An Adversarial Learning Model for Intrusion Detection in Real Complex Network Environments

Ying Zhong[1], Yiran Zhu[2], Zhiliang Wang[1,4](✉), Xia Yin[3,4], Xingang Shi[1,4], and Keqin Li[5]

[1] Institute for Network Sciences and Cyberspace at Tsinghua University, Beijing, China
zhongy18@mails.tsinghua.edu.cn, wzl@cernet.edu.cn
[2] Beijing Normal University, Beijing, China
[3] Department of Computer Science and Technology at Tsinghua University, Beijing, China
[4] Beijing National Research Center for Information Science and Technology, Beijing, China
[5] Department of Computer Science, State University of New York, New Paltz, USA

Abstract. Network intrusion detection plays an important role in network security. With the deepening of machine learning research, especially the generative adversarial networks (GAN) proposal, the stability of the anomaly detector is put forward for higher requirements. The main focus of this paper is on the security of machine learning based anomaly detectors. In order to detect the robustness of the existing advanced anomaly detection algorithm, we propose an anomaly detector attack framework MACGAN (maintain attack features based on the generative adversarial networks). The MACGAN framework consists of two parts. The first part is used to analyze the attack fields manually. Then, the learning function of GAN in the second part is used to bypass the anomaly detection. Our framework is tested on the latest Kitsune2018 and CICIDS2017 data sets. Experimental results demonstrate the ability to bypass the state-of-the-art machine learning algorithms. This greatly helps the network security researchers to improve the stability of the detector.

Keywords: Generative adversarial networks · Traffic anomaly detection · Machine learning security

1 Introduction

Intrusion detection system (IDS) is a key component in securing computing infrastructure. The purpose of this component is to prevent violations of the defense mechanism. In fact, IDS itself is part of the computing infrastructure, so they may also be attacked by adversaries [1]. In [2], skilled attacker can use ambiguity in the traffic flow seen by the monitor to evade detection. Many evasion techniques are proposed in [3], but they are all limited to botnet traffic evasion. And, many of the anomaly detection algorithms used for experiments are now outdated.

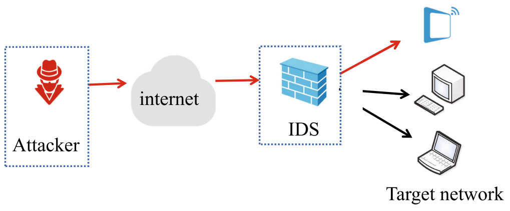


Fig. 1. Network attacks in the presence of IDS.

Because of the development of machine learning, the accuracy of network traffic anomaly detection algorithms continues to increase, but the stability of the algorithm itself faces enormous challenges [4–6]. Attackers can carefully design adversarial samples that are not too different from normal samples, and make machine learning algorithms make completely different decisions. Aiming at the lack of research on adversarial learning in the existing network IDS, this paper proposes adversarial sample generation algorithms to reveal potential security problems in the existing IDS. In order to explore the robustness of machine learning-based IDS, as shown in Fig. 1, we will attack wireless devices in the presence of IDS.

In summary, we propose a new attack method based on GAN. The reason for choosing GAN is that we need the characteristics of GAN to generate adversarial samples [25]. The generator of GAN modifies some of the fields that can be disturbed so that its data features are close to those of benign data packets. Therefore, the adversarial sample can deceive the anomaly detector well while maintaining the attack features of the traffic. We call it the MACGAN model. To the best of our knowledge, this paper is the first work to bypass network intrusion detection in real network traffic.

The main contributions of this paper are summarized below.

- For the network IDS, we design the MACGAN attack model. This model can bypass anomaly detectors by modifying meaningless fields. The main reason is that we add the target anomaly detector to the model so that it can be fitted by the discriminator of GAN. We attack before the anomaly detector extracts traffic features, which has practical application significance.
- We propose to divide the fields of network packets into perturbable and non-perturbable parts. We perturb the fields that can be perturbed so that the current network packets will not be detected by the anomaly detector after the perturbation.

- We design a series of experiments for the MACGAN model. We first explore the effects of parameters in GAN on the experiment, and then test the impact of the number of iterations on the attack effect. Finally, experiments on the latest Kitsune2018 and CICIDS2017 datasets prove the effectiveness of our attack model. In order to further explain the practicality of the attack, we attack other classification algorithms, and the attack effect is significant.

The rest of this paper is organized as follows. Section 2 presents the related work. In Sect. 3, we design a MACGAN attack model, and provides a detailed description about how to bypass traffic anomaly detector. Performance evaluation is in Sect. 4. Section 5 concludes the paper.

2 Related Work

We analyze the application of the adversarial samples in anomaly detection [1]. [7] investigated the performances of the state-of-the-art attack algorithms against deep learning-based intrusion detection on the NSL-KDD data set. The roles of individual features in generating adversarial examples were explored. [8] showed that by modifying on average as little as 1.38 of the input features, an adversary could generate malicious inputs which effectively fooled a deep learning based NIDS. Therefore, when designing such systems, it was crucial to consider the performance from not only the conventional network security perspective but also the adversarial machine learning domain. [9] presented an approach to generate explanations for incorrect classifications made by the data-driven IDS. An adversarial approach was used to find the minimum modifications (of the input features) required to correctly classify a given set of misclassified samples. The magnitude of such modifications was used to visualize the most relevant features that could explain the reason for the misclassification. [10] proposed the use of GANs for generating network traffic in order to mimic other types of traffic. In particular, they modified the network behavior of a real malware in order to mimic the traffic of a legitimate application, and therefore avoided detection. [11] investigated how adversarial examples affect the performance of deep neural network (DNN) trained to detect abnormal behaviors in the black-box model. They demonstrated that adversary could generate effective adversarial examples against DNN classifier trained for NIDS even when the internal information of the target model was isolated from the adversary. In [12], a framework of the GAN, IDSGAN, was proposed to generate the adversarial attacks, which could deceive and evade the IDS. The internal structure of the detection system was unknown to attackers, thus adversarial attack examples performed the black-box attacks against the detection system.

Among the adversarial sample generation algorithms mentioned in the above literature, some of them were not used for traffic anomaly detection, but were used for other aspects [10]. Some algorithms used data sets that were too ideal and not representative, such as NSLKDD [7,9,12]. The others could completely bypass the anomaly detector, but the corresponding samples had lost the features of the attack [8,9,12]. Therefore, we need to design an algorithm that can attack the latest cyber attack data without losing the significance of the sample itself.

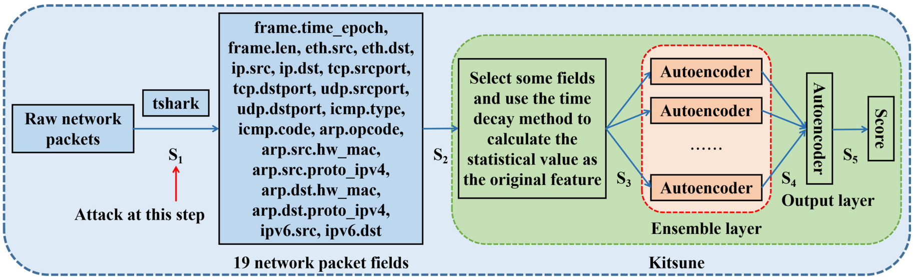

Raw network packets
tshark
S1
Attack at this step
frame.time_epoch, frame.len, eth.src, eth.dst, ip.src, ip.dst, tcp.srcport, tcp.dstport, udp.srcport, udp.dstport, icmp.type, icmp.code, arp.opcode, arp.src.hw_mac, arp.src.proto_ipv4, arp.dst.hw_mac, arp.dst.proto_ipv4, ipv6.src, ipv6.dst
19 network packet fields
S2
Select some fields and use the time decay method to calculate the statistical value as the original feature
S3
Autoencoder
Autoencoder
……
Autoencoder
Ensemble layer
S4
Autoencoder
Output layer
S5
Score
Kitsune

Fig. 2. Internal structure of Kitsune [1].

3 MACGAN Model: An Adversarial Learning Model for Intrusion Detection

The existing mainstream network traffic anomaly detection algorithms are based on machine learning methods. These methods require extraction of features. The establishment of these features is inseparable from the choice of fields. Our general idea is to first ensure that the fields necessary for the attack cannot be modified. For the remaining fields called non-attack field, we bypass the anomaly detector through the sample generation function of GAN.

3.1 Analysis of Advanced Anomaly Detection Algorithm

Figure 2 depicts the advanced anomaly detection algorithm Kitsune. In order to make the attack algorithm more versatile, we put the attack step on S_1, because many anomaly detectors are modeled based on network packet fields. Thus, if the abnormal detection algorithm are based on field modeling, our attack mode also applicable. The details of the specific Kitsune algorithm can be found in [13].

3.2 MACGAN Attack Model

There are many versions of GANs, which are selected according to different design requirements. In order to prevent the non-convergence and instability of GAN, we design MACGAN based on the Wasserstein GAN structure [14]. It is an improved model of GAN, for the evasion attacks against IDS. The framework of MACGAN is described in Fig. 3, where the noun of the input and output data represents the sample set, and the letter represents each sample. For example, in

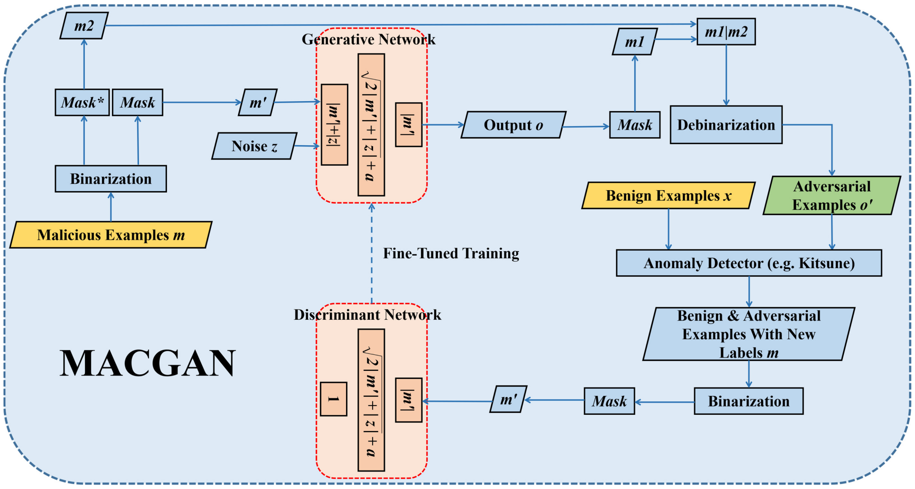


Fig. 3. The framework of MACGAN.

"Malicious Examples m", "Malicious Examples" represents a malicious data set, and "m" represents each malicious sample. Algorithm 1 illustrates the training process of MACGAN.

Generation Module: The generation network G is a three-layer feed forward neural network. The malicious sample m (traffic packet labeled abnormal) is first binarized, and then the non-attack field is reserved by $mask$ to obtain m'. m' is connected to the noise vector z as an input of G. The input layer size of the generation network is $|m'| + |z|$, and hidden layer size is $\sqrt{2|m'| + |z|} + a$, a is the adjustment constant between [1,9], and the output layer size is $|m'|$. The output of the generated network is o. To ensure the enforceability and aggression against the sample, o and $mask$ are operated together, and only the generation data of the non-attack field is retained, and $m1$ is obtained. At the same time, $mask$ is reversed to get $mask*$, and all the bits except the non-attack field can be reserved by $mask*$. The original malicious data m is done with $mask*$ and gets $m2$. After $m1$ and $m2$ are superimposed, they are debinarized to obtain a new sample o'.

Discriminating Module: The discriminating module is composed of an abnormality detector K (Kitsune) that is expected to be deceived and a discriminating network D. The benign sample x and the new sample o' retain the binary vector of the non-attack field as the input of D. Before the input, the two samples need to be discriminated by the anomaly detector, and the sample attributes are re-marked according to the discriminating result (benign or malicious). The input layer size of the discriminant network is $|m'|$, the hidden layer size is $\sqrt{2|m'| + |z|} + a$, a is the adjustment constant between [1,9], and the output layer size is 1. The sample data after updating the label passes through the discriminant network, and the discriminating result is output. Then, the loss function is calculated based on the discriminating result. When the input sample is o', if K discriminates o' as a malicious sample, that is, $K(o') = 0$, then $D(o')$ is made as 0 as possible; if K discriminates o' as a benign sample, That is, $K(o') = 1$, so that $D(o')$ tends to be as large as 1. When the input sample is x, if K discriminates x as a benign sample, that is, $K(x) = 1$, $D(x)$ is made as close as possible to 1. After the loss is calculated, the gradient backward propagation is performed, and the parameters of the discriminant model are updated, so that the degree of fitting of D to K is continuously improved.

Algorithm 1 The training process of MACGAN

```
Input:
    Malicious traffic examples $M$, the vector $mask$ for protecting the non-attack field, benign traffic examples $X$, the learning rate $\alpha$, the clipping parameter $c$, the batch size $n$, the number of iterations of the critic per generator iteration $n_{critic}$, the noise $Z$ for the adversarial generation, initial discriminator parameters $\omega_0$, initial generator parameters $\theta_0$.
Output:
    The trained generator $g_\theta$ and the trained discriminator $f_\omega$.
1: Step 1: Merging 19 fields (after the tag) in Kitsune with other perturbed fields (such as optional fields in the network packet) to form a malicious traffic.
2: Step 2: Vectorize malicious samples and add perturbations to marked fields.
3: //Step 3
4: while $\theta$ has not converged do
5:     while $n_{critic}$-- do
6:         Sample $\{M^{(i)}\}_{i=1}^{n} \sim P_r$ a batch from the malicious traffic examples.
7:         Sample $\{X^{(i)}\}_{i=1}^{n} \sim P_g$ a batch from the benign traffic examples.
8:         Sample $\{Z^{(i)}\}_{i=1}^{n} \sim P_z$ a batch of prior samples.
9:         $f_\omega \leftarrow \nabla_\omega[\frac{1}{n}\sum_{i=1}^{n} f_\omega(X^{(i)} * mask) - \frac{1}{n}\sum_{i=1}^{n} f_\omega(g_\theta(M^{(i)} * mask + Z^{(i)}))]$ //* - Hadamard Product
10:        $\omega \leftarrow \omega + \alpha \cdot RMSProp(\omega, g_\omega)$ //$RMSProp$ - An optimization algorithm: Root Mean Square Prop
11:        $\omega \leftarrow clip(\omega, -c, c)$
12:    end while
13:    $g_\theta \leftarrow -\nabla_\theta \frac{1}{n}\sum_{i=1}^{n} f_\omega(g_\theta(M^{(i)} * mask + Z^{(i)}))$
14:    $\theta \leftarrow \theta - \alpha \cdot RMSProp(\theta, g_\theta)$
15: end while
16: Step 4: Inverse vectorization, the meaningless fields after the disturbance are mapped to meaningful fields by hashing.
17: return $g_\theta$, $f_\omega$.
```

With the increase of the number of trainings, the discriminative ability of D tends to be consistent with the anomaly detector. At the same time, the ability to generate forged samples is continuously strengthened. Finally, D cannot effectively distinguish the generated adversarial examples and the original real samples generated by G. The adversarial examples generated at this time are not only discriminated by the anomaly detector as benign classification, but also use the mask to preserve the aggressiveness of the sample, and realize the deception of the anomaly detector.

4 Experiments and Evaluation

This section covers our experimental results. Our codes are available at the open-source code repository[1]. In order to systematically evaluate our method, we want to check the following two points: (1) How about our attack model's performance based on Kitsune anomaly detection algorithm. (2) How about our attack model's performance based on multiple anomaly detection algorithms.

4.1 Metrics for Evaluating Anomaly Detection Algorithm

We use the following metrics to evaluate the effectiveness of our MACGAN model. Attack Effect Rate $(AER) 1 - \frac{TPR_{\text{After\_attack}}}{TPR_{\text{Before\_attack}}}$: It measures the extent to which an attack method reduces the accuracy of anomaly detection. True Positive Rate $(TPR) \frac{TP}{TP+FN}$: It measures the proportion of outliers that are correctly identified. Among them, True Negative (TN): a measure of the number of normal events rightly classified normal. True Positive (TP): a measure of the number of abnormal events rightly classified abnormal. False Positive (FP): a measure of normal events misclassified as attacks. False Negative (FN): a measure of attacks misclassified as normal. Higher AER means better attack performance and higher TPR means better detection performance.

4.2 Datasets and Experimental Settings

Datasets. We investigate DARPA, KDD99, NSL-KDD, UNSW_NB and other data sets, and find that some of these data sets are not real traffic, and some are real traffic but are outdated and cannot represent changing attack behavior. Thus, two data sets, Kitsune2018[2] [13] and CICIDS2017[3], are used in this paper to evaluate the performance of our scheme.

The Kitsune dataset comes from two parts. The first part is to attack in a real IP camera video surveillance network. For example, an attack can affect the availability and integrity of the video uplink. In order to establish a more noisy network attack environment, the second part of the attack data comes from the attack environment of 9 IoT devices and 3 PCs, including wireless network equipment. There are 9 data sets used to evaluate Kitsune: OS Scan, Fuzzing, Video Injection, ARP MitM, Active Wiretap, SSDP Flood, SYN DoS, SSL Renegotiation and Mirai. See [13] for detailed description.

CICIDS2017 contains benign and the most up-to-date common attacks, which resembles the true real-world data (PCAPs). It also includes the results

[1] https://github.com/zyyrrr/MACGAN.
[2] https://goo.gl/iShM7E.
[3] https://www.unb.ca/cic/datasets/ids-2017.html.

of the network traffic analysis using CICFlowMeter with labeled flows based on the time stamp, source and destination IPs, source and destination ports, protocols and attack (CSV files). The implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. We use the original label from CICIDS2017.

Experiment Environment. In the experiment, PyTorch [15] is adopted as the deep learning framework to implement MACGAN. The purposed model is run and evaluated on a Linux PC with Intel Core i7-5500 and 1080Ti GPU.

Table 1. Identify data sets that require further attack. TPR measures the proportion of abnormaly that are correctly identified. 926554.8557720063 is expressed in the form of 9e5. *no* represents a range of dataset sizes we choose that do not require further attack anomaly detection algorithms. Because TPR itself is very low. *yes* stands for the need to use the MACGAN algorithm for further attacks.

Attack Name	Benign + Malicious	train + test	Threshold	TPR	Need attack
OS Scan	65845 + 34155	14000 + 86000	12.194	0.441	No
Fuzzing	77206 + 22794	50000 + 50000	0.348	0.004	No
Video Injection	80395 + 19605	40000 + 60000	1.713	0.002	No
ARP MitM	61995 + 38005	50000 + 50000	9e5	0.001	No
Active Wiretap	62285 + 37715	50000 + 50000	4e10	$5e-05$	No
SSDP Flood	62295 + 37705	50000 + 50000	27.571	0.994	Yes
SYN Flood	92962 + 7038	10000 + 90000	1e3	0.004	No
SSL Renegotiation	89468 + 10532	40000 + 60000	33.151	0.005	No
Mirai	69999 + 30001	55000 + 45000	1.595	0.818	Yes
CICIDS2017	71860 + 28140	35000 + 65000	4.958	0.659	Yes

Table 2. Analysis of fields that can be disturbed based on three data sets. The numbers here are the numbers of the 19 network packet fields in Fig. 2.

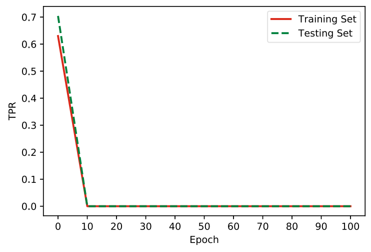
Data set	Non-attack field number	Reserved	Modified
Mirai	0, 2–5, 14, 16–18	0, 3 ,5, 7, 9, 18	2, 4, 14, 16, 17
SSDP Flood	1–7, 9, 14, 16–18	0, 3, 5, 7, 9, 11, 13–16, 18	1, 2, 4, 6, 17
CICIDS2017	1–5, 14, 16–18	0, 1, 3, 5, 7, 9, 11, 18	2, 4, 14, 16, 17

4.3 Experimental Results and Implications

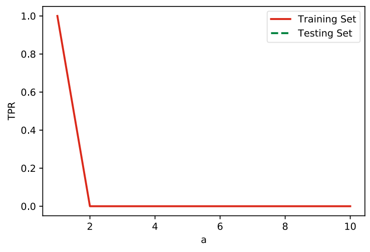
Attack Based on Kitsune Anomaly Detection Algorithm. This section illustrates a group of experiments to verify the effectiveness of our MACGAN attack model. We first sample the original data set of this algorithm. There are 100,000 benign samples along with malicious samples. The current TPR is tested in the case of $FPR = 0.001$. We decide whether to further use our attack algorithm based on the value of TPR. Table 1 describes which data sets can be used for further attacks. Among them, TPR of Kitsune on the SSDP data set can reach 0.994. The training set and test set size are both 50000. In order to ensure the robustness of our attack algorithm, we also use the CICIDS2017 data set to conduct experiments.

According to Table 1, we further attack Kitsune on the three data sets Mirai, SSDP Flood, and CICIDS2017 when we specify the size of the training sample. In order to maintain the aggressiveness of these attack methods, we not only retain the fields that these attacks must retain. The details are shown in Table 2.

From the previous section, we have been able to determine which fields can be attacked. Figure 4 shows the experiment on the CICIDS2017 data set. It can be seen from Fig. 4(a) that when the algorithm is iterated to the tenth times, Kitsune's TPR is already zero, which proves the effectiveness of our attacks. At the same time, in order to analyze the influence of the hidden layer parameter a on the convergence of the algorithm, we supplemented the experiment. As shown in Fig. 4(b), when a is greater than 2, the attack effect can be best. Figure 5 and 6 are similar to the attack effect in Fig. 4. The difference is that the value of convergence point a of Mirai data set in Fig. 5 is 3 and 9, and the value of convergence point a of SSDP Flood data set in Fig. 6 is greater than 6.

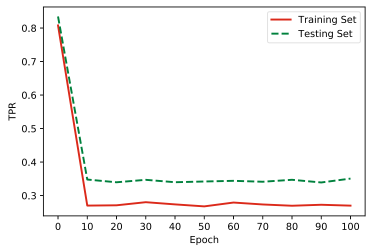


(a) CICIDS2017_TPR.

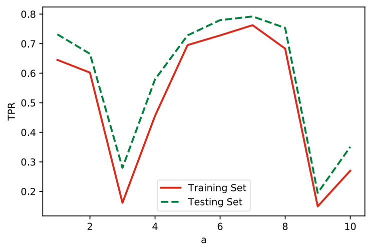


(b) CICIDS2017_TPR_a.

Fig. 4. Changes in TPR under the Mirai data set.

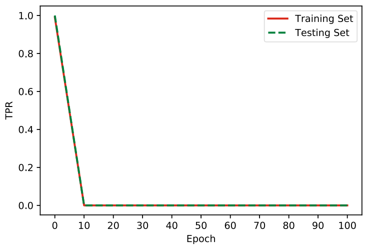


(a) Mirai_TPR.

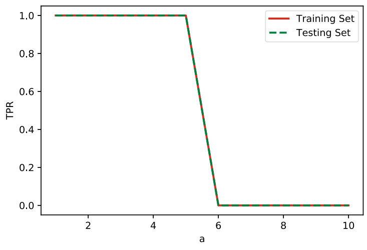


(b) Mirai_TPR_a.

Fig. 5. Changes in TPR under the SSDP_Flood data set.



(a) SSDP_Flood_TPR.



(b) SSDP_Flood_TPR_a.

Fig. 6. Changes in TPR under the SSDP_Flood data set.

Table 3. AER under different parameters a. The larger AER, the better the attack effect. TPR_{before} is abbreviated as T_b. Negative number indicates a side effect of the attack. Hidden layer size is $\sqrt{2|m'| + |z|} + a$.

a	Mirai (T_b=0.818)	SSDP Flood (T_b=0.994)	CICIDS2017 (T_b=0.659)
$a = 1$	0.106	−0.006	−0.517
$a = 2$	0.187	−0.006	1
$a = 3$	0.658	−0.006	1
$a = 4$	0.293	−0.006	1
$a = 5$	0.111	−0.006	1
$a = 6$	0.047	1	1
$a = 7$	0.968	1	1
$a = 8$	0.079	1	1
$a = 9$	0.762	1	1
$a = 10$	0.572	1	1

Next, we introduce the concept of AER. As shown in Table 3, this is an assessment of the attack effect under different parameters a of different data sets. We can see that when a is 7, our attack effect can be best on different data sets. In short, as long as we can satisfy the detection part of the anomaly detector and change the undiscovered part by disturbance, our attack effect will be better. For example, the increase in the number of network addresses and the spread of timestamps will reduce the likelihood that a real attacker will be discovered.

Attacks Based on Multiple Anomaly Detection Algorithms. In order to further verify the effectiveness of our attack, we select another 300,000 data packets from the CICIDS2017 dataset for experiments. We use Isolation Forests (IF) [16] and Gaussian Mixture Models (GMM) [17]. IF is an ensemble based method of outlier detection, and GMM is a statistical method based on the expectation maximization algorithm. Then we use support vector machine (SVM) from [18], sparse autoencoder finetuned neural network (SAE) from [19], restricted boltzmann machine fine-tuned neural network (RBM) from [1] and kitsune from [20]. All classifiers use Kitsune's feature extraction method. We can see from Fig. 7 that the Kitsune algorithm has the highest TPR before being attacked, which can reach 0.998. But after being attacked, the detection effect is greatly reduced, and TPR is almost reduced to 0. Other algorithms have the same trend.

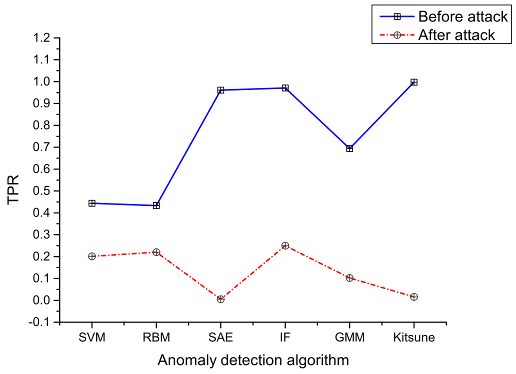


Fig. 7. Attack effects of different algorithms based on the 300,000 CICIDS2017 dataset.

5 Conclusion

The development of machine learning has facilitated network anomaly detection. However, attacks on machine learning methods are also needed to consider. This paper proposes an anomaly detector attack framework MACGAN based on GAN. Our attack effect is better, indicating that the robustness of the machine based anomaly detector needs to be further improved. Inspired by the works [21–24], we will defense our MACGAN attack model in our future work. For example, we can design a defense GAN to let it play a dynamic game with MACGAN.

Acknowledgment. This work is supported by the National Key Research and Development Program of China under Grant No. 2018YFB1800204.

References

1. Corona, I., Giacinto, G., Roli, F.: Adversarial attacks against intrusion detection systems: taxonomy, solutions and open issues. Inf. Sci. **239**, 201–225 (2013)
2. Handley, M., Paxson, V., Kreibich, C.: Network intrusion detection: evasion, traffic normalization, and end-to-end protocol semantics. In: USENIX Security Symposium (2001)

3. Stinson, E., Mitchell, J.C.: Towards systematic evaluation of the evadability of bot/botnet detection methods. In: WOOT (2008)
4. Barreno, M., Nelson, B., Joseph, A.D., Tygar, J.D.: The security of machine learning. Mach. Learn. **81**(2), 121–148 (2010). https://doi.org/10.1007/s10994-010-5188-5
5. Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D.: Can machine learning be secure?. In: AsiaCCS (2006)
6. Moosavi-Dezfooli, S.-M., Fawzi, A., Frossard, P.: DeepFool: a simple and accurate method to fool deep neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2574–2582 (2016)
7. Wang, Z.: Deep learning-based intrusion detection with adversaries. IEEE Access **6**, 38367–38384. IEEE (2018)
8. Clements, J.H., Yang, Y., Sharma, A., Hu, H., Lao, Y.: Rallying Adversarial Techniques against Deep Learning for Network Security. CoRR, vol. abs/1903.11688 (2019)
9. Marino, D.L., Wickramasinghe, C.S., Manic, M.: An adversarial approach for explainable AI in intrusion detection systems. In: IECON 2018–44th Annual Conference of the IEEE Industrial Electronics Society, pp. 3237–3243 (2018)
10. Rigaki, M., Garcia, S.: Bringing a GAN to a knife-fight: adapting malware communication to avoid detection. In: IEEE Security and Privacy Workshops (SPW), pp. 70–75. IEEE (2018)
11. Yang, K., Liu, J., Zhang, V.C., Fang, Y.: Adversarial examples against the deep learning based network intrusion detection systems. In: IEEE Military Communications Conference (MILCOM), pp. 559–564. IEEE (2018)
12. Lin, Z., Shi, Y., Xue, Z.: IDSGAN: Generative adversarial networks for attack generation against intrusion detection. arXiv preprint arXiv:1809.02077 (2018)
13. Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A.: Kitsune: an ensemble of autoencoders for online network intrusion detection. In: NDSS (2018)
14. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International Conference on Machine Learning, pp. 214–223 (2017)
15. Paszke, A., Gross, S., Chintala, S., Chanan, G.: Automatic differentiation in pytorch. In: NIPS 2017 Autodiff Workshop: The Future of Gradient-based Machine Learning Software and Techniques (2017)
16. Liu, F.T., Ting, K.M., Zhou, Z.-H.: Isolation forest. In: IEEE International Conference On Data Mining, pp. 413–422 (2008)
17. Reynolds, D.: Gaussian mixture models. Encyclopedia Biometrics, pp. 827–832 (2015)
18. Sahu, S.K., Jena, S.K.: A multiclass SVM classification approach for intrusion detection. In: Bjørner, N., Prasad, S., Parida, L. (eds.) ICDCIT 2016. LNCS, vol. 9581, pp. 175–181. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-28034-9_23
19. Yan, B., Han, G.: Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system. IEEE Access 41238–41248 (2018)
20. Fiore, U., Palmieri, F., Castiglione, A., Santis, A.D.: Network anomaly detection with the restricted Boltzmann machine. Neurocomputing 13–23 (2013)
21. Madani, P., Vlajic, N.: Robustness of deep autoencoder in intrusion detection under adversarial contamination. In: Proceedings of the 5th Annual Symposium and Bootcamp on Hot Topics in the Science of Security. ACM (2018)
22. Carlini, N., Wagner, D.A.: Towards evaluating the robustness of neural networks. In: IEEE Symposium on Security and Privacy (SP), pp. 39–57 (2017)

23. Ma, S., Liu, Y., Tao, G., Lee, W.C., Zhang, X.: NIC: detecting adversarial samples with neural network invariant checking. In: NDSS (2019)
24. Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE Symposium on Security and Privacy (SP), pp. 582–597. IEEE (2016)
25. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)