Towards an Efficient Defense against Deep Learning based Website Fingerprinting

Zhen Ling[§], Gui Xiao[†], Wenjia Wu[§], Xiaodan Gu[§], Ming Yang[§*], and Xinwen Fu[‡]
[§]School of Computer Science and Engineering, Southeast University, China
[†]School of Cyber Science and Engineering, Southeast University, China
Email: {zhenling, xiaogui, wjwu, xdgu, yangming2002}@seu.edu.cn
[‡]Department of Computer Science, University of Massachusetts Lowell, Lowell, MA, USA
Email: xinwen_fu@uml.edu

IEEE INFOCOM 2022 - IEEE Conference on Computer Communications | 978-1-6654-5822-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/INFOCOM48880.2022.9796685

***Abstract*—Website fingerprinting (WF) attacks allow an attacker to eavesdrop on the encrypted network traffic between a victim and an anonymous communication system so as to infer the real destination websites visited by a victim. Recently, the deep learning (DL) based WF attacks are proposed to extract high level features by DL algorithms to achieve better performance than that of the traditional WF attacks and defeat the existing defense techniques. To mitigate this issue, we propose a-genetic-programming-based variant cover traffic search technique to generate defense strategies for effectively injecting dummy Tor cells into the raw Tor traffic. We randomly perform mutation operations on labeled original traffic traces by injecting dummy Tor cells into the traces to derive variant cover traffic. A high level feature distance based fitness function is designed to improve the mutation rate to discover successful variant traffic traces that can fool the DL-based WF classifiers. Then the dummy Tor cell injection patterns in the successful variant traces are extracted as defense strategies that can be applied to the Tor traffic. Extensive experiments demonstrate that we can introduce 8.1% of bandwidth overhead to significantly decrease the accuracy rate below 0.4% in the realistic open-world setting.**

***Index Terms*—Anonymous communication systems, website fingerprinting, cover traffic**

I. Introduction

Since user concerns over privacy continuously rise, anonymous communication systems are pervasively employed by hundreds of thousands of users all over the world to protect their communication privacy. The Onion Router (Tor) becomes the most popular anonymous communication system due to its strong anonymity protection capability and well user experience design. However, various traffic analysis techniques against Tor are investigated to de-anonymize users' privacy. The *Website fingerprinting* (WF) attack is such a single-end attack [34] that allows a single local and passive attacker (e.g., a local network administrator or an Internet service provider (ISP)) to passively record and analyze the encrypted network traffic between users and Tor network so as to infer websites visited by the users.

To effectively perform the WF attack, the attacker first leverages the encrypted network traffic profile, like "fingerprints", to train an appropriate machine learning classifier and then employ the classifier to predict victims' visited website using their traffic. Although a Tor client multiplexes multiple flows into a single TCP connection and packs user data into fixed-size transmission units (i.e., Tor cells) in an attempt to eliminate the traffic profile, the sophisticated traffic feature engineering and diverse machine learning algorithms are deeply studied to find the effective traffic features and train robust classifiers to infer the real destination websites visited by the users. The accuracy of traditional WF attacks mainly relies on the selection of features and classifiers. Specifically, the effective handcrafting traffic features (i.e., packet size distribution, traffic burst, and packet timing interval) are extracted based on expert knowledge. In addition, the traditional machine learning classifiers such as Support Vector Machine (SVM) [6], [20], [21], k-Nearest Neighbors (k-NN) [31], and random forest [11] are studied to achieve more than 90% accuracy against Tor. To mitigate the WF attacks, various defense techniques [2], [4], [5], [8], [9], [14], [19], [23], [27], [33] that introduce dummy packets and/or packet delay are proposed to eliminate the distinguishable traffic features so as to defend against the WF attacks.

Recently, many efficient deep learning (DL) approaches have proven successful in various areas [16], [26]. In addition, DL models are leveraged to extract the features for WF attacks [3], [24], [25], [28], [29]. Since the high level features extracted by DL-based WF classifiers with sophisticated architectures are more robust than the handcrafting features used by the traditional machine learning WF classifiers, it is more difficult to design an effective defense method based on intuition and expert knowledge to eliminate high level traffic features. Furthermore, confronted with the DL-based WF attacks, the existing defense techniques [2], [14], [23], [33] are ineffective and incur too much bandwidth overhead and/or latency overhead.

In this paper, we leverage Tor dummy cells to design a-genetic-programming-based variant cover traffic search technique to fight against the DL-based WF attacks [25], [28]. We first collect a number of labeled traffic traces of each website that can be correctly identified by the classifiers and construct an initial population. Then we perform multiple mutation operations on each trace in the population by randomly choosing positions in the trace and injecting Tor dummy cells. We attempt to extract the high level features by employing the feature maps of the DL-based WF classifier [28] that is able to achieve the high performance. Subsequently, we deliberately design a fitness function to compute the feature distance and evaluate the generated variant cover traffic traces

* Corresponding author: Prof. Ming Yang of Southeast University, China.

so as to find successful variant traces that can be misclassified by the DL-based WF classifiers. The mutation direction control mechanism is proposed by leveraging the fitness function and a sliding window to effectively generate variant traces. By repeating the search process, the successful variant traces of all the websites can be discovered. The dummy cell injection patterns (i.e., injection positions and directions in the successful variant cover traffic traces) extracted from successful variant traces then can be recorded as the defense strategies applied to the traffic between the users and Tor on the fly. To evaluate the feasibility and effectiveness of our approach, we perform extensive experiments in both closed-world and open-world settings to validate the defense strategies against the DL-based WF classifiers, including Convolutional Neural Network (CNN) [25], Stacked Denoising Autoencoders (SDAE) [25], Deep Fingerprinting (DF) [28].

Our major contributions are summarized as follows.

- To the best of our knowledge, we are the first to leverage the genetic programming search technique to explore the effective and efficient defense strategies to fight against the DL-based WF attacks. Stochastic mutation operations are performed on the original labeled Tor traffic traces by injecting a small number of Tor dummy cells to generate variant cover traffic traces. Then the successful variant traces are selected to produce the defense strategies.
- We employ the feature maps of the classifier that achieves the high performance to design a fitness function used to calculate the feature distance between the traces and a pre-selected least similar target website. On the basis of the fitness function, we propose a mutation direction control mechanism so as to quickly find a successful variant cover traffic trace for each website that can mislead the latest WF classifiers.
- We leverage a well known large dataset to construct an initial population, including 80,000 traces of 200 websites. We evaluate the feasibility and efficiency of our approach against the state-of-the-art DL-based WF classifiers in both closed-world and open-world settings. The experimental results demonstrate that we can significantly decrease the accuracy of the DL-based WF classifiers to around 0.4% by introducing only below 8.1% of bandwidth overhead. Moreover, our approach can be applied to defend against DL-based traffic analysis attacks to protect the communication privacy.

The rest of this paper is organized as follows. We present anonymous communication network and the website fingerprint attack in Section II. In Section III, we introduce the genetic programming based variant cover traffic search technique, including the basic idea and the detailed design of our system. Then we conduct extensive empirical experiments to demonstrate the feasibility and effectiveness of the search technique in Section IV. We review related work in Section V. Finally, we conclude this paper in Section VI.

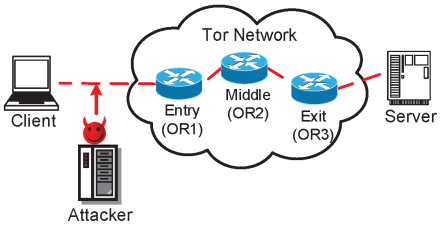

Fig. 1. Website fingerprinting attack

II. Background

In this section, we briefly introduce the anonymous communication network and the WF attack.

A. Anonymous communication system

Anonymous communication systems, such as Tor, are designed to protect users' communication privacy from attackers. Without loss of generality, we take the multi-hop low-latency anonymous communication system Tor as an example to show how it preserves the users' communication privacy. A user first installs a Tor client that provides a local proxy for applications such as a browser on her computer. To anonymously communicate with a remote web server, the Tor client creates three hop paths, referred to as circuits, and the data is encrypted and packed into fixed-size Tor cells. Then the Tor cells are relayed between the browser and the remote web server via the circuits. The first-hop Tor node and the third-hop one in a circuit are referred to as the entry node and the exit node respectively. Since the exit node communicates with the server on behalf of the user, the server cannot learn the real IP address of the user. Moreover, since a network eavesdropper at the user side sitting between the Tor client and the entry node can only observe the destination IP address of the entry node, she cannot determine the real destination server visited by the user. In this way, the user can anonymously communicate with the remote web server.

B. Website Fingerprinting Attack

To de-anonymize users' communication privacy, a type of single-end attacks [34], referred to as WF attacks, is investigated to infer the websites that are accessed via Tor by victims. To this end, an attacker first accesses a list of websites through Tor and collects the network traffic traces between her computer and the Tor entry node. A network traffic trace labeled as its corresponding website consists of a sequence of network packets. Then the attacker deeply analyzes the labeled traces so as to derive the handcrafted features (e.g., packet length, packet timing, and traffic burst) and uses a supervised machine learning method (e.g., SVM [6], [20], [21], k-NN [31], and random forest [11]) to train a classifier. Eventually, the attacker is able to extract the features from a captured unlabeled trace of a victim and leverage the output scores from the WF classifiers to infer the visited websites. To conduct the WF attack, the attacker could either be a malicious local network administrator or own a malicious entry node so as to passively sniff the network traffic traces between the victim and the Tor entry node as shown in Figure 1.

Recent researches on WF attacks [25], [28] demonstrate that DL-based approaches outperform the traditional machine learning based ones. The WF attacks [25] using SDAE, CNN, and LSTM can automatically extract more complex high level features from the traces for the classifiers. The DL-based WF attack [28] DF, using a CNN variant with a more sophisticated architecture, is studied to demonstrate that it can achieve better performance than that of the traditional approaches. Moreover, since the features are extracted by the DL algorithm, it is nontrivial to understand the features, let alone defend against the DL-based WF attacks. In fact, the DL-based WF attack [28] can effectively defeat the state-of-the-art defense techniques including WTF-PAD [14] and Walkie-Talkie [33].

The performance evaluation methods of the WF attack can be categorized into two classes: *closed-world* and *open-world*. In the closed-world setting, it is assumed that the user can only access a small number of websites that are on the attacker's monitored list. Then the attacker can train the classifier using a small dataset that only contains the traces from the monitored websites and identify if the user visits the specific websites on the list using a captured unlabeled trace. Although the closed-world setting is unrealistic [13], it is commonly used to assess the performance of distinct WF classifiers for comparison purpose. The open-world setting is more realistic. The user is allowed to access any websites whether they are on the monitored list or not. A standard open-world classifier [11], [20], [28], [31] is trained by using the traces collected from the monitored websites and a number of unmonitored websites, since the traces from unmonitored websites can help the classifier improve the performance. Then the attacker uses the classifier to determine whether a captured trace is from a monitored website or not.

III. Variant Cover Traffic Search Technique

In this section, we first introduce the basic idea of our variant cover traffic search technique. Then we elaborate on the critical designs of our method.

A. Threat model

In the website fingerprinting attack [25], [28], a *local* and *passive* attacker is assumed to be capable of inferring the websites visited by a user as shown in Figure 1. "local" indicates that the attacker can have access to the network link between the user and the Tor entry node, while "passive" indicates that the attacker can covertly record the network traffic between the user and the entry node without tampering the traffic. We assume that the attacker collects the labeled traffic and trains both the closed-world and open-world classifiers in advance. Then the DL-based WF classifiers, including CNN, SDAE [25], and DF [28], are deployed to inspect the traffic on the fly and perform the WF attacks.

B. Basic Idea

Our goal is to search effective and efficient positions and directions in labeled traffic traces of each website for injecting dummy traffic so as to produce variant cover traffic that can fool the DL-based WF classifier. In this paper, a sequence of positions and directions in the traffic traces used for injecting the dummy Tor cells is referred to as a *dummy cell injection pattern*. To search the effective and efficient dummy cell injection patterns, we employ the *genetic programming* search technique on the labeled traces so as to generate variant cover traffic traces by randomly performing a series of mutation operations on the original traces. We leverage the feature maps of the classifier that achieves the high performance to design a fitness function used to calculate the feature distance between the traces and a pre-selected least similar target website. On the basis of the fitness function, we propose a mutation direction control mechanism so as to quickly find a successful variant cover traffic trace for each website that can mislead the latest WF classifiers. Once a variant cover traffic trace is found, we extract the injection patterns and apply them to the traffic between the Tor client and the exit node on the fly so as to defend against the various DL-based WF attacks.

Figure 2 illustrates the workflow of our variant cover traffic search technique. We first elaborately collect a number of labeled network traffic traces of each website to build an initial population. It is assumed that the traffic traces are correctly classified by the target DL-based WF classifiers. To search effective and efficient dummy cell injection patterns, a series of mutation operations are conducted on each trace to select positions in the traces and inject Tor dummy cells in some direction (i.e., either from the Tor client to the exit node or the opposite direction). As a result, we can produce a generation of variant cover traffic traces. Then a mutation direction control mechanism and a fitness function are designed to use to direct the search and determine if a successful variant trace is found. Upon discovering the successful variant traces, we record the Tor dummy cell injection patterns that can be applied to the Tor traffic on the fly so as to prevent the visited websites from being identified by the DL-based WF classifiers. Finally, if the maximum number of generation is reached, we stop searching the injection patterns.

C. Population Initialization

We construct an initial population of the pre-processed network traffic traces for each website. The raw TCP traffic traces of a list of websites are collected between a user and a Tor entry node when the user employs a web browser to visit these websites via the Tor client. Then the raw traces are processed to extract the Tor cells using the method proposed by [32]. As a result, the pre-processed trace consists of a sequence of 1 and -1, where 1 represents a Tor cell emitted from the user to the website and -1 represents a Tor cell sent in the opposite direction. Denote the number of target websites as N that is determined by the attacker's classifiers. We elaborately choose n exemplar traces for each website to initialize the traces pool. Denote the exemplar trace set for the i^{th} website as $\mathcal{A}_i = \{x_i^1, x_i^2, \ldots, x_i^n\}$, where x_i^j is the j^{th} exemplar trace from the i^{th} website ($j = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, N$). If one of the traces is successfully classified to the corresponding website with the probability of over 90% by the DL-based WF

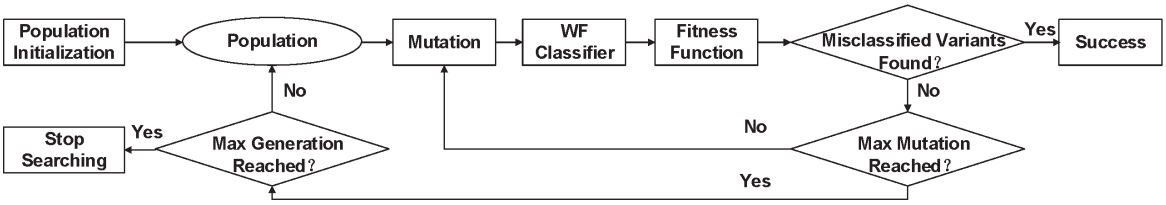

Fig. 2. Workflow of variant cover traffic search technique

attacks, it is chosen as an exemplar trace. Then we duplicate each exemplar trace m times to initialize the population. The purpose of the replication is to simultaneously find a trace that can successfully mislead the classifiers. Let $c_{i,j}^{q}$ $(q = 1, 2, , \ldots, m)$ be the q^{th} duplicate trace of the exemplar trace x_i^j. Finally, the size of the population is $N * n * m$.

After constructing the initial population, we randomly select one out of top k least similar target websites for each website so as to find a dummy cell injection pattern to transform the original traffic pattern of a website to that of its target website. To this end, we first generate a target pool for each website. Each target pool consists of top k least similar websites. We then leverage a representative feature vector to represent a website and compute the feature distances between feature vectors of any websites. We use the DF model [28] to compute the representative feature vectors for each website. Since this deep learning model is able to achieve over 98% accuracy on Tor traffic and outperforms other existing models, it is the best choice to use the DF model to extract representative deep features for each website. Let $\Phi(\cdot)$ be the feature maps of the DF model that are used to map a data vector to a feature vector. By using n exemplar traces for each website selected in the population initialization phase, we can calculate the average feature vector μ_i for the i^{th} website as follows:

$$\mu_i = \frac{1}{n} \sum_{j=1}^{j=n} \Phi(x_i^j) \tag{1}$$

We use the average feature vectors as the representative feature vectors for each website. Then we calculate feature distances between any websites as below:

$$D(\mu_i, \mu_i(t)) = \ell_2(\mu_i, \mu_i(t)) \tag{2}$$

where ℓ_2 is the norm distance and $\mu_i(t)$ is the average feature vector of a target website. On the basis of the derived feature distances, we choose the top k least similar websites for each website and randomly select a target website t $(t \neq i)$ from the target pool for the i^{th} website.

D. Mutation Operation

We perform mutation operations on each trace so as to produce the variant traces of each website. The mutation operation is a Tor dummy cell injection in a position of the trace, since the dummy traffic is an effective defense technique against various WF attacks. We leverage the Tor dummy cell to design three mutation operation strategies on the traces, including injecting a dummy Tor cell from the Tor client to the exit node (i.e., inject 1 into the trace), injecting a dummy Tor cell in the opposite direction (i.e., inject -1 into the trace), and injecting a dummy Tor cell in either direction (i.e., inject 1 or -1 into the trace). After one of the mutation operation strategies is selected, we randomly select a position in the trace each time and perform the mutation operation to produce a generation of new variant traces. We define a maximum number of injection, $\mathcal{M}_I$, for each trace to control the bandwidth overhead of our defense approach.

E. Fitness Function

A fitness function is used to design a mutation direction control mechanism to quickly find a variant trace that fool the WF classifiers. It outputs a feature distance between a feature vector of a variant trace and an average feature vector of a target website. Let $x_i^j(o)$ be the o^{th} mutation of the j^{th} trace of the i^{th} website. The fitness function in the closed-world setting is defined by

$$\mathcal{F}(x_i^j(o)) = D(\Phi(x_i^j(o)), \mu_i(t)) \tag{3}$$

In the open-world setting, there are $N+1$ websites where the $(N+1)^{th}$ website corresponds to the unmonintored website. The fitness function in the open-world setting is defined by

$$\mathcal{F}'(x_i^j(o)) = D(\Phi(x_i^j(o)), \mu_{N+1}) \tag{4}$$

μ_{N+1} is the average feature vector of $N * n$ exemplar traces selected from the unmonitored websites. If the variant trace of monitorted website can be misclassified as an unmonitored website, the variant trace is claimed to evade successfully.

F. Mutation Direction Control

To effectively and efficiently find the variant trace, we design a mutation direction control mechanism to guide the search direction. In essence, the genetic programming is a search-based optimization algorithm. We leverage the mutation operation strategies to perform a search for a successful injection pattern so as to find an appropriate variant trace that can be misclassified. To direct the search, we use a sliding window and the feature distance to determine whether the search direction is correct or not. Therefore, if the search is not efficient, we can stop the search. Denote the size of the sliding window as $\mathcal{W}$. Intuitively, after performing mutation operations for $\mathcal{W}$ times, the feature distance should decrease at least $\frac{\mathcal{W}}{\mathcal{M}_\mathcal{I}} D(\Phi(x_i^j), \mu_t)$. In this way, we can guide the search direction to gradually decrease the feature distance between the feature vector of the variant trace and the average feature vector of the target website, ideally, to 0 before reaching the maximum number of mutation operations $\mathcal{M}_\mathcal{I}$. On the basis

of this intuition, we can have the constraint for directing the search in the closed-world setting as follows:

$$\mathcal{F}(x_i^j(h)) - \mathcal{F}(x_i^j(o)) \geq \frac{\mathcal{W}}{\mathcal{M}_{\mathcal{I}}} D(\Phi(x_i^j), \mu_t) \quad s.t. \;\; 0 < o - h < \mathcal{W} \tag{5}$$

Likewise, the constraint for directing the search in the open-world setting becomes

$$\mathcal{F}'(x_i^j(h)) - \mathcal{F}'(x_i^j(o)) \geq \frac{\mathcal{W}}{\mathcal{M}_{\mathcal{I}}} D(\Phi(x_i^j), \mu_{N+1}) \quad s.t. \;\; 0 < o - h < \mathcal{W} \tag{6}$$

Once the constraints of Equation (5) and (6) is satisfied within a sliding window, it indicates that the search direction is right and effective. Then we update the initial starting point of the sliding window to current mutation stage so as to ensure that the search direction is correct in each sliding window. Otherwise, we stop the search and then we put the trace back to the population waiting for the mutation operations on the next generation as presented in Section III-H.

G. Variant Cover Traffic Selection

After performing the Tor dummy cell injections, we should select successful variant traces that can confidently mislead the WF classifiers. If the search direction of current traces cannot satisfy the conditions of Equation (5), or the number of mutation operations reaches the maximum insertion value $\mathcal{M}_{\mathcal{I}}$, we should determine whether the WF classifiers can misclassify the variant cover traffic traces at this point. Denote the variant trace of a duplicate exemplar trace $c_{i,j}^q$ as $v_{i,j}^q$. Then the ratio of the feature distance between the variant trace and the target website to that between the original trace and the target website becomes

$$r = \frac{D(\Phi(v_{i,j}^q), \mu_t)}{D(\Phi(c_{i,j}^q), \mu_t)} \tag{7}$$

If the ratio r is smaller than a threshold T and the variant traces can fool the WF classifiers, the traces are chosen as the successful variant traces of the corresponding website. Once a duplicate trace $c_{i,j}^q$ of an exemplar trace x_i^j is able to mislead the WF classifiers, we stop the mutation for the exemplar trace x_i^j. Therefore, each exemplar trace has up to one successful variant trace. Upon discovering a successful variant trace, we remove the other duplicate traces of the corresponding exemplar trace from the population and record the dummy Tor cell injection patterns. If variant traces of all n exemplar traces of each website are successfully discovered or the maximum generation number is reached, we perform the variant cover traffic selection. To reduce the bandwidth overhead of our defense approach, we select one variant trace from these successful variant traces of a website that has the minimum number of injected Tor dummy cells and use the injection pattern for the corresponding website. If the maximum generation number is not reached, we put the variant traces back to the population again to wait for the mutation operations of the next generation. Finally, the successful injection patterns could be applied to the traces of the websites on the fly so as to keep the WF classifiers from identifying the traces as the corresponding websites.

H. Next-generation Population

The failed variant traces should be put back to the population to perform the mutation operations again. To increase the search speed, we build a injection pattern pool. The injection pattern pool contains the successful patterns and the promising patterns. Once a variant trace fails to fool the WF classifiers, we randomly select an injection pattern in the pool or select the original trace. If an injection pattern is chosen, it is applied to the original trace of the failed one and then the new variant trace is put back to the next-generation population. Otherwise, the original trace is directly put back to the population.

IV. Experimental Evaluation

We implement the genetic programming based variant cover traffic search prototype system. In this section, we perform extensive experiments using considerable computing resources to evaluate the feasibility and effectiveness of the genetic programming based variant cover traffic search technology using a well known large dataset.

A. Dataset

In our experiment, we employ a well known dataset collected by Rimmer *et. al.* [25] stating from January 2017. This dataset is commonly used by the various DL-based WF attacks and defenses for performance evaluation purposes. We use their dataset to construct a closed-world and an open-world dataset in our experiments.

Closed-world dataset (CW). To construct our closed-world dataset, we extract the 200 topmost popular websites from the Alexa top websites in the Rimmer's dataset [25]. The homepage of each website is visited 1000 times. Therefore, we have a total of $200,000$ (200*1000) traces in our closed-world dataset. We use 800 traces of each website, i.e., 200*800, to train the closed-world models, including CNN, SDAE, and DF.

Open-world dataset (OW). The open-world dataset contains a monitored website and an unmonitored website dataset. The data in the monitored dataset is from the closed-world dataset, i.e., $200,000$ traces from the top 200 websites. Since the unmonitored website dataset in the Rimmer's dataset [25] contains a single trace of each website from the top $400,000$ of Alexa websites, we extract $200,000$ traces from the top $200,000$ of Alexa websites in the dataset and construct our unmonitored website dataset. In total, we have $400,000$ traces in the open-world dataset. In the open world scenario, all unmonitored websites are regraded as the 201^{st} website. We use 750 traces of each website in the monitored websites, i.e., $150,000$ (200*750), and $150,000$ in the unmonitored websites to train the open-world models, i.e., CNN, SDAE, and DF.

The raw network traffic traces in the dataset are pre-processed using the method proposed by [32]. Consequently, a network traffic trace consists of sequences of 1 and -1 (1 for outgoing package to websites and -1 for incoming package from websites). According to the work [25], we use the first

TABLE I
DETECTION RATE (DR) ON THE NO-DEFENDED DATASET IN THE CLOSED-WORLD AND OPEN-WORLD SETTING

Models		CNN	SDAE	DF
DR	CW	95.96%	98.75%	**99.99%**
	OW	92.19%	96.82%	**99.05%**

3,000 data of a trace as the input of the CNN model to achieve the best performance. The input of the SDAE and DF model is the first 5,000 data of each trace to obtain the high performance as shown in the work [25] and [28]. Although the closed-world scenario is unrealistic [21], we can use them to evaluate the effectiveness of our defense approach against these four DL-based WF classifiers for comparison purpose.

B. Experimental Setup

In our experimental setting, we use 4 machines with 12 various NVIDIA GPU cards, including 4 GTX 1080 Ti cards, 5 GTX 3090 cards, 2 Tesla K80 cards, and 1 GTX Titan rtx card to verify the effectiveness of the our method.

We preprocess the dataset to construct an initial population and target pools, and calculate the average feature vectors of the 200 websites and the unmonitored website. To construct an initial population, we first elaborately choose 20 traces for each 200 topmost website from the closed-world dataset (i.e., the monitored website dataset in the open-world setting) and then duplicate each trace for 20 times. Recall that the selection principle of the 20 traces is that the probability of identifying the trace as the corresponding website is over 90%. Hence, the size of the initial population size is 80,000. Then, the feature maps of the DF model are used to calculate the average feature vectors using 20 traces of each website. 10 least similar websites are selected by comparing the feature distances between the average feature vectors of any website so as to construct a target pool for each website. Then we randomly select a target website from the target pool of each website to increase the effectiveness of our defense. The average feature vectors of each target website is used in the fitness function for the mutation direction control purpose. Likewise, we select $4,000$ traces in the monitored website dataset to computer the average feature vector of the unmonitored website. In addition, we empirically set the maximum generation number to 10,000. If the maximum generation number is reached, we stop searching the injection patterns.

The metrics used to evaluate our defense approach are the *bandwidth overhead (BO)* and the *detection rate (DR)*. The bandwidth overhead indicates the ratio of the number of Tor dummy cells injected between the Tor client and the exit node to the total of cells in the traces used by the WF classifiers. The detection rate indicates whether the traces using our injection patterns can be correctly identified by these DL-based WF classifiers. The lower the detection rate with little bandwidth overhead is, the better performance our defense method achieves.

C. Experimental results

We evaluate the existing CNN, SDAE, and DF based WF attacks using the closed-world and open-world dataset as a baseline of our method. The performance of the three WF attacks are shown in Table I.

As we can see from the table, in the closed-world scenario, the CNN, SDAE, and DF model are able to achieve high accuracy 95.96%, 98.75%, and 99.99%, respectively. While the performance of the DF model outperforms that of the CNN and SDAE model on no-defended dataset. Compared with the closed-world scenario, the detection rate decreases due to the significantly increased data size in the open-world scenario. Similar to the closed-world scenario, the detection rate of the DF model is higher than that of the other two models.

To evaluate the effectiveness and efficiency of our injection patterns, we use 1,000 labeled traces of each website from the closed-world dataset and apply the injection patterns to these traces. Then we have 200,000 (200*1,000) variant traces and use three models [25], [28], i.e., SDAE, CNN, and DF model, to evaluate the detection rate of our injection defense patterns in the closed-world and open-world setting for comparison . We set the threshold T of the r in Equation (7) as 10%.

In the closed-world and open-world setting, we conduct comprehensive experiments in an attempt to find the appropriate sliding window size $\mathcal{W}$ and the maximum number of injection $\mathcal{M}_{\mathcal{I}}$ of the three injection patterns to decrease both the bandwidth overhead and the detection rate. Since the input size of the CNN model is 3000 and the input size of the SDAE and DF model is 5000, we set the maximum number of injections $\mathcal{M}_{\mathcal{I}}$ as 1000, 1200, and 1400 for the three models to control the bandwidth overhead of our defense.

Given the maximum number of injected Tor dummy cells, Figure 3 illustrates the correlation between the detection rate of the CNN model and sliding window sizes in the closed-world setting. The detection rate of the CNN model decreases when the sliding window size increases. As a matter of fact, the number of average injected dummy cells in the successful injection patterns can increase by using greater sliding window size and maximum number of injected Tor dummy cells. Moreover, when 1 is injected, the lowest detection rate of the CNN model is 1.4% using the sliding window 500 and the maximum injection number 1400. In other words, we can obtain the most efficient injection patterns by injecting 1.

Figure 4 depicts the detection rate of the SDAE model in the closed-world setting in light of the sliding window size and maximum number of injected Tor dummy cells, respectively. As demonstrated in the figure, the lowest detection, i.e., 1.4%, is achieved by injecting 1 with the maximum number of injected cells of 1400 and the sliding window size of 600. In addition, when 1 or -1 is injected and the maximum number of injected cells is 1400, the best detection rate of the SDAE is 14.3% detection rate using the sliding window size as 600. Therefore, the performance of the injection pattern against SDAE by injecting 1 is much better than that of the injection patterns by injecting -1 and injecting 1 or -1.

The correlation between the detection rate of the DF model and sliding window sizes is demonstrated in Figure 5. When 1 is injected and the maximum number of injected cells is 1400, the detection rate of the DF model can be reduced to 1.7%

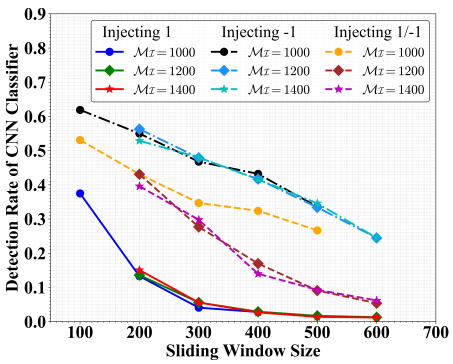

Fig. 3. Detection rate of the CNN model using three different mutation strategies versus sliding window sizes in the closed-world setting

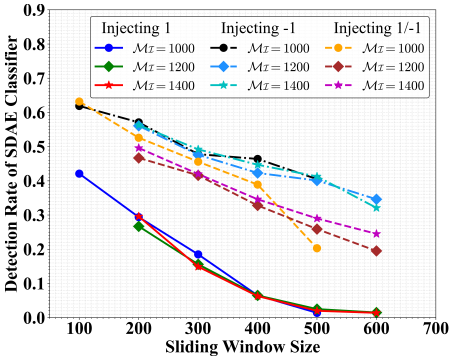

Fig. 4. Detection rate of the SDAE model using three different mutation strategies versus sliding window sizes in the closed-world setting

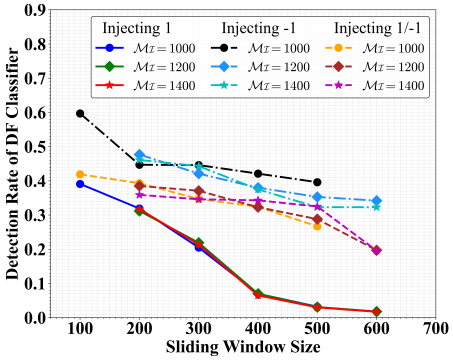

Fig. 5. Detection rate of the DF model using three different mutation strategies versus sliding window sizes in the closed-world setting

TABLE II

BANDWIDTH OVERHEAD (BO) AND DETECTION RATE (DR) ON THE THREE INJECTION PATTERNS (IP) AGAINST THE THREE MODELS IN THE CLOSED-WORLD. $\mathcal{M}_\mathcal{I}$: THE MAXIMUM NUMBER OF INJECTION. $\mathcal{W}$: THE SLIDING WINDOW SIZE

Models / IP	CNN				SDAE				DF			
	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR
Injecting 1	1000	500	16.7%	1.7%	1000	500	10.7%	2.7%	1000	500	10.1%	3.2%
	1200	500	16.4%	1.6%	1200	600	12.1%	1.5%	1200	600	12.0%	1.8%
	1400	500	16.8%	**1.4%**	1400	600	14.2%	**1.4%**	1400	600	12.0%	**1.7%**
Injecting -1	1000	500	19.1%	33.6%	1000	500	11.9%	37.6%	1000	500	15.8%	39.6%
	1200	600	21.8%	**24.5%**	1200	600	13.8%	34.6%	1200	600	17.9%	34.2%
	1400	600	21.7%	24.9%	1400	600	13.9%	**32.1%**	1400	600	18.1%	**32.3%**
Injecting 1/-1	1000	500	17.2%	9.6%	1000	500	11.4%	20.3%	1000	500	10.9%	26.7%
	1200	600	20.5%	**5.4%**	1200	600	12.7%	15.3%	1200	600	12.4%	19.7%
	1400	600	20.3%	6.2%	1400	600	12.4%	**14.3%**	1400	600	14.0%	**19.3%**

using the sliding window size as 600. By using injecting 1 or -1 mutation strategy, we can still decrease the detection rate below 20% by introducing a little higher bandwidth overhead. Therefore, the strategy of injecting 1 is the best one to achieve the lowest detection rate.

Table II demonstrates the bandwidth overhead and detection rate on the three injection patterns against the CNN, SDAE, and DF model in the closed-world setting. When 1 is injected, the bandwidth overhead of the injected Tor cells in the three models is the lowest. However, the injected cell number is the largest when -1 is injected. Therefore, injecting 1 mutation strategy can obtain the most efficient injection patterns. We use the sliding window size as 500 and 600 to evaluate the bandwidth overhead as the detection rates using such sliding window sizes are much better than the others. Then, we calculate the average numbers of injected Tor dummy cells to obtain the bandwidth overhead. As illustrated in the table, when 1 is injected, the best detection rates are 1.7%, 1.4%, and 1.4% of the DF, SDAE, and CNN model, while the bandwidth overhead of the three models are only 12.0%, 14.2%, and 16.8%, respectively. Since the detection rate of the DF model outperforms that of the CNN and SDAE model using the no-defended dataset in Table I, the detection rate of the DF model is still higher than that of the CNN and SDAE model. Moreover, when 1 or -1 are injected, our defense method can fool the CNN model with the 5.4% detection rate and 20.5% bandwidth overhead. Although we use the feature maps of the DF model to compute the feature vectors, our defense method can be effectively applied to these three models. It implies that our method can achieve strong generalizability.

Figure 6 depicts the detection rate of the CNN model in the open-world setting in terms of the sliding window size and maximum number of injected Tor dummy cells. We can see that the smallest detection can achieve 0.1% by injecting 1 with the maximum number of injected cells of 1200 and sliding window size of 300. Compared with the closed-world setting, the detection rate of the CNN model in the open-world setting considerably decreases. In addition, when 1 or -1 is injected and the maximum number of injected cells is 1200, the detection rate of the CNN can reach 0.5% using the sliding window size as 600. It implies that the high level features extracted by the CNN model in the open-world setting may contain the website traffic pattern on both two directions, i.e., from the Tor client to the exit node and the opposite direction.

The correlation between the detection rate of the SDAE model and sliding window sizes is shown in Figure 7. When 1 is injected, the detection rate of the SDAE model can be the smallest, i.e., 0.1%, with the sliding window 500 and the maximum injection number 1400. Similar to the results of the CNN model, when 1 or -1 is injected, we can also derive a low detection rate of 2.3% with the sliding window 600 and the maximum injection number 1400. As a result, we can conclude

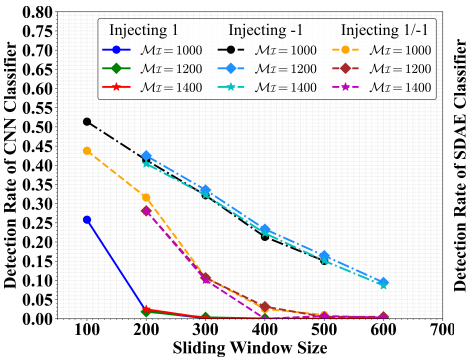

Fig. 6. Detection rate of the CNN model using three different mutation strategies versus sliding window sizes in the open-world setting

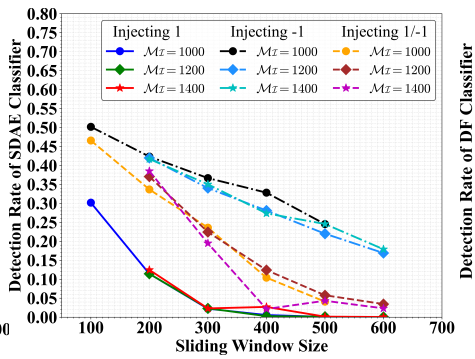

Fig. 7. Detection rate of the SDAE model using three different mutation strategies versus sliding window sizes in the open-world setting

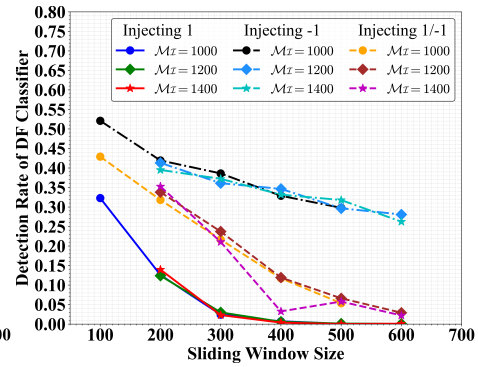

Fig. 8. Detection rate of the DF model using three different mutation strategies versus sliding window sizes in the open-world setting

TABLE III

BANDWIDTH OVERHEAD (BO) AND DETECTION RATE (DR) ON THE THREE INJECTION PATTERNS (IP) AGAINST THE THREE MODELS IN THE OPEN WORLD. $\mathcal{M}_\mathcal{I}$: THE MAXIMUM NUMBER OF INJECTION. $\mathcal{W}$: THE SLIDING WINDOW SIZE

Models / IP	CNN				SDAE				DF			
	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR	$\mathcal{M}_\mathcal{I}$	$\mathcal{W}$	BO	DR
Injecting 1	1000	300	10.3%	0.1%	1000	400	8.9%	0.6%	1000	400	8.1%	0.7%
	1200	300	10.2%	**0.1%**	1200	400	8.4%	0.3%	1200	400	8.0%	0.6%
	1400	300	10.1%	0.2%	1400	500	10.2%	**0.1%**	1400	500	8.1%	**0.4%**
Injecting -1	1000	500	17.1%	15.1%	1000	500	13.1%	24.5%	1000	500	13.4%	29.8%
	1200	600	20.3%	**9.5%**	1200	600	13.4%	**16.9%**	1200	600	14.7%	28.1%
	1400	600	20.4%	8.7%	1400	600	13.2%	17.9%	1400	600	15.1%	**26.2%**
Injecting 1/-1	1000	500	16.7%	1.0%	1000	500	11.3%	4.1%	1000	500	10.2%	5.4%
	1200	500	16.9%	**0.5%**	1200	600	12.1%	3.5%	1200	600	12.1%	2.9%
	1400	500	16.1%	0.7%	1400	600	12.4%	**2.3%**	1400	600	12.2%	**2.2%**

that the SDAE model in the open-world setting extracts the website traffic pattern on both two directions as well.

Figure 8 shows how the detection rate of DF model changes with the increase in sliding window sizes in the open-world setting. When 1 is injected, we can achieve the best detection rate, i.e., 0.4%. In addition, by injecting 1 or -1, the detection rate can reach 2.2% with the sliding window 600 and the maximum number of injection 1400.

Table III depicts the detection rate and bandwidth overhead on the three mutation strategies against the three models. It can be observed that we can achieve the best detection rates and bandwidth overhead by injecting 1. The best detection rates are 0.4%, 0.1%, and 0.1%, while the bandwidth overhead is only 8.1%, 10.2%, and 10.2% of the DF, SDAE, and CNN model, respectively. It indicates that obfuscating the features of the traffic patterns in the web page requests from the Tor client to the web server can effectively defeat the three DL-based WF attacks. Moreover, the detection rates of the CNN, SDAE, and DF model can significantly decrease to 0.5%, 2.3%, and 2.2% when injecting 1 or -1. We can conclude that the high level features extracted by the three models in the open-world setting use the website traffic pattern on both directions.

V. RELATED WORK

We are the first to explore whether the genetic programming based defense approach can defend against the start-of-the-art DL-based WF attacks. The Tor dummy cells are randomly injected into the labeled traces so as to generate the variant cover traffic traces that can mislead the DL-based WF classifier to make a wrong decision. Upon discovering the successful variant traces, the Tor dummy cell injection patterns are recorded to protect the Tor traffic on the fly from the DL-based WF attacks. Considerable empirical experiments are performed to demonstrate the effectiveness and efficiency of the successful injection patterns.

The traffic analysis techniques against Tor can be categorized into two groups [34]: single-end attacks and end-to-end attacks. The WF attack is a type of single-end attacks that allow a single attacker to sniff network traffic between a user and an anonymous communication system so as to infer the websites accessed by the user in light of the traffic profile. The first WF attack against Tor [12] is evaluated by Herrmann *et al.* Since undistinguishable features, i.e., packet length frequencies, are explored for identifying the website traffic, accuracy achieved in a closed-world setting is only 3%. Then Panchenko *et al.* [21] enhance the WF attack to 55% accuracy by exploiting new features and a SVM classifier. Substantially, Cai *et al.* [6] and Wang *et al.* [32] significantly improve the accuracy of the WF attack to 90% by using an edit-distance based SVM classifier.

New features and more advanced classifiers [11], [20], [31]

are proposed to further enhance the WF attack accuracy. For example, Wang *et al.* [31] employ a k-NN classifier and new features to reach 91% accuracy. Panchenko *et al.* [20] investigate a Radial Basis Function (RBF) kernel based SVM and features such as accumulated sum of the packet lengths to achieve 91% accuracy in the closed-world setting and 96% true positive rate as well as 1.9% false positive rate in the open-world setting respectively. Hayes and Danezis [11] achieve 91% accuracy in the closed-world setting and 88% true positive rate and 0.5% false positive rate in the open-world setting by using a random forest classifier and features such as packet ordering.

The first DL-based WF attack is proposed by Abe and Goto [1] to use a SDAE model and reach 88% accuracy in the closed-world setting and 86% true positive rate and 2% false positive rate in the open-world setting. Rimmer *et al* [25] explore three DL-based WF classifiers including SDAE, CNN, and LSTM and improve the DL-based attack to 96% accuracy in a closed-world setting. The state-of-the art DL-based WF attack DF [28] that outperforms the existing attacks is proposed by Sirinam *et al.* by using a variant CNN with a more complex architecture. Their WF attack can reach 98% accuracy in the closed-world setting and achieve 99% precision and 94% recall in the open-world setting. Furthermore, their DL-based WF attack can be still effective even when the defense techniques [14], [33] are applied to the Tor traffic. Moreover, Bhat *et al.* [3] present a deep CNN model Var-CNN based WF attack with reference to a ResNet based architecture. Var-CNN achieves the accuracy 98.8% with limited training data in the closed-world setting. TF [29] can reach 85% accuracy with a few train data leveraging triplet networks for N-shot learning.

Various defense methods are introduced to mitigate the WF attacks. Juarez *et al.* [14] propose the WTF-PAD using an adaptive padding strategy to protect Tor traffic against the WF attacks. The WTF-PAD defense method can drop the accuracy of the k-NN attack [31] from 92% to 17% with 60% bandwidth overhead. Walkie-Talkie (W-T) proposed by Wang and Goldberg [33] is based on half-duplex communication with a web server. It reduce the accuracy of the WF attacks to 50% with just 31% bandwidth overhead. Moreover, 34% latency overhead is also introduced due to the use of half-duplex communication. The DF-based WF attack [28], however, can achieve up to 90% accuracy against WTF-PAD in the closed-world setting and 98.4% top-2 accuracy against W-T. Then a new defense method, referred to as Mockingbird [23], is introduced to leverage GAN [7], [10] to generate adversarial examples and reduce the accuracy of the latest DL-based WF attacks from 98% to 38%-58% while incurring 58% bandwidth overhead. Abusnaina *et al.* [2] present the DFD defense based on injecting dummy Tor cells within every burst and decrease the accuracy of DL-based WF attacks to 13.98% with 14.43% bandwidth overhead.

The end-to-end attacks focus on confirming the communication relationship between the sender and the receiver via anonymous communication systems. The end-to-end attacks include active watermarking attacks [17], [18], [22], [35] and passive traffic analysis attacks [15], [36]. Since the active watermarking attacks can actively manipulate the traffic to embed a watermarking into the traffic so as to significantly improve the true positive rate and reduce the false positive rate, the watermarking attacks outperform the passive traffic analysis attacks. For example, Tian *et al.* [30] study how to de-anonymize the communication relationship using the Freenet.

VI. Conclusion

In this paper, to defend against the state-of-the-art DL-based attacks, we introduce a-genetic-programming-based variant covert traffic search technique to find successful cover traffic that can mislead the DL-based WF classifiers. We leverage a number of labeled traces from a well known dataset to construct initial population. Then multiple mutation operations are performed by injecting the Tor dummy cells into the traces so as to generate variant cover traffic traces. The feature distance-based fitness function in both closed-world and open-world settings is designed to select the successful variant traces that can fool the DL-based WF classifiers. Moreover, the mutation direction control mechanism is carefully investigated to direct the search in order to ensure the search efficiency. We repeat the procedure until the maximum generation is reached. After discovering all of the successful variant traces, the injection patterns are recorded and used for the Tor traffic on the fly between the Tor client and the exit node. Considerable empirical experiments are performed to demonstrate the feasibility and efficiency of our approach. In the open-world setting, the detection rate is 0.4% with just 8.1% bandwidth overhead against DF model. Further, in the closed-world setting, our defense approach achieves 1.7% detection rate with only 12.0% bandwidth overhead against the DF model. Our approach can also be used to fight against DL-based traffic analysis attacks to protect the communication privacy.

Acknowledgments

This research was supported in part by National Key R&D Program of China 2018YFB0803400, by National Natural Science Foundation of China Grant Nos. 62022024, 61972088, 62072103, 62102084, 62072102, 62072098, 61972083, and 62132009, by US National Science Foundation (NSF) Awards 1931871, and 1915780, US Department of Energy (DOE) Award DE-EE0009152, by Jiangsu Provincial Natural Science Foundation for Excellent Young Scholars Grant No. BK20190060, Jiangsu Provincial Natural Science Foundation of China under Grant No. BK20190340, Jiangsu Provincial Key Laboratory of Network and Information Security Grant No. BM2003201, Key Laboratory of Computer Network and Information Integration of Ministry of Education of China Grant Nos. 93K-9, and Collaborative Innovation Center of Novel Software Technology and Industrialization. Any opinions, findings, conclusions, and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] K. Abe and S. Goto. Fingerprinting attack on Tor anonymity using deep learning. *Proceedings of the Asia Pacific Advanced Network (APAN)*, 42:15–20, 2016.

[2] A. Abusnaina, R. Jang, A. Khormali, D. Nyang, and D. Mohaisen. DFD: Adversarial Learning-based Approach to Defend Against Website Fingerprinting . In *Proceedings of the 39th IEEE International Conference on Computer Communications (INFOCOM)*, pages 2459–2468, 2020.

[3] S. Bhat, D. Lu, A. Kwon, and S. Devadas. Var-CNN: A Data-Efficient Website Fingerprinting Attack Based on Deep Learning. *Proceedings on Privacy Enhancing Technologies (PET)*, 2019(4):292–310, 2019.

[4] X. Cai, R. Nithyanand, and R. Johnson. CS-BuFLO: A Congestion Sensitive Website Fingerprinting Defense. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES)*, pages 121–130, 2014.

[5] X. Cai, R. Nithyanand, T. Wang, R. Johnson, and I. Goldberg. A systematic approach to developing and evaluating website fingerprinting defenses. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 227–238, 2014.

[6] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson. Touching from a Distance: Website Fingerprinting Attacks and Defenses. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 605–616, 2012.

[7] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan. Generative Adversarial Networks: A Survey Towards Private and Secure Applications. *ACM Computing Surveys*, 2022.

[8] 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 *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 332–346, 2012.

[9] J. Gong and T. Wang. Zero-delay Lightweight Defenses against Website Fingerprinting. In *Proceedings of the USENIX Security Symposium (Security)*, pages 717–734, 2020.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. *Advances in neural information processing systems (NeurIPS)*, 27, 2014.

[11] J. Hayes and G. Danezis. k-fingerprinting: a Robust Scalable Website Fingerprinting Technique. In *Proceedings of the USENIX Security Symposium (Security)*, pages 1187–1203, 2016.

[12] D. Herrmann, R. Wendolsky, and H. Federrath. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial Naive-Bayes classifier. In *Proceedings of the ACM Workshop on Cloud Computing Security*, pages 31–42, 2009.

[13] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt. A critical evaluation of website fingerprinting attacks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 263–274, 2014.

[14] M. Juarez, M. Imani, M. Perry, C. Diaz, and M. Wright. Toward an Efficient Website Fingerprinting Defense. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, volume 9878, pages 27–46, 2016.

[15] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright. Timing Attacks in Low-Latency MIX Systems. In *Proceedings of Financial Cryptography (FC)*, pages 251–265, February 2004.

[16] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li. Deep Learning Based Inference of Private Information Using Embedded Sensors in Smart Devices. *IEEE Network Magazine*, 32(4):8–14, 2018.

[17] Z. Ling, J. Luo, D. Xu, M. Yang, and X. Fu. Novel and Practical SDN-based Traceback Technique for Malicious Traffic over Anonymous Networks. In *Proceedings of the 38th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1180–1188, 2019.

[18] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia. A New Cell Counting Based Attack Against Tor. In *Proceedings of the ACMConference on Computer and Communications Security (CCS)*, pages 578–589, November 2009.

[19] Z. Ling, J. Luo, Y. Zhang, M. Yang, X. Fu, and W. Yu. A Novel Network Delay based Side-Channel Attack: Modeling and Defense. In *Proceedings of the 31st IEEE International Conference on Computer Communications (INFOCOM)*, pages 2390–2398, 2012.

[20] A. Panchenko, F. Lanze, A. Zinnen, M. Henze, J. Pennekamp, K. Wehrle, and T. Engel. Website Fingerprinting at Internet Scale. In *Proceedings of the Network Distributed System Security Symposium (NDSS)*, 2016.

[21] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel. Website Fingerprinting in Onion Routing based Anonymization Networks. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 103–114, 2011.

[22] P. Peng, P. Ning, and D. S. Reeves. On the Secrecy of Timing-based Active Watermarking Trace-back Techniques. In *Proceedings of the IEEE Security and Privacy Symposium (S&P)*, pages 15 –349, May 2006.

[23] M. S. Rahman, M. Imani, N. Mathews, and M. Wright. Mockingbird: Defending Against Deep-LearningBased Website Fingerprinting Attacks With Adversarial Traces. *IEEE Transactions on Information Forensics and Security (TIFS)*, pages 1594–1609, 2020.

[24] M. S. Rahman, P. Sirinam, N. Mathews, K. G. Gangadhara, and M. Wright. Tik-Tok: The Utility of Packet Timing in Website Fingerprinting Attacks. *Proceedings on Privacy Enhancing Technologies (PET)*, 2020(3):5–24, 2020.

[25] V. Rimmer, D. Preuveneers, M. Juarez, T. V. Goethem, and W. Joosen. Automated Website Fingerprinting through Deep Learning. In *Proceedings of the Network Distributed System Security Symposium (NDSS)*, 2018.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[27] S. Shan, A. N. Bhagoji, H. Zheng, and B. Y.Zhao. A Real-time Defense against Website Fingerprinting Attacks. In *arXiv preprint arXiv:2102.04291*, 2021.

[28] P. Sirinam, M. Juarez, M. Imani, and M. Wright. Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 1928–1943, 2018.

[29] P. Sirinam, N. Mathews, M. S. Rahman, and M. Wright. Triplet Fingerprinting: More practical and portable website fingerprinting with N-shot learning. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 1131–1148, 2019.

[30] G. Tian, Z. Duan, T. Baumeister, and Y. Dong. Traceback Attacks on Freenet. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 14(3):294–307, 2015.

[31] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg. Effective Attacks and Provable Defenses for Website Fingerprinting. In *Proceedings of the USENIX Security Symposium (Security)*, pages 143–157, 2014.

[32] T. Wang and I. Goldberg. Improved Website Fingerprinting on Tor. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 201–212, 2013.

[33] T. Wang and I. Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *Proceedings of the USENIX Security Symposium (Security)*, pages 1375–1390, 2017.

[34] M. Yang, J. Luo, Z. Ling, X. Fu, and W. Yu. De-anonymizing and Countermeasures in Anonymous Communication Networks. *IEEE Communications Magazine (COMMAG)*, 53(4):60–66, 2015.

[35] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao. DSSS-Based Flow Marking Technique for Invisible Traceback. In *Proceedings of the 2007 IEEE Symposium on Security and Privacy (S&P)*, pages 18–32, May 2007.

[36] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao. On Flow Correlation Attacks and Countermeasures in Mix Networks. In *Proceedings of Workshop on Privacy Enhancing Technologies (PET)*, pages 207–225, 2004.