WF-GAN: Fighting Back Against Website Fingerprinting Attack Using Adversarial Learning

Chengshang Hou[†‡], Gaopeng Gou[†‡], Junzheng Shi[†‡], Peipei Fu[†‡] Gang Xiong[†‡*],
[†]*Institute of Information Engineering*, *Chinese Academy of Sciences*
[‡]*School of Cyber Security*, *University of Chinese Academy of Sciences*
Beijing, China
{houchengshang, gougaopeng, shijunzheng, fupeipei, xionggang}@iie.ac.cn

***Abstract*—Website Fingerprinting (WF) attack is an side-channel attack which aims at encrypted web traffic. WF attackers recognize encrypted website traffic through constructing fingerprinting for each website using the flow-based features extracted from encrypted traffic. WF defense typically aims at modifying the features of the encrypted websites. However, those countermeasures either cause high overhead or fail to counter the subsequent WF attacks. Especially, the newest WF attacks, which are based on deep neural network, is able to classify the defended traffic by directly learning from the labeled defended traffic. In this paper, we propose an novel defense through making use of the trick that machine learning models are vulnerable to adversarial exmaples. We design WF-GAN, a GAN with an additional WF classifier component, to generate adversarial examples for WF classifiers through adversarial learning. As the website set is divided into source and target website, WF-GAN are trained to map websites features from source set to adversarial examples and make adversarial examples more similar to the website features in the target set. The experimental result shows that WF-GAN achieves 90% success rate with at most 15% overhead for untargeted defense, which outperforms previous defense. In addition, adversarial examples based defense support targeted defense, which is not support by traditional defense. The result shows that WF-GAN achieves over 90% targeted defense success rate when the target websites set is twice as many as the source website set.**

***Index Terms*—network security, privacy, adversarial learning**

I. Introduction

Website fingerprinting (WF) attacks typically target at encrypted tunnels such as SSL/TLS, SSH and VPN, which are used to secure network communications. Especially, the attack causes privacy risk to the Tor [1], which is the widely used anonymity network. The attackers construct unique fingerprinting to recognize encrypted website traffic through modeling the flow-based features of encrypted traffic. As supervised machine learing applied, the construction of website fingerprinting is integrated into the process of training a classifer [2], [3]. Firstly, primitive flow-based features, including packets length sequence and packet direction sequence, are extracted from traffic flows. Then, high order semantic features (e.g. the number of outgoing and incoming packet, the consecutive packet in the same direction) are computed based on those primitive features treated as the input of WF classifiers. Finally, WF classifier is trained to learn the label of traffic flow in the training stage and predicts label of traffic flow in the testing stage.

*Gang Xiong is the corresponding author, Email: xionggang@iie.ac.cn.

Recently, several studies make use of deep learning to directly classify the primitive features of encrypted traffic [4]–[6]. Despite the state-of-the-art WF defenses consuming a high overhead to defend against WF attack, WF attack based on deep neural network exhibits the capability of identifying the defended traffic when using the labeled defened traffic to train nerual networks. Fortunately, deep neural networks have been proven to be vulnerable to adversarial examples [7], [8]. Adversarial machine learning community show that adversarial examples are effective to defend against machine learning based attacks [9].

In this paper, we make use of adversarial examples to defend against WF attack. Adversarial examples are small perturbations of the original examples which result in classifier misclassify the original example to a specific class (targeted attack) or any classes except for the correct class(non-targeted attack). In addition, WF defenses are supposed to generate synthetic traffic features which are difficult to find the difference between the fake and the real, we consider leverage the recently proposed Generative Adversarial Networks (GANs) to achieve this goal.

Specifically, we combine GAN and adversarial examples to defend against WF attacks. We propose to divide traffic into source traffic and target traffic. We train a generative network to generate perturbation for the traffic features from the source set and to the target traffic features. Simultaneously, we train a discriminator network to classify the source traffic features and target traffic features. The generator produces fake target traffic features by recurrent training both generator and discriminator. Moreover, we train a WF attack classifier to receive the fake traffic features and using an optimized based adversarial generation algorithm to update the parameter of the generator. We name this approach WF-GAN. WF-GAN achieve two goals: (i) perturb the traffic features from original class into the other classes (untargeted defense) and (ii) map the generated traffic features to the most similar targeted traffic (targeted defense). By posing the WF defense problem as an adversarial example generation problem, we significantly improve the defense success rate and reduce the overhead. The key contributions of our work are as follows:

- We propose a novel WF defense using GANs to generate adversarial exmaple for WF attacks. We design a GAN with a WF model components, named WF-GAN. WF-GAN is structured to input both source websites traffic and target websites traffic, which not only have the ability of untargeted defense but also support targeted defense. We train a generative adversarial network to automatically learn to generate adversarial example. The generator after training can generate adversarial example without accessing the WF model anymore.
- We use the burst-based model as WF model to replace the original model which use sequence features as input. By this way, WF-GAN is able to optimize adversarial example using the gradient information. We show the effectiveness of adversarial example based on both white-box model and black-box model. In a black-box model, we use an WF model using a smaller convolutional kernel size than the original model.
- We evaluate WF-GAN and show it outperforms the state-of-the-art defense, Walkie Talkie. It achieves 50% success rate with 30% overhead. On the same attack model and dataset, WF-GAN achieves 90% success rate with 5-15% overhead. For the targeted defense, WF-GAN achieves 90% success rate when the source set size is 30 in the black box scenarios.

The remainder of this paper is structured as follows. We give a brief discussion of website fingerprinting and background of adversarial example in Section II. We describe the design of WF-GAN in Section III. In Section IV and V, we perform comprehensive experiments on a public dataset collected over Tor. Finally, we conclude in Section VI.

II. Related Work

A. WF Attacks and Defenses

1) website fingerprinting attack: WF attacks initially make use of machine learning techniques to extract traffic features and classify encrypted website, such as k-Nearest-Neighbor [10], Support-vector machines [3]. Recently, WF attacks based on these algorithms has been outperformed by WF attack based on Deep Neural Networks (DNNs). Rimmer et al. [4] propose the first DNN-based attack, AWF. They show that DNN-based WF attack can identify thousands of websites by automatcially extract features from the raw traffic data. Sirinam et al. [5] propose Deep Fingerprinting (DF), which defeat the existing defense such as WTF-PAD [11] and Walkie-Talkie [12].

2) Website Fingerprinting Defense: WTF-PAD [11] and Walkie-Talkie [12] are two state-of-the-art low-latency defenses. Juarez et al. [11] propose WTF-PAD, which inserts fake burst features sampled from the traffic features of other classes into the traffic features of the original class. Li et al. [13] demonstrate that WTF-PAD defense is vulnerable from the view of information leakage. Wang and Goldberg [12] propose Walike-Talkie, which mold the other traffic feature into the original traffic features. However, WF attack can achieve high Top-2 accuracy on Walkie-Talkie.

B. Generative Adversarial Networks

Goodfellow et al. [14] propose Generative adversarial Networks (GANs) and use GANs to generates synthetic image that are similar to real image from noise data. Xiao et al. [15] propose AdvGAN to generate adversarial examples for image data, which inspires our work. We adapt AdvGAN to formulate WF-GAN with several modification, including input and loss function.

Rencently, there are a series of studies that apply GANs to process network traffic, including traffic generation [16], intrusion detection evasion [17], adapting malware communication [18], class imbalance in traffic classification [19]. For example, Ring et al. [16] proposed a flow-based traffic generation to augment training data sets for network intrusion detection. They demonstrates GANs are promising to generate high quality traffic features for both consecutive and categorical data. For more related work, Li et al. [20] use GAN to camouflage traffic. They propose FlowGAN, which dynamically mimic benign traffic to circumvent censorship. FlowGAN applies to mimic traffic feature for a specific website while WF-GAN is able to defend many websites at the same time.

C. Adversarial Example Attack

Szegedy et al. [7] are the first to discover that images can be misclassified by deep neural networks through applying a small perturbation. Goodfellow et al. [8] explain adversarial examples are caused by the linear nature of neural networks. They also propose fast gradient sign method (FGSM), an adversarial example generation algorithm that minimizes the L_∞ norm of perturbation. Mardy et al. [21] generalize the FGSM by projected gradient descent (PGD) which iteratively moves adversarial example along the sign of the gradient with projection. Carlini and Wagner [22] propose an optimization-based method (C&W) that supports three norms (L_0, L_2, L_∞), which is currently the strongest adversarial example generation algorithm. WF-GAN apply the C&W algorithm to optimize the generator.

For adversarial example used in WF defense, Imani et al. [23] propose Mockingbird that leveraging adversarial example to defend against WF attack. This work is mostly closed to our defense. Mockingbird claims use C&W to generate adversarial exmaple. However, they neglect the key target function and only rely on L_2 to minimize the distance to a randomly chosed example, which cause Mockingbird have lower success rate and higher overhead than WF-GAN. In addtion, Mockingbird generates adversarial example in an iterative way. While WF-GAN can craft a batch of adversarial example at same time without iteration after GAN training stage, which makes WF-GAN more efficient.

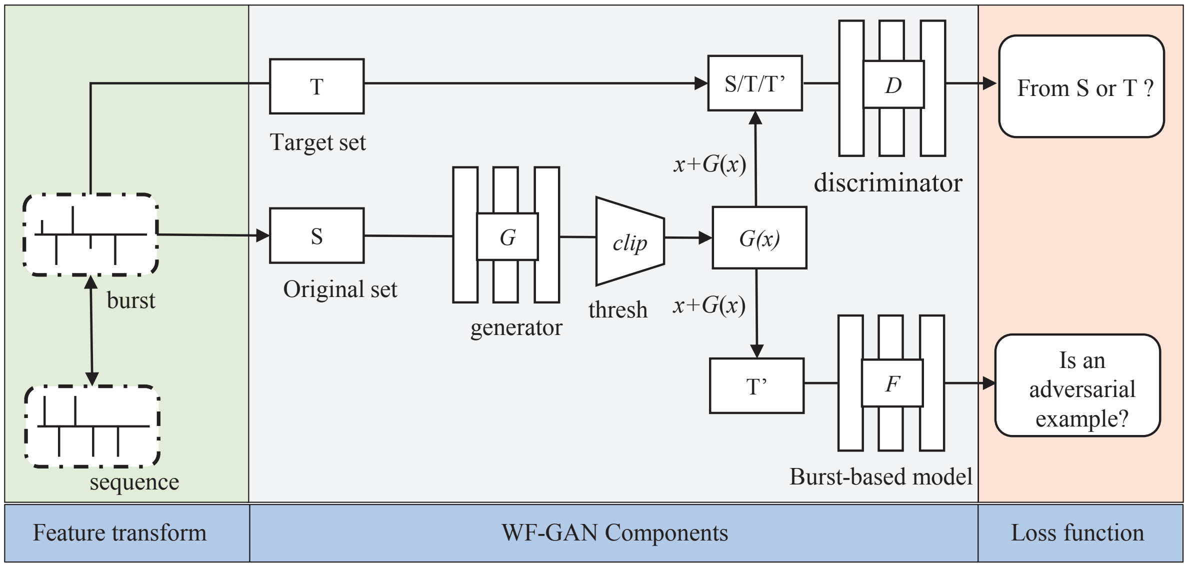

Fig. 1. The WF defense framwork using GAN to generate adversarial exmaples.

III. Methodology

In this section, we present WF-GAN architecture. We refer to the paradigm of AdvGAN [15]. We adapt the discriminator to take both source and target traffic, while the generator takes only source traffic. For the target model, we introduce burst-based model to replace sequence-based model to optimize adversarial examples generated by the generator. For the loss function, we use L_1 distance rather than L_2 distance in traffic features to reduce the blurring introduced by L_2 distance [24]. Fig. 1 illustrates the overview of the WF-GAN components.

A. Feature Transform

All the state-of-the-art DNN-based WF attacks on Tor take the cell sequence as the model input, which can be seen as the packet direction sequence. The WF defense problem in this context can be posed as generate dummy packets (i.e. perturbations) and inserts into the packet sequence. However, the insertion operation cannot be computed directly. We formulate the operation by transforming the *sequence-based* features into *burst-based* features.

Bursts are continuous outgoing and incoming packets. Typically, outgoing packets and incoming packets are represented by 1 and -1, respectively. To encode sequence-based features into burst-based features, consecutive items with the same sign are accumulated. And to decode burst-based features into sequence-based features, each elements in a burst-based features is broken up into -1 and 1. The burst representation of a traffic flow can be directly added by the generated perturbation.

Here we define untargeted and targeted defense in WF-GAN. We firstly divide the original set into S and T according to a ratio (*S/T ratio*). Intuitively, the S/T ratio impact the defense effect. Given a WF model F and traffic x , untargeted defense is to cause x in S misclassify to any classes. And targeted defense is to x in S misclassify to classes in T further.

B. WF-GAN Components

1) Generator: The generator's input is an example x from S and its output is the corresponding perturbations $G(x)$. The generated adversarial example T' is therefore $x + G(x)$. T' is then feed to the WF model as adversarial example and the discriminator as fake T. In the training stage, the generator updates its parameters according to the feedback from both the discriminator and the WF model.

Through this way, it is easy to restrict the adversarial example $x + G(x)$ to a reasonable input for WF model. For example, the outgoing burst cannot add negative value which means the incoming packet. We use *threshold* to limit the maximum inserted packet. This is the clip operation before generator output perturbation.

2) Discriminator: The discriminator judge if the example is come from the S set or T set. In order to improve the stability and rationality of perturbation generated by the generator, the discriminator is trained to learn the adversarial example from S or T. During the training stage, the discriminator is fed with S, T and T'. The corresponding outputs are $D_{x\sim \mathrm{T}}(x)$, $D_{x\sim S}(x)$ and $D_{x'\sim T'}(x')$, where $x' = x + G(x), x \in S$. The recognition ability of the discriminator augments during the training, while the generator outputs perturbation is more similar to target websites and minimize the added overhead.

3) Target model: Since the adversarial example is the burst representation of the packet direction, it cannot be directly fed into the original model. The operation that decodes the burst representation cannot be executed in parallel, which makes train the proposed GAN slow. We train a substitute WF model that takes burst-based feature as input.

C. Loss Function

First, we describe the loss function of the discriminator. The discriminator is supposed to classify examples from T as positive while examples from S as negative. Thus, it maximize loss (1):

$$\mathcal{L}_D = \mathbb{E}_{x\sim T}\left[\log D(x)\right] + \mathbb{E}_{x\sim S}\left[\log(1 - D(x))\right] \quad (1)$$

Three items compose the loss function of the generator. Firstly, the key object of the generator is $F_{x\sim S}(x) \neq F_{x\sim S}(x+G(x))$. We refer C&W attack [22] and define the first loss as the difference between the logits value (before softmax function) of the orginal label $t = F(x)$ and the maximum logits value among all labels except t, which are calculated as

$$\mathcal{L}_{adv} = \mathbb{E}_{x'\sim T'}\left(\nabla F\left(x'\right)_t - \max_{i\neq(t=F(x))}\left(\nabla F\left(x'\right)_i\right)\right)^{+} \quad (2)$$

The second term is from the discriminator. The adversarial example is evaluated by the (3).

$$\mathcal{L}_{fake} = \mathbb{E}_{x'\sim T'}\left[\log\left(1 - D(x')\right)\right] \quad (3)$$

loss (1) and (3) can be directly used as the accuracy of the discriminator and the probability of fake target traffic $x + G(x)$ identified by the discriminator, respectively. We refer to AdvGAN use LSGAN [25] which loss is the least square between real label and the predicted label of a batch examples which set belongs to.

In addition, we choose L_1 distance to restrict the extent of manipulation. L_1 distance shows a better performance than L_2 distance.

$$\mathcal{L}_{dist} = \mathbb{E}_{x\sim S}\left\|G(x)\right\|_1 \quad (4)$$

In total, we combine these three functions as the target function of the generator. Moreover, α and β is used as a hyperparameters to adjust the weight.

$$\text{minimize } \mathcal{L}_G = \mathcal{L}_{adv} + \alpha\mathcal{L}_{fake} + \beta\mathcal{L}_{dist} \quad (5)$$

IV. Dataset and Experiment Setting

We evaluate the proposed method on the state-of-the-art WF attack, Deep Fingerprinting (DF) [5]. In [5], the authors provide a Tor traffic dataset and perform WF attack against two state-of-the-art defenses on this dataset, including Walkie Talkie [12] and WTF-PAD [11]. We therefore compare performances of the proposed dataset with performances of the two defenses on the dataset.

A. Dataset

DF dataset contains 95 website traffic traces collected from the top Alexa 100 website with 5 websites excluded. For each correctly responded website, 1000 instances are collected and each instance is a packet direction sequence and is represented by 5000-dimentional vectors.

DF dataset is divided into DF training set, WF-GAN training set and test set, which account for 80%, 10% and 10% of total instances respectively. The training set contains 76000 instances. Both WF-GAN training set and test set contain 9500 instances. The original feature set is used to train the sequence-based model, which is used to evaluate the effectiveness of adversarial examples generated by our defense. We transform the original sequence feature dataset to burst feature set to train WF-GAN and burst-based models.

B. Models

1) Sequence-based Model: The model of the DF attack is a Convolutional Neural Network (CNN) with well-designed architecture. Specifically, the input of the DF model is 5000 dimensions and output of the DF model is 95 dimensions units. It contains 4 blocks and each block is comprised of two convolutional layers with batch normalization layer for relieving overfitting. This model attains 98.2% accuracy on the original test set.

2) Normal Burst-based Model: It is used to conduct experiment on white-box scenarios. We transform the 5000-dimensional packet direction vector into 1024-dimentional burst vector, since the length of burst vector which is transformed from the packet directions sequence of the top 100 websites mostly are less than this value. Consequently, the input layer of the burst-based model consists of 1024 neurons and the other layer is identical to the original packet sequence-based model. The burst feature dataset is scaled to a smaller range to promote GAN training. The scale factor is set to 128 after a premilitary experiment. The burst-based model attains 96.2% on the original test set.

3) Weak Burst-based Model: It is used to conduct experiment on black-box scenarios. Due to the attacker's model is likely different with the targeted model used in WF-GAN, we simulate a scenario that an attacker uses a more sophisticated model than the defender. In fact, the defender typically has more prior knowledge which results in a more accurate model. In the WF-GAN training stage, the targeted model is intentionally simplified. In detail, the convolutional layers of the target model contain a half of convolutional kernel size of the original WF model. Specifically, 32, 64, 128 and 256 one-dimensional convolutional kernels reduce to 16, 32, 64 and 128, respectively, for 4 convolutional blocks of the original model. This model attains 95.8% accuracy on the original test set.

C. WF-GAN

We train WF-GAN based on the normal burst-based model and the weak burst-based model. The two WF-GAN is separately evaluate on the sequence-based model. We refer WF-GAN based on normal burst-model as white box scenario. While the WF-GAN based on weak burst-based model is referred as black box scenario. In the GAN training stage, the WF-GAN training set is used to train generator and discriminator.

D. Metric

1) Success Rate: It is used to evaluate the effect of the generated adversarial example for both targeted defense and untargeted defense. The success rate for untargeted defense

(SR-U) indicates the proportion of perturbed instances which is classified to any categories except the correct category. Walkie Talkie and WTF-PAD achieve roughly 50% and 10% SR-U on DF dataset, respectively. The success rate for targeted defense (SR-T) is calculated by the faction of perturbed instances whose label is classified to the target set classes.

2) Overhead: It is used to evaluate the degree of the modification to original traffic. We define overhead as the number of inserted dummy packets which is same to Walkie Talkie and WTF-PAD. In DF dataset, the average packet of train set and test set is 1842 and 1841, respectively. We use 1800 to compute the the overhead percentage. The overhead percentage indicates the incremental proportion of overhead on a specific defense setting. For Walkie Talkie and WTF-PAD, two methods achieve roughly 33% and 60% overhead according to [5].

V. Experimental Result

In this section, we evaluate the performance of the proposed approach on the stat-of-the-art WF attack under various source set size setting and various threshold setting. We train WF-GAN for 300 epochs with loss factor $\alpha = 1$ and $\beta = 3$.

A. Evaluation on S/T Ratio

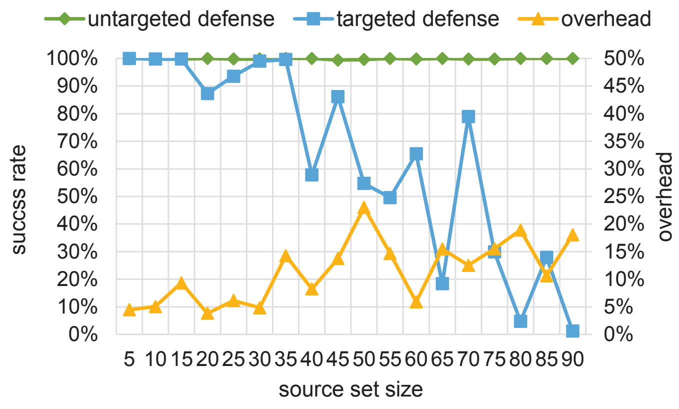

Fig. 2. The defense performance on the burst-based model with different source set size setting

We use different source set size to simulate different S/T ratio. The source set size from 5 to 90, and corresponding target set size from 90 to 5. Fig. 2 shows the defense effectiveness and the overhead of the generator trained with various source set sizes on the burst-based model. The defended website traffic exhibits nearly 100% SR-U across all source set size settings. In case of targeted defense, the SR-T is above 90% as the source set size less than 35. As the number of the source websites increasing, targeted defense shows unstable defense performance. When the source set size ranges from 40 to 60, the SR-T reduce to roughly 60%. When the source set size is greater than 60, the SR-T reduces to 30%.

Fig. 3 illustrates the WF-GAN defense performacne on the sequence model (i.e. original model) at white-box scenarios. The result shows that the adversarial example generated by

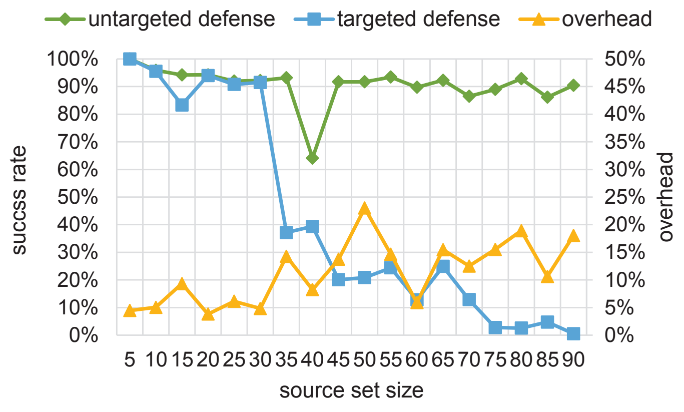

Fig. 3. The defense performance on the sequence model with different source set size setting. WF-GAN is trained using a burst model whose architecture is identical to the sequence model (white box scenario).

WF-GAN is effective for both targeted defense and untargeted defense. Specifically, in case of source set size less than 35 and target set size more than 60, which means the number of targeted websites is two times more than the number of source websites, WF-GAN achieves roughly 90% success rate on both targeted defense and untargeted defense.

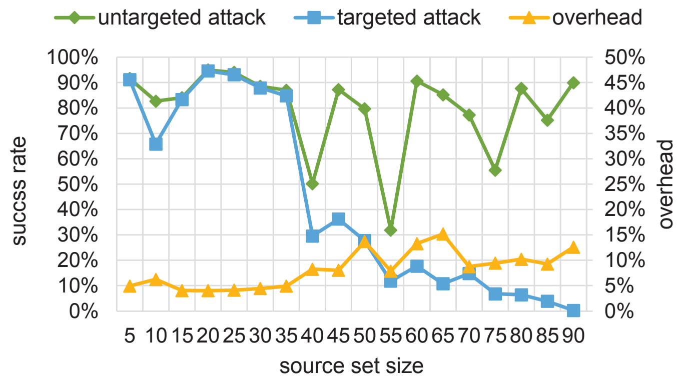

Fig. 4. The defense performance on the sequence model with different source set size setting. WF-GAN is trained using a weak burst model whose architecture is different with the sequence model (black box scenario).

Fig. 4 illustrates the defense performance of WF-GAN trained based on the weak target model. As the S/T ratio less than 0.5, WF-GAN exhibits a similar defense success rate compared to WF-GAN trained based on full convolutional kernel model. While the S/T ratio greater than 0.5, the success rate of untargeted defense sharply fluctuates for different settings. In many cases, our defense achieves 80% success rate.

For the overhead performance, we set the factor of perturbation distance loss β to 5. Compared with $\beta = 5$, the overhead of the generated adversarial example is more stable and slightly lower. WF-GAN achieves roughly 5% overhead when the S/T ratios less than 0.5. Otherwise, WF-GAN achieves about 10% overhead.

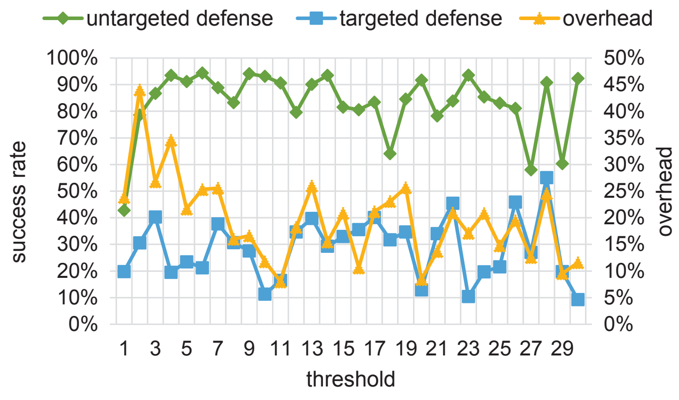

Fig. 5. The defense performance on the sequence model for different threshold setting. The source set size is 50. WF-GAN is trained using a burst model whose architecture is identical to the sequence model (white box scenario).

B. Evaluation on Threshold

In WF-GAN, the threshold is an important hyperparameter. It constrains the number of dummy packets for each burst.Formally, it clips the Δx by L_∞ norm . In order to evaluate the best range of threshold, we conduct an experiment to evaluate the performance of WF-GAN on various threshold settings.

Fig. 5 shows the defense performance under thresholds that ranges from 1 to 30. We choose 50 websites as source set and 45 websites as target set in this experiment. When threshold is more than 1, WF-GAN can reach over 80% success rate of untargeted defense. When threshold more than 4, WF-GAN can reach over 90% success of untargeted defense. However, the SR-T is between 20% and 40% as threshold increasing. As the threshold increases, the proportion of overhead decreases. In our dataset, WF-GAN reach the lowest overhead at a threshold of 11 for WF defense.

C. WF-GAN loss analysis

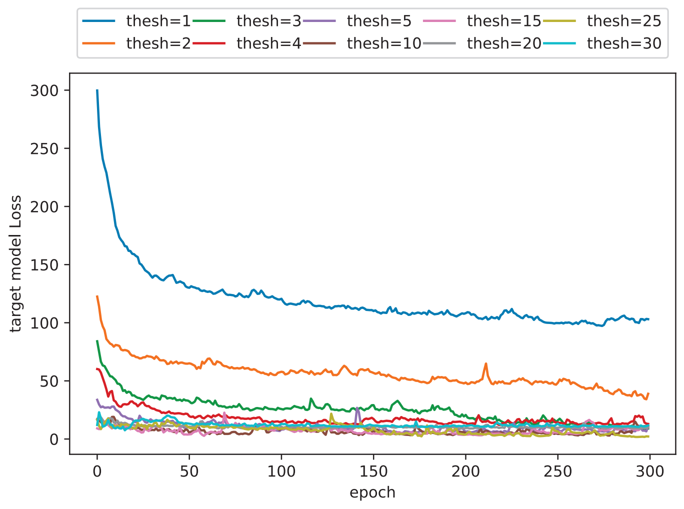

Fig. 6. The comparison of $\mathcal{L}_{adv}$ descent curves at different threshold settings. The epoch number means the number of training steps that all training data are feed to model once time.

We explain the effect for different threshold settings through loss $\mathcal{L}_{adv}$ which is derived from the target model. Fig. 6 gives the representation of $\mathcal{L}_{adv}$ during the 300 training epochs for thresholds from 1 to 4 and thresholds from 5 to 30 by step 5. As the figure shown, when the threshold is 1, $\mathcal{L}_{adv}$ reduces from 300 to 150 at initial epochs but slows down in the subsequent training epochs. The result of WF-GAN on threshold 2 is similar to the result of WF-GAN on threshold 1, whose $\mathcal{L}_{adv}$ reduced to the limitation of roughly 50. $\mathcal{L}_{adv}$ reaches to the vicinity of the optimal value by increasing the threshold. Except threshold of 1 and 2, the minimal value of $\mathcal{L}_{adv}$ achieved by WF-GAN exhibits very limited difference.

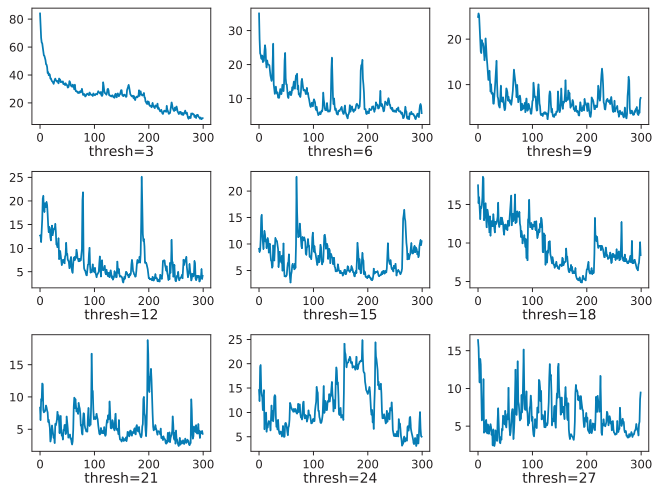

Fig. 7. The curve of $\mathcal{L}_{adv}$ with respect to training epoch number for different threshold settings. The horizontal axis represents the number of each epoch. The vertical axis represents $\mathcal{L}_{adv}$.

Fig. 7 presents the details of $\mathcal{L}_{adv}$ decent curve for threshold more than 2. In the subfigure threshold 3 and 6, as the number of training epoches increases, the loss decreases significantly. However, as the threshold continues to increase, the amplitude of the loss fluctuations increases. This is caused by the parameter of the generator is affected by the other two losses (i.e. $\mathcal{L}_{dist}$ and $\mathcal{L}_{fake}$) after $\mathcal{L}_{adv}$ reaches a small range. As these subfigures show, $\mathcal{L}_{adv}$ mostly concentrate on the range from 5 to 15. When the threshold is more than 12, the fluctuation makes the generator stop at a value that is non-minimal $\mathcal{L}_{adv}$. For instance, threshold 24 and 27 settings exhibit a loss increasing as $\mathcal{L}_{adv}$ reaching a lowest value. The large threshold value is likely to lead a poor performance, especially for WF-GAN trained based on a black box model. Therefore, we choose threshold 10 as the default hyper-parameter value in previous experiments.

Fig. 8 show the training curve of generator and discriminator for different source set size range from 15 to 90. The GAN loss of generator can be seen as the possibility of being identified as a fake target website by the discriminator. In the subfigures of 15 source websites and 30 websites, as the capacity of the discriminator increases, the generator loss does not increase. At the 45 and 60 source websites settings, approximately a

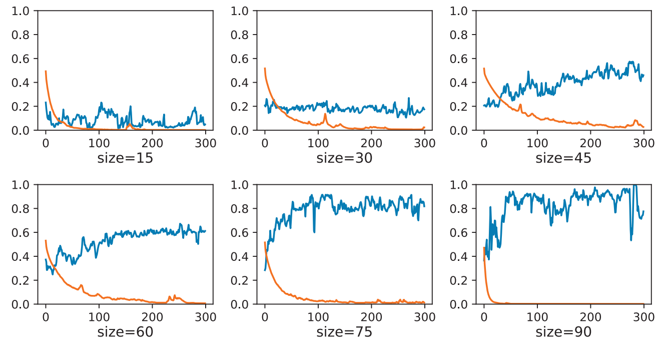

Fig. 8. The generator loss ($\mathcal{L}_{fake}$)and discriminator loss curve ($\mathcal{L}_D$) with respect to training epoch number for different source set size settings. The horizontal axis represents the number of each epoch. The vertical axis means the loss of generator and discriminator, which are represented by blue line and orange line, respectively.

half of adversarial examples are classified as the websites from source websites at the end of training. Finally, almost all adversarial examples are identified by the discriminator when the source website set is twice more than the target website set.

VI. Conclusion

In this paper, we proposed WF-GAN, a generative adversarial network for defending website fingerprinting attack based on deep neural networks. WF-GAN support both untargeted defense and targeted defense. Previous defense methods mostly consider reducing the accuracy of the WF attack on the original traffic, which corresponds to our untargeted defense. We evaluate WF-GAN on both white box scenarios and black box scenarios. The result show that our proposed defense is effective defend encrypted traffic from WF attack with lower overhead than previous defense.

Acknowledgement

This work is supported by The Key research and Development Program for Guangdong Province under grant No. 2019B010137003 and The National Key Research and Development Program of China (No.2016QY05X1000, No. 2018YFB1800200 and No. 2020YFE0200500).

References

[1] E. Petagna, G. Laurenza, C. Ciccotelli, and L. Querzoni, “Peel the onion: Recognition of android apps behind the tor network,” in *Information Security Practice and Experience - 15th International Conference, ISPEC 2019, Kuala Lumpur, Malaysia, November 26-28, 2019, Proceedings*, ser. Lecture Notes in Computer Science, vol. 11879. Springer, 2019, pp. 95–112.

[2] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, “Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail,” in *2012 IEEE symposium on security and privacy*, 2012, pp. 332–346.

[3] J. Hayes and G. Danezis, “k-fingerprinting: A robust scalable website fingerprinting technique,” in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 1187–1203.

[4] V. Rimmer, D. Preuveneers, M. Juárez, T. van Goethem, and W. Joosen, “Automated website fingerprinting through deep learning,” in *NDSS*, 2018.

[5] P. Sirinam, M. Imani, M. Juarez, and M. Wright, “Deep fingerprinting: Undermining website fingerprinting defenses with deep learning,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1928–1943.

[6] S. Bhat, D. Lu, A. Kwon, and S. Devadas, “Var-cnn: A data-efficient website fingerprinting attack based on deep learning,” *PoPETs*, vol. 2019, no. 4, pp. 292–310, 2019.

[7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” *CoRR*, vol. abs/1312.6199, 2013.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *CoRR*, vol. abs/1412.6572, 2014.

[9] J. Jia and N. Z. Gong, *Defending Against Machine Learning Based Inference Attacks via Adversarial Examples: Opportunities and Challenges*. Springer International Publishing, 2020, pp. 23–40.

[10] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, and K. Wehrle, “Website fingerprinting at internet scale.” in *NDSS*, 2016.

[11] M. Juarez, M. Imani, M. Perry, C. Diaz, and M. Wright, “Toward an efficient website fingerprinting defense,” in *European Symposium on Research in Computer Security*, 2016, pp. 27–46.

[12] T. Wang and I. Goldberg, “Walkie-talkie: An efficient defense against passive website fingerprinting attacks,” in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 1375–1390.

[13] S. Li, H. Guo, and N. Hopper, “Measuring information leakage in website fingerprinting attacks and defenses,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 1977–1992.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[15] C. Xiao, B. Li, J. Zhu, W. He, M. Liu, and D. Song, “Generating adversarial examples with adversarial networks,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 2018, pp. 3905–3911.

[16] M. Ring, D. Schlör, D. Landes, and A. Hotho, “Flow-based network traffic generation using generative adversarial networks,” *Computers & Security*, vol. 82, pp. 156 – 172, 2019.

[17] Z. Lin, Y. Shi, and Z. Xue, “IDSGAN: generative adversarial networks for attack generation against intrusion detection,” *CoRR*, vol. abs/1809.02077, 2018.

[18] M. Rigaki and S. Garcia, “Bringing a GAN to a knife-fight: Adapting malware communication to avoid detection,” in *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*. IEEE Computer Society, 2018, pp. 70–75.

[19] P. Wang, S. Li, F. Ye, Z. Wang, and M. Zhang, “Packetcgan: Exploratory study of class imbalance for encrypted traffic classification using CGAN,” *CoRR*, vol. abs/1911.12046, 2019.

[20] J. Li, L. Zhou, H. Li, L. Yan, and H. Zhu, “Dynamic traffic feature camouflaging via generative adversarial networks,” in *7th IEEE Conference on Communications and Network Security, CNS 2019, Washington, DC, USA, June 10-12, 2019*. IEEE, 2019, pp. 268–276.

[21] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018*, 2018.

[22] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57.

[23] M. Imani, M. S. Rahman, N. Mathews, and M. Wright, “Mockingbird: Defending against deep-learning-based website fingerprinting attacks with adversarial traces,” 2019.

[24] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 5967–5976.

[25] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “Least squares generative adversarial networks,” in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2813–2821.