric. The state-of-the-art DFAL algorithm is used as a baseline algorithm for performance comparison. The maximum number of iterations for model retraining is set as $N = 3$. We run both our proposed algorithm (denoted as Gen-AAL) and DFAL 10 times and record the average performance. In each run, we use a different random seed for data sampling and model training. blueAs shown in Fig. 3, the proposed GAN model achieves a 98.86% success rate with only one iteration of model retraining by which 25 labels from the IDS has been used. In comparison, the success rate of the DFAL algorithm is 95.48% at the 1st iteration and eventually reaches 96.25% at the 3rd iteration by which 45 labels has been used. In terms of perturbation magnitude, the proposed Gen-AAL algorithm has an average ℓ_2-distance of 1.174 at the 1st iteration of model retraining, which is slightly higher than the average perturbation of 0.506 by the DFAL algorithm. Samples of the original feature points and the corresponding adversarial feature points are shown in Fig. 4. As shown in the figure, the adversarial feature points are close to the original feature points thanks to the regularization on the magnitude of perturbation. These experimental results show that our proposed method outperforms the state-of-the-art in the success rate of adversarial attacks against an black-box IDS model despite having a perturbation higher than DFAL.

Due to the highly limited number of training data points, our GAN model is prone to overfitting. As a result, the success rate of adversarial attack is sensitive to hyperparameters related to model training. In particular, when the GAN model learning rate and S-IDS model training epochs are too small, the attack success rate will have a large fluctuation in the first few iterations of model retraining. Increasing the number of epochs for GAN training helps to reduce this fluctuation but can also slightly reduce the attack success rate if the number of epochs for GAN training is higher than the optimal value.. A comparison of adversarial attack success rates with different choices of hyperparameter values is shown in

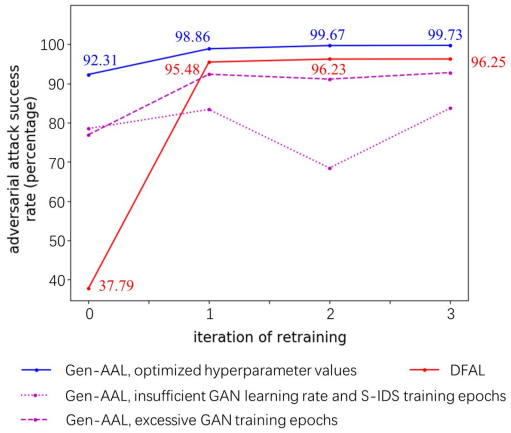

Figure 3: Adversarial attack success rate vs. iteration of model retraining, where the rate values of Gen-AAL with optimal tuning and DFAL are shown.

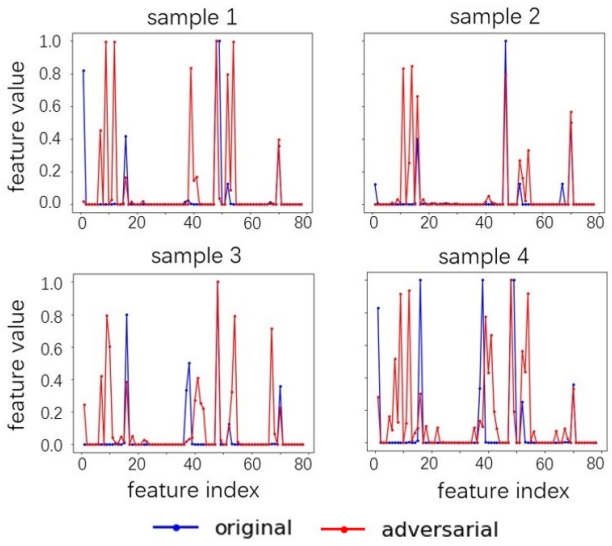

Figure 4: Samples of the original feature points and the perturbed feature points after active learning.

Fig. 3. To achieve a high success rate in adversarial attacks, it is important to attain a balance between the trainings of the VAE model and the S-IDS model through hyperparameter tuning. In theory, the maximal iteration of model retraining can be as large as the user wants until all the data points in the unlabeled data pool are labeled by the IDS. In practice, since an attacker tends to minimize the number of times the IDS is queried out of the concern for stealthiness, the model retraining can be stopped early once a satisfying attack success rate is achieved. Therefore, in our experiments, the maximal iteration of model retraining in active learning is set as 3, while a satisfying attack success rate of 98.86% is already obtained in the first iteration.

5 CONCLUSION AND FUTURE WORK

In this paper, an adversarial attack method is developed to generate adversarial examples against a black-box IDS model. In the proposed method, the adversarial feature points generated by the VAE model are not only used to attack the black-box IDS model, but also used to sample unlabeled feature points to query the IDS model. By showing this possible way of using ML algorithms against network intrusion detection, we intend to reveal the potential threat of ML-aided malware and the necessity for developing the corresponding countermeasures. Experimental results show that such dual usage of VAE generator improves the effectiveness and efficiency of an adversarial attack.

In the proposed method for adversarial attack, both the GAN model and the IDS model operate in some predefined feature space. In a practical malware intrusion, the feature space defined in the GAN model may not be the same as the feature space defined in the IDS model. Hence, the actual performance of the proposed adversarial attack method depends on the choice of features for the GAN model and how precisely the malicious network traffic can be reshaped to meet the computed values of the adversarial features. Compared with wired networks, wireless networks are facing special challenges in network traffic feature extraction due to the limited visibility of the nodes collecting audit data and the unstable signal strength for traffic-based collection sensors [16]. These challenges may increase the error in malicious traffic reshaping. As a result, it may take more labels from the black-box IDS and more iterations of model retraining to launch a successful adversarial attack. In terms of computation, the trained generator model (VAE) is only 58.2 KB when saved as a Python dictionary object file, and therefore should be small enough to load in a small scale mobile device such as a smart phone in a wireless network.

One direction of future work is to restrain the adversarial perturbations to only the non-functional features, which are the features whose change of value do not affect the functionality of the malware according to the attack principles and purposes. Another direction of future work is to implement the algorithm in an actual mobile network platform in order to evaluate the attack performance in a real-world scenario. Yet another future direction of our work is to consider both reactive and proactive defensive strategies to adversarial examples; such as defensive distillation [18], gradient masking [24] and adversarial training [5].

ACKNOWLEDGMENTS

This work is funded by the ARL CRA: MACRO: Models for Enabling Continuous Reconfigurability of Secure Missions grant W911NF-13-20045. Any opinions, findings, or conclusions found in this paper are those of the authors and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] Michael Atighetchi, Partha Pal, Franklin Webber, and Christopher Jones. 2003. Adaptive use of network-centric mechanisms in cyber-defense. In *Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 2003.* IEEE, 183–192.

[2] Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841* (2018).

[3] Tugba Erpek, Yalin E Sagduyu, and Yi Shi. 2018. Deep learning for launching and mitigating wireless jamming attacks. *IEEE Transactions on Cognitive Communications and Networking* 5, 1 (2018), 2–14.

[4] Osama Faker and Erdogan Dogdu. 2019. Intrusion detection using big data and deep learning techniques. In *Proceedings of the 2019 ACM Southeast Conference.* 86–93.

[5] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[6] Govind P Gupta and Manish Kulariya. 2016. A framework for fast and efficient cyber security network intrusion detection using apache spark. *Procedia Computer Science* 93 (2016), 824–831.

[7] Xuxian Jiang. 2011. Security alert: New stealthy android spyware-plankton-found in official android market. *Department of Computer Science, North Carolina State University, URL: http://www. csc. ncsu. edu/faculty/jiang/Plankton* (2011).

[8] Bill Karakostas. 2016. Event Prediction in an IoT Environment Using Naïve Bayesian Models.. In *ANT/SEIT*. 11–17.

[9] Sydney Mambwe Kasongo and Yanxia Sun. 2019. A deep learning method with filter based feature engineering for wireless intrusion detection system. *IEEE Access* 7 (2019), 38597–38607.

[10] Jin-Young Kim, Seok-Jun Bu, and Sung-Bae Cho. 2017. Malware detection using deep transferred generative adversarial networks. In *International Conference on Neural Information Processing*. Springer, 556–564.

[11] N. O. Leslie. 2018. Using semi-supervised learning for flow-based network intrusion detection. In *Proceedings of the 23rd International Command and Control Research and Technology Symposium (ICCRTS): Multi-Domain C2* (Pensacola, FL: ICCRTS).

[12] David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*. Springer, 3–12.

[13] Zilong Lin, Yong Shi, and Zhi Xue. 2018. Idsgan: Generative adversarial networks for attack generation against intrusion detection. *arXiv preprint arXiv:1809.02077* (2018).

[14] Nuno Martins, José Magalhães Cruz, Tiago Cruz, and Pedro Henriques Abreu. 2020. Adversarial Machine Learning applied to Intrusion and Malware Scenarios: a systematic review. *IEEE Access* 8 (2020), 35403–35419.

[15] Jelena Milosevic, Francesco Regazzoni, and Miroslaw Malek. 2017. Malware threats and solutions for trustworthy mobile systems design. In *Hardware Security and Trust*. Springer, 149–167.

[16] Robert Mitchell and Ray Chen. 2014. A survey of intrusion detection in wireless network applications. *Computer Communications* 42 (2014), 1–23.

[17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2574–2582.

[18] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 582–597.

[19] MR Gauthama Raman, Nivethitha Somu, Kannan Kirthivasan, and VS Shankar Sriram. 2017. A hypergraph and arithmetic residue-based probabilistic neural network for classification in intrusion detection systems. *Neural Networks* 92 (2017), 89–97.

[20] Maria Rigaki and Sebastian Garcia. 2018. Bringing a gan to a knife-fight: Adapting malware communication to avoid detection. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 70–75.

[21] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*. 108–116.

[22] Yi Shi, Yalin E Sagduyu, Kemal Davaslioglu, and Jason H Li. 2018. Generative adversarial networks for black-box API attacks with limited training data. In *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. IEEE, 453–458.

[23] Nasrin Sultana, Naveen Chilamkurti, Wei Peng, and Rabei Alhadad. 2019. Survey on SDN based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications* 12, 2 (2019), 493–501.

[24] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* (2017).

[25] Huiwen Wang, Jie Gu, and Shanshan Wang. 2017. An effective intrusion detection framework based on SVM with feature augmentation. *Knowledge-Based Systems* 136 (2017), 130–139.

[26] Di Wu, Binxing Fang, Junnan Wang, Qixu Liu, and Xiang Cui. 2019. Evading Machine Learning Botnet Detection Models via Deep Reinforcement Learning. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 1–6.

[27] Renxian Zhang, Dehong Gao, and Wenjie Li. 2012. Towards scalable speech act recognition in twitter: tackling insufficient training data. In *Proceedings of the Workshop on Semantic Analysis in Social Media*. Association for Computational Linguistics, 18–27.

[28] Yajin Zhou and Xuxian Jiang. 2012. Dissecting android malware: Characterization and evolution. In *2012 IEEE symposium on security and privacy*. IEEE, 95–109.