Network Intrusion Detection System based on Generative Adversarial Network for Attack Detection

Abhijit Das[1]
Research Scholar, Department of CSE
PESITM, Affiliated to VTU (Belagavi)
Shivamogga, Karnataka, India

Dr. S G Balakrishnan[2]
Associate Professor, Department of ISE
Mahendra Engineering College
Salem,India

Dr. Pramod[3]
Associate Professor, Department of ISE
PESITM, Affiliated to VTU (Belagavi)
Shivamogga, Karnataka, India

***Abstract*—The Intrusion Detection System (IDS) is the main element to prevent malicious traffic on the network. IDS will quickly increase the ability to detect network threats with the help of Deep Learning algorithms. As a result, attackers are finding new ways to evade identification. Polymorphic attacks, search for the attackers, as they can bypass the IDS. Generative Adversarial Networks (GAN) is a method proven in generating various forms of data. It is becoming popular among security researchers as it can produce indistinguishable data from the original data. This work proposed a model to generate DDoS attacks using a GAN. Several techniques have been used to regenerate the feature selection to identify the attack and generate polymorphic data. The data will change feature profile in every cycle to test if the IDS can detect the new version of attack data. Simulation results from the proposed model show that with constant changing attack profiles, defending arrangements that handle incremental knowledge will yet stay exposed to current attacks.**

***Keywords*—Generative adversarial networks; network threats; deep learning; intrusion detection; feature selections**

I. Introduction

The Internet is being used in many fields, like data transfer, e-learning, and many more, and its growth has impacted all aspects of life. This increasing usage of the Internet causes concerns about network security and needs constant improvements in securing Internet technologies from various attacks. Examples of these attacks include DDoS attacks, Man-in-the-middle attacks, Phishing, Password-based attack, SQL injection, and many more. Network vulnerabilities can cause damage to small or large organizations. According to one survey, 98% of businesses in the UK depend on Information Technology services. Over 43% of small scale and 72% of large-scale organizations suffered from cyber-attacks in the past years. There are many tools available to secure or prevent cyber-security attacks, including but not limited to: Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), Anti-malware, Network Access Control, Firewalls. Among those, one of the most commonly used and effective tool is the Intrusion Detection System.

IDS analyzes the data traffic is to be distinguished from the malignant and the normal traffic, and to generate a warning, so that all the necessary precautions must be taken to avoid damage [1]. With the development of network attacks and security, improved detection and prevention systems. Artificial intelligence (AI) is now widely used for security tools in the IDS [2],and activists have begun to use artificial intelligence techniques to malicious attacks [3] [4] . AI and deep learning algorithms require a large amount of data to train and test the models. Some of the techniques that can be used for the production of large data sets to finish the malware detection [5] [6] and security orchestration, [7].

One of the frameworks to generate adversarial data is Generative Adversarial Networks (GAN). It is an architecture of two neural networks: the Generator and the Discriminator. The Generator uses gradient descent or the response from the discriminator and generates adversarial data. The discriminator distinguishes between the original and the adversarial data. The Generator and the discriminator compete in this way, and, in the end, the Generator produces synthetic or adversarial data [8].GAN has been utilized in research to generate various types of datasets like images [9], sound [10], text [11], and network attack data [12].

II. Related Works

The recent development in deep learning, intrusion detection systems are getting advanced with these methods. However, there is limited research testing the integrity of the advanced IDS against adversarial data.

According to a study by [13], the authors created a framework that generates adversarial malware using GAN to bypass the detection system. The objective of this research is to use a black-box malware detector because most of the attackers are unaware of the detection techniques used in the detection system. Instead of directly attacking the black-box detector, researchers created a model that can observe the target system with corresponding data. Then this model calculates the gradient computation from the GAN to create adversarial malware data. With this technique, the authors received a model accuracy of around 98%.

This section covers some previous works on generating adversarial attack data using the Wasserstein GAN. The Wasserstein GAN model was introduced in [14], and it improves upon the traditional GAN. Wasserstein GAN is an extension of traditional GAN that finds an alternate method of training the Generator. In WGAN the Discriminator provides a critic score that depicts how real or fake the data generated.

To generate a malicious file [12] proposed a method that uses WGAN so that a detection system signifies the adversarial malicious file as a regular file. They have achieved an accuracy of around 99%, proving that their method can generate adversarial malicious files that can bypass the detection system.

A recent study in [15] uses Wasserstein GAN to generate simulated attack data. According to the authors, many tools can generate simulated attack data. However, this process could take a long time and a lot of resources. Using the proposed technique, they have produced millions of connection records with just one device and within a short period. They used the KDD Cup 1999 dataset as the training set. Their experiment suggests that as compared to GAN, the Wasserstein GAN learns faster and generates better results. A paper published by Ring et al. [16] proposed a method that produces flow-based attack data using Wasserstein GAN. This research uses the CIDDS dataset to test and train the proposed method. They have suggested that the flow-based dataset consists of categorical features like IP address, port numbers, etc. The GAN is unable to process categorical data. They have also proposed a method to preprocess the categorical data and transform them into continuous data. Lastly, they have used several techniques to evaluate the quality standard of the adversarial data. Results suggest that it is possible to generate real network data using this method.

A recently published paper by Lin et al. [17] discussed the benefits of WGAN. It proposed a technique IDSGAN to generate adversarial attack data and test the attack against the Intrusion Detection System. They have utilized the NSL-KDD as the benchmark dataset to generate an adversarial attack on an IDS. They have tested this technique with various machine learning classifiers like Support Vector Machine, Naïve Bayes, Multilayer Perceptron, Linear Regression, Decision Tree, Random Forrest. They have used four types of attacks, Probe, DoS, User to Root, Root to Local to generate adversarial attack data.

This research aims to create a framework that detect attacks using GAN, motivated by [17].

- This work begins with the important feature selection method using SHAP. This work identified the most critical features from the dataset that contribute to a DDoS attack.
- The next goal is to Generate adversarial data using the selected feature set and evaluate the IDS if it can detect the adversarial attack, followed by preparing the IDS with the produced adversarial data.
- This work propose a polymorphic engine that updates the feature profile of the attack by Manual feature update and Automated feature update.
- The research work have conducted a comprehensive simulation and analyzed the results to compare the Reinforcement Learning method against the Manual Feature profile attacks and presented how many cycles an attacker can bypass an IDS with polymorphic adversarial DDoS attacks.

The primary objective of the Generative Model is to learn the unknown probability distribution of the population from which the training observations are sampled from. The most popular GAN architectures are DCGAN[18], Conditional GAN[19], BiGAN[20], Cycle GAN[21].

III. Methodologies

A. Datasets and Feature Selection

Datasets: This work uses a dataset published by the Canadian Institute of Cyber Security, CIC-IDS2017, published in [22] by Lashkari et al., which, according to the authors, supersedes the datasets generated earlier by the institute. CI-CIDS2017 consists of eight different files that contain regular traffic and attack traffic data. Moreover, this dataset consists of various types of attacks along with the normal network flow. This dataset also covers all the available standard protocols like HTTP, HTTPS, FTP, SSH, and email protocols. The dataset consists of more than 70 features that are important as per the latest network standards, and most of them were not available in the previously known datasets.

Feature Selection: Feature selection is an essential aspect of the Deep Learning technique. SHAP (Shapley Additive ex-Planations) [23] is one of the new feature selection techniques. The goal of the proposed method is to signify the contribution of each feature to the predicted value. Two critical measures to define feature importance are Consistency and Accuracy. The authors of the paper discuss that SHAP is the method that satisfies these qualities. The SHAP values explained by the authors are based on Shapley values that are a concept from game theory. The idea behind Shapely values is that the outcome of each possible combination (or coalition) of each feature needs to be examined to determine the importance of a single feature. The mathematical explanation of this is as follows in Equation 1:

$$g(z') = \phi_0 + \sum_{J=1}^{M} \phi_j {Z_j}' \tag{1}$$

Here, g represents the overall result of the Shapely values, $z' \epsilon \{0,1\}^M$ is a coalition vector, M is the max coalition size, and ϕ represents the presence of feature j that contributes towards the final output. The authors have described a coalition vector as simplified features in the paper. In coalition vector, 0 means the corresponding value is not present" and 1 means it is "present." Equation 1 can be called a power set and can be explained as a tree as follows.

Equation 1 can be called a power set and can be explained as a tree shown in Figure. 1 as follows

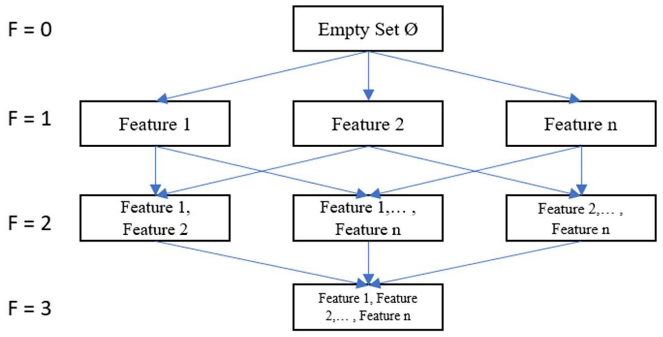


Fig. 1. Power Set of Features.

Each node here represents a coalition of features. Edges represent the inclusion of a feature that was not present in the previous coalition. Equation 1 trains each coalition in the power set of the features to find the most critical feature from the dataset.

The following results were obtained as shown in Fig. 2 by running the SHAP explainability model on the CICIDS2017 data file that shows the list of essential features responsible for the DDoS attack in the most important to least important order. Furthermore, the dark red color represents a higher impact of a feature, and the blue color represents a lower impact of a feature on the output value.

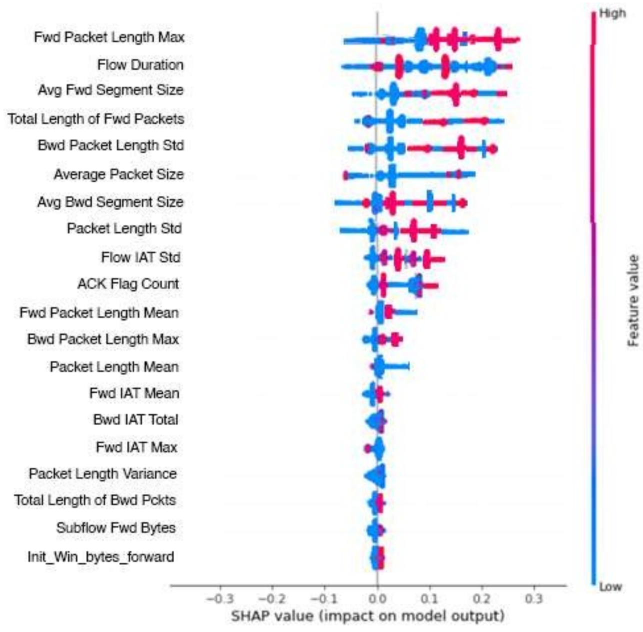


Fig. 2. Summary Plot with Feature Impact using SHAP.

So, from the results obtained in the Fig. 2, This work used these features like functional features that contribute to the DDoS attacks.

B. Adversarial Attack Generation using Wasserstein GAN

The methodologies used in this research involves the Generative Adversarial model that produces adversarial attacks, training IDS by earlier generated polymorphic datasets, polymorphic engine to generate polymorphic DDoS attacks, and use the polymorphic data to attack the IDS. DDoS attack data from the CICIDS2017 [22] used to Generate the adversarial attack by combining a random noise vector of the same size as the selected features from the dataset to train the model. The framework is a feed-forward neural network that consists of 5 linear layers. The input layer consists of neurons as per the selected number of features, and the output layer consists of 1 neuron as shown in Fig. 3.

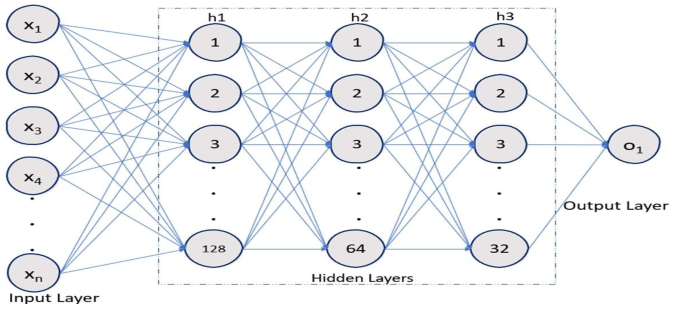


Fig. 3. Neural Network of the Generator.

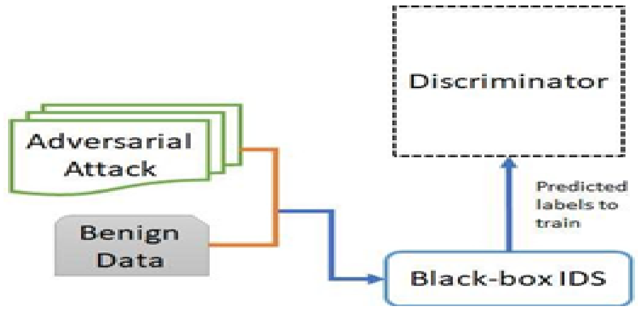


Fig. 4. Training the Black-box IDS.

The input layer receives several numbers of features according to the experiment, and the output layer generates the desired data. The Generator consists of 3 hidden layers that are optimal for this scenario; the results showed fewer layers would underfit the training data. Anything more than that overfits the training data.

In the next step, the generated adversarial attack combined with the benign or normal network flow data will be fed to the Intrusion Detection System.

The IDS will detect the attack and sends predicted labels to the Discriminator as shown in Fig. 4, the detection success rate, and the Discriminator will send the critique to the Generator using the backpropagation so that in the next cycle, the Generator can improve the production of adversarial DDoS attack. The IDS consists of 4 layers, from which the input and output layer consists of 2 neurons each. The IDS consists of 2 hidden layers that are ideal because it only detects if the test data consists of an attack or benign.

The signature-based black-box intrusion detection system used to test the detection rate of the adversarial DDoS attacks. The reason for using this is that most of the time, the type of attack detection system is unknown to the attackers. Attackers rely on the responses received from the detection system, and black-box IDS is the right choice for this model as shown in Fig. 5. The input layer accepts two types of data from the black-box IDS. The output layer provides two critics, one for the Generator and one for itself.

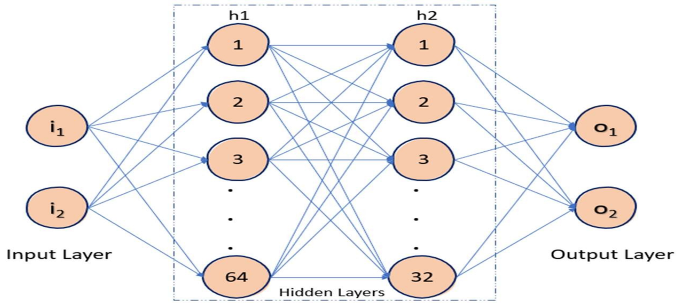


Fig. 5. Neural Network of the IDS and the Discriminator.

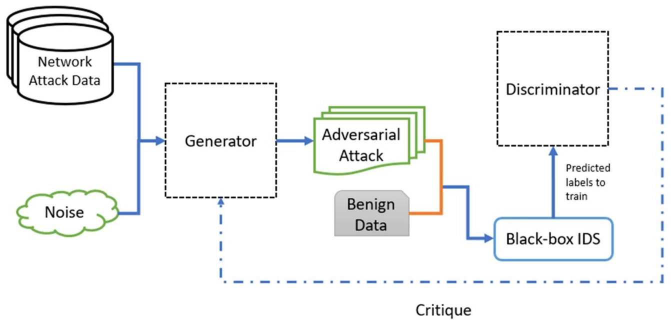


Fig. 6. Generating Adversarial DDoS Attack.

Loss functions used to calculate the Loss[17] for the Generator and the discriminator, shown in the Equation Equation 2 as follows.

$$P_G = E_{M_{\epsilon S_{attack}, N}} - D(G(M, N)) \tag{2}$$

Figure 6 depicts that the generated adversarial data is DDoS attack or abnormal or normal. Here, P_G represents the Penalty to the Generator in attack vector, and in noise vector. E is calculated random inputs value to the model. S_{attack} represents. If the penalty is less to the model means the model is performing well and produces attack datasets that can bypass IDS shown in Equation 3.

$$P_D = A_{S_{\epsilon B_{bengine}}} D(S) + A_{S_{\epsilon B_{attack}}} - B_{S_{\epsilon B_{attack}}} D(S) \tag{3}$$

Here,P_D represents the Penalty to the discriminator. "E" is overall calculated feature values of the models attack datasets. "A" is the actual feature value of benign and the attack data. The lesser the penalty to the discriminator means the discriminator performs well. It calculates if the generated data is closer to the DDoS attack or benign or regular data.

Algorithm – 1 shows the process that was represented in figure 5.

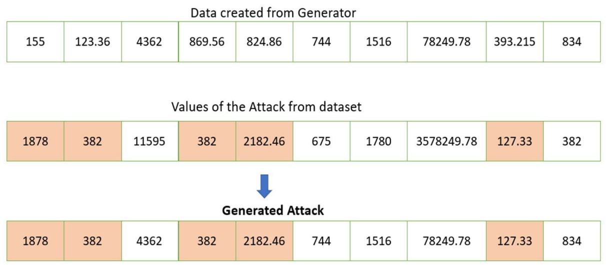
Algorithm 1 Adversarial Attack Generator

```
Require: Input:
  Initiator-noisy vector N, DDoS Attack Datasets
  Critic / Discriminator - Sattack, and Sbenign
  Output: Trained Critic / Discriminator and Generator
  1: for epochs = 1, ... , Maximum EPOCHS do
  2: for N-iterations, do
  3: Initiator create adversarial intrrrusion attack using Sattack, and
  Revise the penalty by PG once it receives the critique.
  4: end loop
  5: While generating adversarial DDoS data and feed the data to IDS to test if it
  detects the attack.
  6: for D-iterations, do
  7: receive detected labels from the IDS and sends a critic to the Generator.
  Update the penalty using PD function.
  8: end loop
  9: end loop
```

C. How the Generator Fabricate an Adversarial Attack

This section specifies the details about the learning process of the Generator and how it produces adversarial data. If the generator continuously generates random data, the data will be unmeaningful, which can change the entire network flow data. So, the Generator needs to produce the data to maintain the intensity of an attack. To ensure that, the work need to maintain the feature values constant that have higher SHAP values as shown in Fig. 2.

Here is a sample of how the Generator produces an adversarial attack by the proposed technique. In this diagram, the darker shade explains the feature values of the features that are contributing to the attack. Whereas non highlighted values depict the feature value of a regular or non-attack feature.



Data created from Generator

155	123.36	4362	869.56	824.86	744	1516	78249.78	393.215	834

Values of the Attack from dataset

1878	382	11595	382	2182.46	675	1780	3578249.78	127.33	382

Generated Attack

1878	382	4362	382	2182.46	744	1516	78249.78	127.33	834

Fig. 7. The Process to Generate Adversarial DDoS Attack.

This Fig. 7 explains that to maintain the attack's intensity, the study need to keep that functional attack features static and only change the feature values that are not contributing to the attack

D. Training an IDS with the Earlier Created Adversarial Data

Fig. 8 depicts IDS training process to evaluate IDS performance with the adversarial data. In this section, the study I will discuss the training of the IDS to evaluate the performance of IDS. Following diagram that depicts the training process.

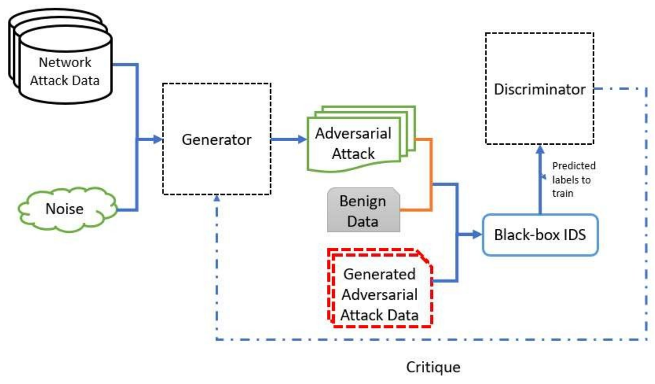


Fig. 8. Training the Black-Box IDS.

This research work considered three inputs to train the IDS: normal or benign data, new adversarial data, and previously generated adversarial data. The IDS learns about the adversarial data and tries to detect the DDoS attack data. Algorithm 2 suggests the overall process for the same.

Algorithm 2 Training IDS by Adversarial DDoS datasets

```
Require: Generator – N noisy error data + Initial Attack Data
  IDS – Benign or Normal Data, Adversarial Datassets, and Earlier Generated Attack Data
  Critic / Discriminator – Sattack and Sbenign
  Output:
  Critic / Discriminator, Generator, and trained IDS
  1: for each epochs = 1 , . . . , MAX EPOCHS do
  2: for G-repetitions, do
  3: The generator generates attacks from datasets using Sattack and Renew loss applying PG function
  4: end of the loop
  5: for D-repetitions, do
  6: Critic / Discriminator distinguishes this data to Bbenign and Battack
  7: Renew loss applying PD function
  8: Feed Battack (attack data) and Earlier Generated attack Data
  9: end loop
```

E. Polymorphic Engine to Generate Polymorphic Attack

Three different methods used for the Polymorphic engine to generate Polymorphic Attack are as follows.

1. Update new features in the attack profile after the IDS detects previous adversarial attacks. Algorithm 3 will discuss the process.

Algorithm 3

```
Require: Input – Use five functional attack features with a high impact score from the
Ensure: shortlisted features and five normal features.
  1: Generate adversarial DDoS data and attack the IDS.
  2: Train the IDS so that it can detect previously generated adversarial DDoS data.
  3: Use the same set of features to generate an adversarial DDoS attack. Again, go to step – 2.
  If the Generator fails to evade the IDS, choose one functional feature with a high
  feature score, one normal or benign feature from the predefined set of features, and swap
  them with the used features.
  4: Go to step – 1.
  5: In the end, the IDS will detect all the Polymorphic adversarial DDoS attacks;
  the program will stop.
```

2. Add new features from the predefined list of features in the current attack profile after the IDS detects previous adversarial attacks, and the following algorithm 4 will discuss the process.

Algorithm 4

```
Require: Input – Use five functional attack features with a high impact score from
  the shortlisted features and five normal features.
Ensure: shortlisted features and five normal features.
  1: Generate adversarial DDoS data and attack the IDS.
  2: Train the IDS so that it can detect previously generated adversarial DDoS data.
  3: Use the same set of features to generate an adversarial DDoS attack. Again, go to step – 2.
  If the Generator cannot deceive the IDS with the same set of features, choose one new
  functional feature with a high impact score, one feature that represents benign
  data, and add them to the previous attack profile.
  4: Go to step – 1.
  5: At the end, the IDS will detect all the Polymorphic adversarial DDoS attacks. The program will stop.
```

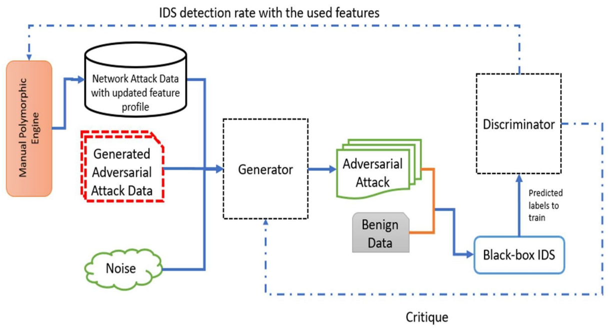

IDS detection rate with the used features
Manual Polymorphic Engine
Network Attack Data with updated feature profile
Generated Adversarial Attack Data
Noise
Generator
Adversarial Attack
Benign Data
Discriminator
Predicted labels to train
Black-box IDS
Critique

Fig. 9. Manual Process to Generate Polymorphic Adversarial Attack.

In the above methods shown in Fig. 9, the research work assumed that an attacker would manually modify the feature profile and train the model with the new feature profile every time the ISD detects a polymorphic attack. This study considered using only a total of 20 features that were provided by the SHAP method.

3. It will be challenging to keep manually changing the feature profile if the study will use more than 20 features. So as an alternative a Reinforcement Learning method has been used to automate the feature profile selection for generating a polymorphic attack as shown in Fig. 10.

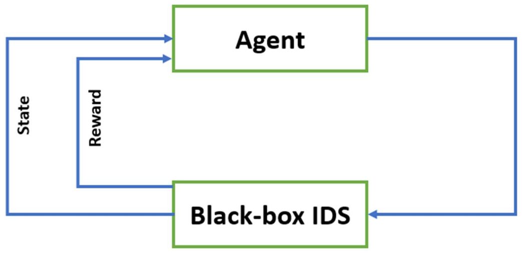


Fig. 10. Function of RL in this Framework.

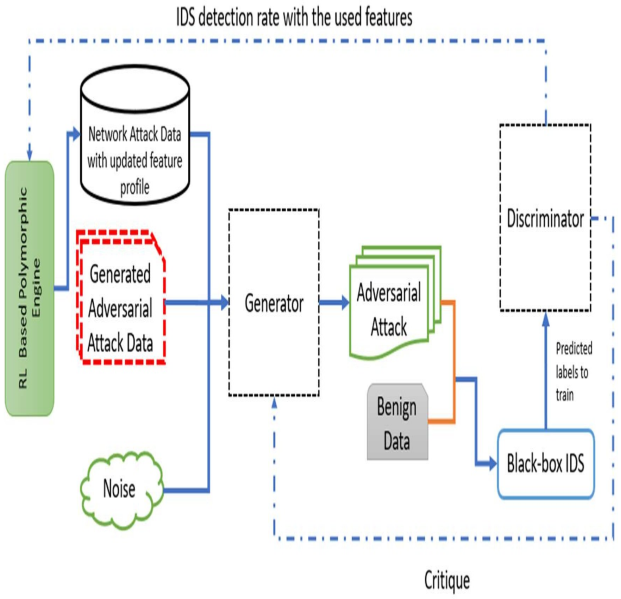

IDS detection rate with the used features
RL Based Polymorphic Engine
Network Attack Data with updated feature profile
Generated Adversarial Attack Data
Noise
Generator
Adversarial Attack
Benign Data
Discriminator
Predicted labels to train
Black-box IDS
Critique

Fig. 11. Automated RL that Generates Polymorphic Adversarial Attack.

The Reinforcement Learning method is an ML-based technique that focuses on retraining the algorithm following a trial-and-error approach. The agent in this architecture evaluates the current IDS attack detection score. Then the agent takes action and receives feedback from IDS. Positive feedback is a reward, and negative feedback is a penalty to the agent. The following algorithm will explain the process. The overall process of generating a polymorphic attack is explained in the following algorithm 5 and Fig. 11.

Algorithm 5

```
Require: Input – Use any five features with a high impact score and any 5 with the lowest score from the shortlisted features.
Ensure: shortlisted features and five normal features.
  1: Generate adversarial DDoS data and attack the IDS.
  2: Train the IDS and check if the adversarial attack evades the IDS. Continue using the current feature set to generate an attack.
  3: Get the attack success rate; if the attack FAILS to evade, The RL algorithm adds new features in the existing feature set to
  generate a polymorphic attack.
  4: If the new polymorphic attack fails to evade the IDS, the RL algorithm will get a penalty. The RL will ignore these features, and
  if the new polymorphic attack evades the IDS, the RL will get a reward.
  5: The RL agent will learn combinations of the attack feature profile and generate a new polymorphic adversarial DDoS attack.
  6: The algorithm stops when the Generator can no longer generate a polymorphic adversarial attack.
```

F. *Performance Evaluation*

To evaluate the performance and the results of this work, the research work used the following parameters.

Accuracy - Represents the fraction of precisely classified data in comparison to the total processed data. The formula to calculate accuracy is as follows

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{4}$$

Precision – a ratio between True Positive values and all the positive values received from the Deep Learning model.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall – a ratio between correctly detected samples over total sample data. It is also known as a ratio between True Positives and the sum of True Positives and False Negatives.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

F1-Score – a calculation of a mean of precision and recall.

$$F1 - Score = 2X \frac{Precision X Recall}{Precision + Recall} \tag{7}$$

IV. Results and Discussion

The experimental setup has done by using libraries like PyTorch, Scikit-learn, Pandas, Numpy, Matplotlib. Hyper-parameters are essential properties that define the characteristics of the training process of the Deep Learning model. The hyper-parameters used in this research are BatchSize, learningrate, CriticIters, Optimizer, Epochs to the optimization and training process of the model.

This section describes the results of various experiments for different scenarios and analyses of findings.

A. Attack Generation

The first step of the research is to generate an adversarial DDoS attack to evade this Black-box IDS. As seen in Fig. 12 graph initially, Generator produces data that is unable to bypass the IDS. However, after training the Generator for 100 epochs, it discovers to create adversarial data to deceive IDS.

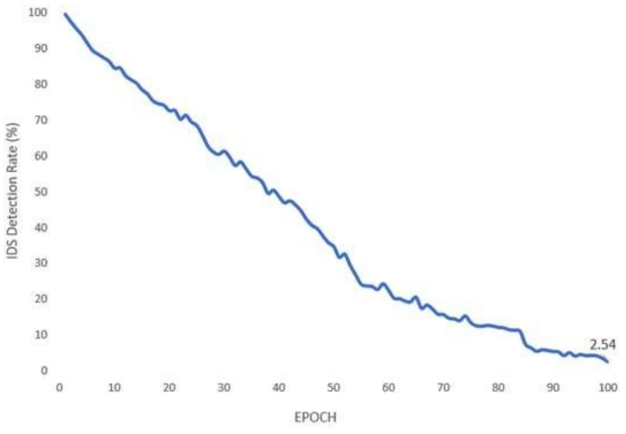


Fig. 12. Adversarial DDoS Attack Generation.

B. Training IDS by Adversarial DDoS Information

This section describe the result of the discovery time of IDS after training. As shown in Fig. 13, in initial cycles, IDS struggles to detect the attacks. However, after training it for 100 epochs, it detects almost all the attacks.

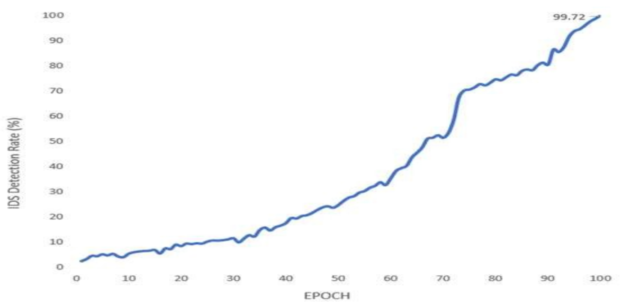


Fig. 13. Detection Rate after Training the IDS.

C. Polymorphic Adversarial DDoS Attack Generation

This section illustrates the detection rate of the Black Box IDS under the generation of polymorphic adversarial attacks.

In the first experiment, new features have been selected manually to produce polymorphic attacks. For this test, limited features from the datasets have been used. The following is the initial result using algorithm 3.

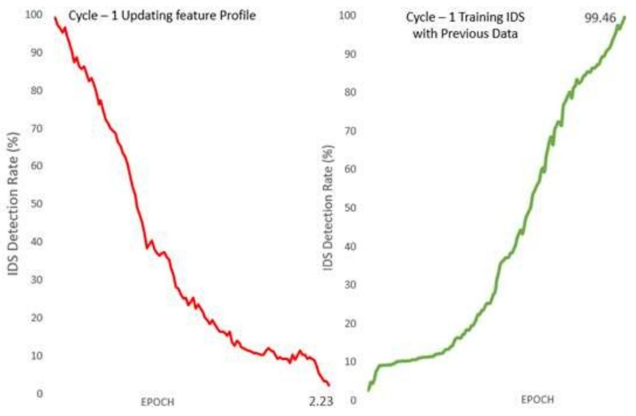


Fig. 14. Polymorphic Adversarial DDoS Attack using Algorithm 3.

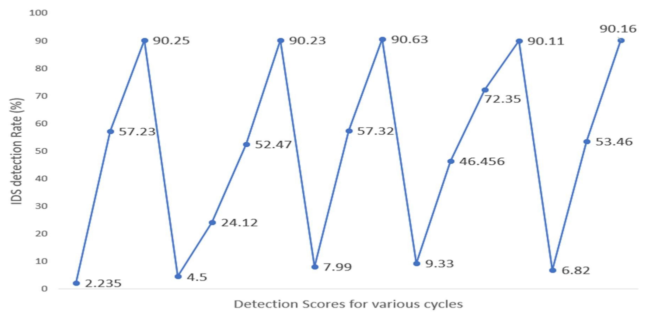


Fig. 15. IDS Detection Rate for Each Attack Cycle (using algorithm 3).

In the Fig. 14, above result, a red-colored graph suggests a polymorphic attack being generated and proceed towards the BlackBox IDS. As seen, the polymorphic attack can deceive the IDS. The green-coloured graph depicts the training of IDS by earlier generated polymorphic adversarial DDoS datasets. After 100 epochs, IDS detects the polymorphic adversarial DDoS attack. The following result indicates all the cycles of polymorphic attacks on the IDS. The Generator utilizes the same combination of the features to generate attacks until an IDS detects all the previous attacks.

Each data point in Fig. 15 depicts the IDS detection rate. Once the IDS detects all the previous versions of the polymorphic DDoS attack that uses the same feature set (as seen in Fig. 15), the generator manually selects new predefined features and generates a new polymorphic adversarial DDoS attack. For this test, only a group of 10 features have been used.

In the next test, the research work used a technique that follows algorithm 4 to revise the attack to generate a polymorphic adversarial DDoS attack. For this experiment, the work has been began with ten features to generate polymorphic attack data. To generate a new polymorphic attack, two new features have been added in the existing attack data and used a total of 20 features. The Fig. 16 is the first result of the initial polymorphic attack.

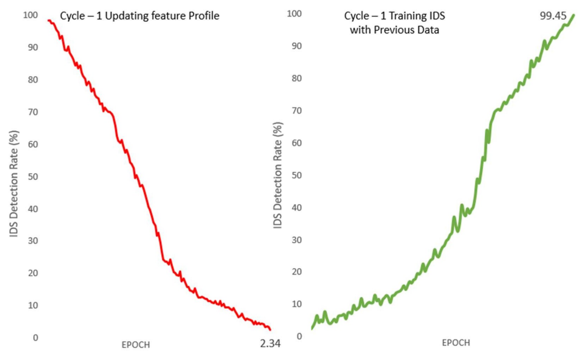


Fig. 16. Polymorphic Adversarial DDoS Attack using Algorithm 4.

Each data point in Fig. 17 depicts the IDS detection rate. Once the IDS detects all the previous versions of the polymorphic DDoS attack that uses the same feature set, the generator manually selects new predefined features and generates a new polymorphic adversarial DDoS attack. For this test, a group of 20 features have been used. In this test, the Generator can deceive the IDS for a total of 18 cycles using this technique.

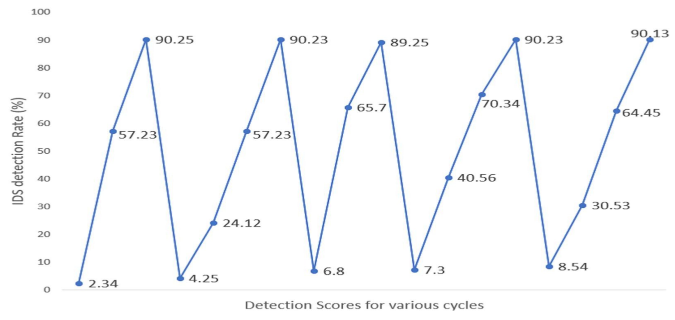


Fig. 17. IDS Detection Rate for Each Attack Cycle (using algorithm 4).

The first two experiments focus on testing if the Generator can produce polymorphic adversarial DDoS attack data by updating the feature profile manually. After confirming the possibility of doing so, the next step is to automatically select features and manipulate the attack feature profile to generate polymorphic adversarial attack data. To automate this task, the Reinforcement Learning technique has been applied. It receives an IDS detection rate and learns to select new features, add them to the old feature set, and create a new feature set. This experiment also indicates the number of times a generator can produce polymorphic adversarial DDoS data. To examine this,four sets of feature combinations have been used for each test to generate the automated Polymorphic adversarial DDoS attack.

- The first test includes a total of 40 features from the dataset
- The second test includes a total of 50 features from the dataset
- The third test includes a total of 60 features from the dataset
- The fourth test includes a total of 76 features from the dataset...

The above experiments begin with ten features, from which 5 are a functional feature with a high impact score, and 5 are usual or benign.

D. Test Evaluation

The Table I describes the overall values for the Precision, Recall, and F1-score for each test.

TABLE I. NONLINEAR MODEL RESULTS

Sl. No.	TEST	ACCURACY	PRECISION	RECALL	F1-SCORE
1	Manual Test – 1 (using Algorithm 2)	98.58	96.24	92.91	0.953
2	Automated Test using 40 features (using Algorithm 4)	98.27	94.41	92.44	0.935
3	Automated Test using 50 features (using Algorithm 4)	96.97	93.58	91.69	0.928
4	Automated Test using 60 features (using Algorithm 4)	96.34	93.22	91.43	0.921
5	Automated Test using 76 features (using Algorithm 4)	94.42	91.79	91.58	0.916

E. Analysis

This research work ran 5 test scenarios with different feature combinations. 2 experiments consist of a manual feature selection technique to generate polymorphic adversarial DDoS attack data and four tests with an Automated feature selection technique. A manual feature selection technique utilized as a benchmark and compared this technique to the automated feature selection technique to analyze for how many cycles the polymorphic attack evades the Black-Box IDS.

The following Fig. 18 graphs will be useful to compare these five different scenarios.

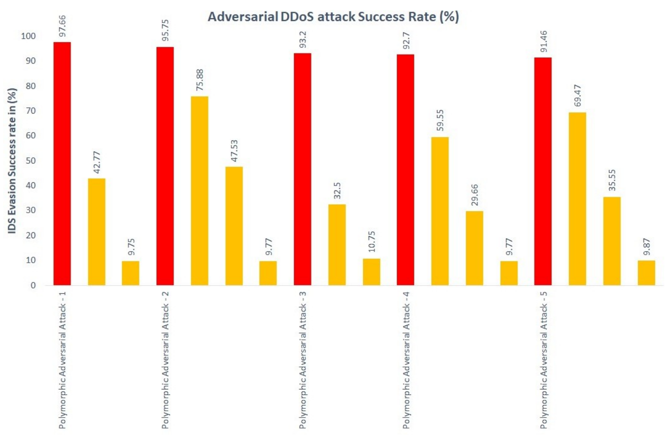


Fig. 18. Test - 1 Polymorphic Adversarial Attacks using Manual Feature Selection.

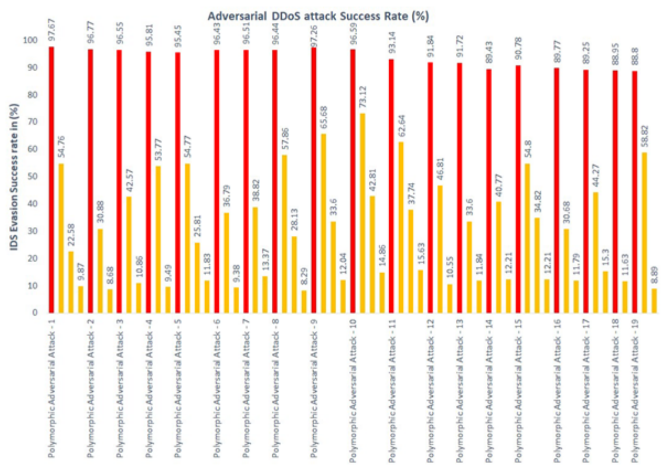


Fig. 19. Test - 2 Polymorphic Adversarial Attacks using Automated Feature Selection.

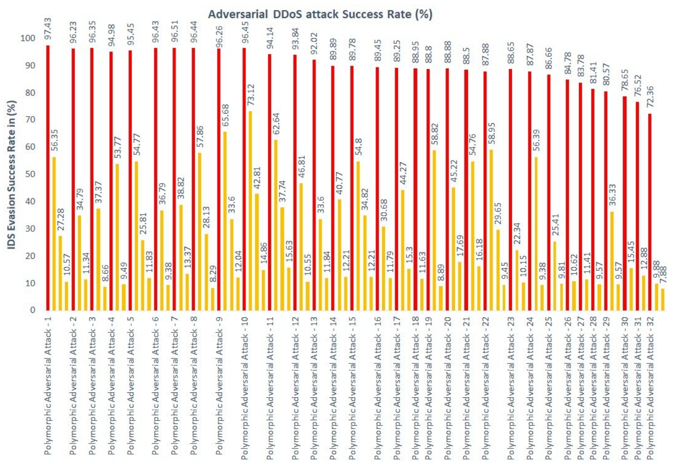


Fig. 22. Test - 5 Polymorphic Adversarial Attacks using Automated Feature Selection.

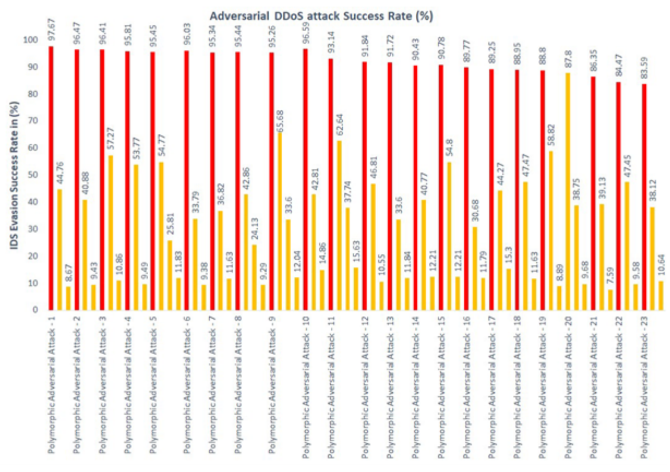


Fig. 20. Test - 3 Polymorphic Adversarial Attacks using Automated Feature Selection.

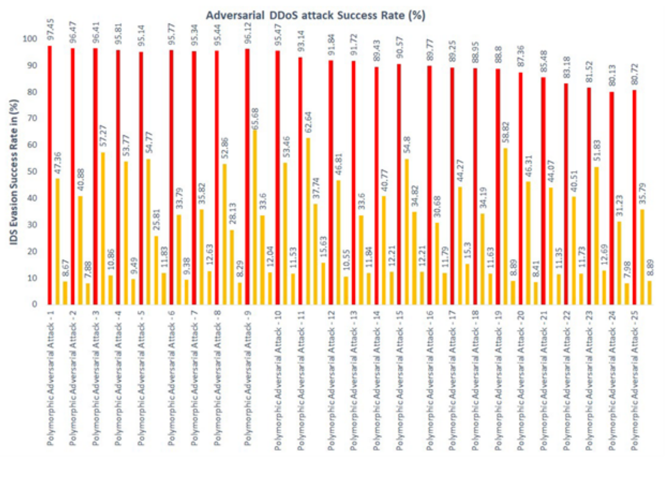


Fig. 21. Test - 4 Polymorphic Adversarial Attacks using Automated Feature Selection.

In all the above results shown in Fig. 18, 19, 20, 21, 22, the Polymorphic DDoS adversarial attack successfully evading the IDS; the orange bar suggests the polymorphic attack is becoming weak once the IDS detects them. By counting the red bar, It has been observed that how many times the Generator produced a polymorphic attack in each cycle. Fig. 19 suggest that when the Generator uses a small number of features, more than 90% of the polymorphic attack evades the IDS. By noticing these figures, it is clear that using fewer features to generate a polymorphic attack has a higher evasion rate but fewer chances of generating more polymorphic attacks.

Fig. 20, 21, 22 suggest that initially, more than 90% of the polymorphic attacks can evade the IDS. However, results propose that if the Generator utilizes more features to generate a polymorphic DDoS attack, the success rate gets lower each time. Comparing all the results confirms that while using a fewer number of features to generate polymorphic adversarial DDoS attacks, the attack success rate stays up to the acceptable amount. However, when more features have been used, the attack success rate depletes after certain cycles.

Now the Table II describes the total runtime for each experiment.

TABLE II. Model Evaluation

Sl. No.	TEST	TOTAL RUNTIME
1	Test – 1 Manual Feature profile update (with a total of 10 features)	30.43 minutes
2	Test – 3 Automated Feature profile update (with a total of 40 features)	75.31 minutes
3	Test – 5 Automated Feature profile update (with a total of 50 features)	90.45 minutes
4	Test – 6 Automated Feature profile update (with a total of 60 features)	145.37 minutes
5	Test – 5 Automated Feature profile update (with a total of 76 features)	173.55 minutes

As observed from the above table, if the test uses a small number of features, it takes less time to run the simulation. The run time rises upon increasing features to generate a polymorphic DDoS attack.

V. Conclusions and Future Work

The work proposed a framework to create polymorphic adversarial DDoS attacks using a CICIDS2017 dataset using a Wasserstein GAN. To generate polymorphic attacks, three different techniques have been proposed that change the feature profile of the attack. New features have been selected manually each time to generate polymorphic adversarial attacks in the first two techniques. Furthermore, to automate the feature selection to generate polymorphic attacks, a Reinforcement Learning technique has been applied in each technique; the Generator creates a polymorphic attack until no more new features are remaining to choose from the feature set.

From the results, it has been observed that the Generator can produce polymorphic adversarial DDoS. Results also depict that while using a small number of features to create a polymorphic attack, the attacks were successfully deceiving the IDS with more than a 90% success rate while using a manual selection of features.

In the future, it could be interesting to consider using other variants of GAN like DCGAN, Conditional GAN, BiGAN, Cycle GAN to generate adversarial network attack data and evaluate the detection systems. Another limitation of this research is that it focused on generating only one type of attack, as every attack has different functional features. It would be difficult to use one Generator to create other types of attacks with the same generator. So it would be interesting to use multiple generators for each type of attack and evaluate the performance of the IDS against all types of polymorphic adversarial network attacks.

Acknowledgment

This work was supported by the Research Center of PES Institute of Technology and Management, Shivamogga, Karnataka, Computer Science and Engineering Department affiliated to Visvesvarya Technological University and APS College of Engineering, Bangalore.The authors are grateful for this support. I sincerely thank all those who helped me in completing this task.

References

[1] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," Journal of Network and Computer Applications, vol. 36, no. 1, pp. 16–24, Jan. 2013, DOI: 10.1016/j.jnca.2012.09.004.

[2] U. Sabeel, S. S. Heydari, H. Mohanka, Y. Bendhaou, K. Elgazzar and K. El-Khatib, "Evaluation of Deep Learning in Detecting Unknown Network Attacks," 2019 International Conference on Smart Applications, Communications and Networking (SmartNets), Sharm El Sheik, Egypt, 2019, pp. 1-6, doi: 10.1109/SmartNets48225.2019.9069788.

[3] M. Gadelrab, A. A. El Kalam, and Y. Deswarte, "Manipulation of Network Traffic Traces for Security Evaluation," in 2009 International Conference on Advanced Information Networking and Applications Workshops, May 2009, pp. 1124–1129, DOI: 10.1109/WAINA.2009.36.

[4] F. Skopik, G. Settanni, R. Fiedler, and I. Friedberg, "Semi-synthetic data set generation for security software evaluation," in 2014 Twelfth Annual International Conference on Privacy, Security and Trust, Jul. 2014, pp. 156–163, DOI: 10.1109/PST.2014.6890935.

[5] "CrowdStrike Introduces Enhanced Endpoint Machine Learning Capabilities and Advanced Endpoint Protection Modules." Internet: https://www.crowdstrike.com/resources/news/crowdstrike-introduces-enhancedendpoint- machine-learning-capabilities-and-advanced-endpoint-protection-modules

[6] M. Berninger and A. Sopan. "Reverse Engineering the Analyst: Building Machine Learning Models for the SOC." Internet: https://www.fireeye.com/blog/threatresearch/ 2018/06/buildmachine-learning-models-for-the-soc.html

[7] "Use Cases: Demisto's Top Machine Learning Use Cases – Part 1." Internet: https://blog.demisto.com/demistos-top-machine-learning-use-cases-part-1

[8] I. Goodfellow et al., "Generative Adversarial Nets," in Advances in Neural Information Processing Systems 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680.

[9] S. Yu, H. Dong, F. Liang, Y. Mo, C. Wu, and Y. Guo, "SIMGAN: Photo-Realistic Semantic Image Manipulation Using Generative Adversarial Networks," in 2019 IEEE International Conference on Image Processing (ICIP), Sep. 2019, pp. 734–738, DOI: 10.1109/ICIP.2019.8804285.

[10] C. Wan, S. Chuang, and H. Lee, "Towards Audio to Scene Image Synthesis Using Generative Adversarial Network," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Brighton, United Kingdom, 2019, pp. 496-500, DOI: 10.1109/ICASSP.2019.8682383.

[11] Y. Yang, X. Dan, X. Qiu, and Z. Gao, "FGGAN: Feature-Guiding Generative Adversarial Networks for Text Generation," in IEEE Access, vol. 8, pp. 105217- 105225, 2020, DOI: 10.1109/ACCESS.2020.2993928.

[12] J. Zhang, Q. Yan, and M. Wang, "Evasion Attacks Based on Wasserstein Generative Adversarial Network," 2019 Computing, Communications and IoT Applications (ComComAp), 2019, doi: 10.1109/ComComAp46287.2019.9018647.

[13] M. Kawai, K. Ota, and M. Dong, "Improved MalGAN: Avoiding Malware Detector by Leaning Cleanware Features," in 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Feb. 2019, pp. 040–045, DOI: 10.1109/ICAIIC.2019.8669079.

[14] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," arXiv:1701.07875 [cs, stat], Dec. 2017, Accessed: Aug. 19, 2020. [Online]. Available: http://arxiv.org/abs/1701.07875.

[15] H. Xie, K. Lv, and C. Hu, "An Effective Method to Generate Simulated Attack Data Based on Generative Adversarial Nets," in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), Aug. 2018, pp. 1777–1784, DOI: 10.1109/TrustCom/BigDataSE.2018.00268.

[16] M. Ring, D. Schlör, D. Landes, and A. Hotho, "Flow-based Network Traffic Generation using Generative Adversarial Networks," Computers and Security, vol. 82, pp. 156–172, May 2019, DOI: 10.1016/j.cose.2018.12.012.

[17] Z. Lin, Y. Shi, and Z. Xue, "IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection," arXiv:1809.02077 [cs], Jun. 2019, Available: http://arxiv.org/abs/1809.02077.

[18] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," arXiv:1511.06434 [cs], Jan. 2016, Available: http://arxiv.org/abs/1511.06434.

[19] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," arXiv:1411.1784 [cs, stat], Nov. 2014, Available: http://arxiv.org/abs/1411.1784.

[20] U. Mutlu and E. Alpaydın, "Training bidirectional generative adversarial networks with hints," Pattern Recognition, vol. 103, p. 107320, Jul. 2020, doi: 10.1016/j.patcog.2020.107320.

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," in 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 2242–2251, doi: 10.1109/ICCV.2017.244.

[22] I. Sharafaldin, A. Lashkari, and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018

[23] J. Hui, "GAN — DCGAN (Deep convolutional generative adversarial networks)," Medium, Jun. 24, 2018. https://medium.com/@jonathanhui/gan-dcgan-deepconvolutional-generative-adversarial-networks-df855c438f