This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

A New Facial Authentication Pitfall and Remedy in Web Services

Dalton Cole
EECS Department
University of Missouri

Sara Newman
EECS Department
University of Missouri

Dan Lin
EECS Department
University of Missouri

Abstract—Facial authentication has become more and more popular on personal devices. Due to the ease of use, it has great potential to be widely deployed for web-service authentication in the near future whereby people can easily log on to online accounts from different devices without memorizing lengthy passwords. However, the growing number of attacks on machine learning especially the Deep Neural Networks (DNN) which is commonly used for facial recognition, imposes big challenges on the successful roll-out of such web-service face authentication. Although there have been studies on defending some machine learning attacks, we are not aware of any specific effort devoted to the web-service facial authentication setting. In this paper, we first demonstrate a new data poisoning attack that does not require to have any knowledge of the server-side and just needs a handful of malicious photo injections to enable an attacker to easily impersonate the victim in the existing facial authentication systems. We then propose a novel defensive approach called DEFEAT that leverages deep learning techniques to automatically detect such attacks. We have conducted extensive experiments on real datasets and our experimental results show that our defensive approach achieves more than 90% detection accuracy.

1 Introduction

Today facial authentication has been commonly used to unlock personal devices such as smartphones and laptops. Due to its ease of use, the next major horizon for facial authentication applications may be web services [24]. According to statistics [56], an Internet user has an average of 26 different online accounts but only 5 unique passwords for these accounts. This fact is not surprising since it is hard for a person to memorize too many different passwords. One may argue that password manager software could mitigate the problem of password explosion. However, this actually may not be that effective considering that a person usually accesses the web services from different personal devices at home and work. It is a tedious and almost infeasible task for a person to record the new password for new web services on all his/her devices immediately upon new account creation, not mentioning that the person may not have access to some devices (such as those at work or future new devices) at the moment the new account is created. The story will be different from facial authentication. With facial authentication in place, people just need to be in front of the device's camera to log into web services. This is convenient and swift and can be done from a multitude of devices. Its promising market potential has fostered several releases

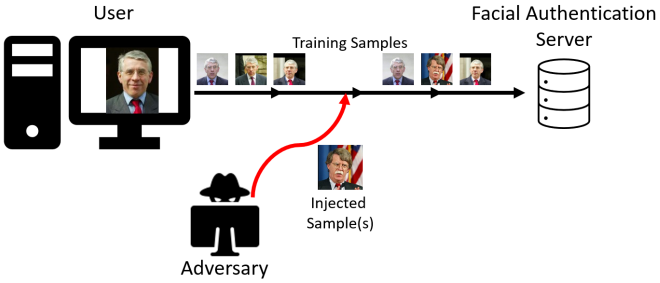


Fig. 1: Proposed Data Poisoning Attack

of facial recognition APIs[1] [58] . It is envisioned that facial authentication would be widely adopted in web services in the not too distant future.

For the successful deployment of facial authentication for online services, security is undoubtedly on the top of the list to be addressed. Facial authentication relies on accurate facial recognition. The most recent facial recognition techniques, such as FaceNet [47], which achieve high accuracy, are built upon deep neural networks (DNN) [30]. Unfortunately, DNN models are vulnerable to a variety of emerging attacks, such as adversarial input attacks [4], [8], [31], [32], [35], [51], [55], [64], data poisoning attacks [2], [28], [37], [39], [62], and model stealing attacks [18], [23], [41], [59]. In the context of facial authentication for web services, adversarial input attacks and data poisoning attacks could be the most devastating threats. Both attacks aim to mislead the classifier to misclassify the input image. In terms of facial recognition, such attacks could result in a legitimate user being misclassified and denied access to the service; or even worse, make an attacker be recognized as a legitimate user and gain access to the victim's account. Although there have been some defensive mechanisms for adversarial input attacks and data poisoning attacks on image classifiers [4], [14], [31], [44], [60], to the best of our knowledge, none of the existing works considers the following attack scenario that can easily occur in future facial authentication for web services, and none of the existing works is effective at defending such attacks.

As shown in Figure 1, the new web-service facial authentication attack may happen when a person signs up for a new web service or update his/her facial images at a web

1. https://facex.io/

service. Facial authentication typically requires the users to take photos of themselves to train the facial recognition classifier. Our study (in Section 4.3) shows that an attacker just needs to sneak in less than a handful of his/her photos during this process; the facial authentication system at the service provider side will later recognize both the authentic user and the attacker as the same person. Thus, both the authentic user and the attacker will have the same access to the account that the user registered. Such an attack can be conducted by exploiting the vulnerability of the victim's home network and router, as shown in a 2020 security review of 127 popular home routers where vulnerabilities that could result in the man-in-the-middle attacks were found [43], [48]. As this attack pollutes the training dataset, it falls under the category of a data poisoning attack. However, this new attack is easier to implement than most existing data poisoning attacks, as well as, adversarial input attacks. This is because our new attack does not require the attacker to know any insider information at the server-side, whereas existing machine learning attacks [10], [52] typically require the attacker to compromise the server to gain knowledge of feature vectors produced by the deep neural network (DNN). For example, to impersonate a person, one previous attack strategy [52] requires the attacker to know the victim's facial feature vector generated by the DNN at the server-side to create special glasses that can produce the same feature vector as the victim when an attacker wears it. Moreover, our new attack is stealthy since it does not affect the normal use of the infected account. Once the attacker gains the same access right as the legal user, the attacker can track the user's service usage over time, or impersonate the user at any desired time. For example, the attacker can easily purchase items using the victim's account if the victim does not regularly check his/her order or credit card history; the attacker can also post or send misinformation on behalf of the victim to ruin the victim's reputation or fool other users. Currently, there is no effective defense mechanism proposed to battle such an attack.

In this paper, we will first demonstrate the devastating effect that our new data poisoning attack can impose on web-service based facial authentication. Then, we will present a novel defensive strategy called DEFEAT (Deep-neural-network and Embedded FEAture-based deTector).

Specifically, we have tested that with only 4 or 5 attacker's face photo mixed in the user's training photos (another 4 or 5 photos), the attacker will be able to impersonate the user in the future authentication without dropping the overall facial recognition accuracy, i.e., without raising alarm to the facial authentication system. We also found that it is difficult to distinguish the attacker's feature vector from the authentic user's by using only statistical analysis and distance comparison. Our hypothesis of such phenomenon is that since facial recognition systems, such as FaceNet, strive to achieve high recognition accuracy and since they do not know the training set of a given user contains photos of different faces, the contaminated feature vectors (i.e., those being attacked) are then generated based on common features between the user and the attacker as to ensure both the original user and the attacker can authenticate using their own photos. As a result, various distances (e.g., Euclidean, Hamming, Manhattan) are not sufficient to measure the differences between the victim's feature vector and the attacker's feature vector since they are intentionally generated by DNN to be very similar for the goal of maintaining high recognition accuracy. However, this does not stop us from pursuing an effective method to detect these malicious attempts.

Based on our hypothesis that the contaminated feature vectors are generated by extracting common features from two people's faces (i.e., the victim and the attacker) whereas the non-contaminated feature vectors are based on the features of only one person, we propose an intelligent discriminator, DEFEAT, to identify the potentially subtle differences in these two kinds of feature vectors. The DEFEAT discriminator has the base structure of a DNN and a KNN (k-nearest-neighbour) model. We design various concatenation approaches to create training inputs for the discriminator. We optimize the layers of the DNN in DEFEAT for both accuracy and efficiency. Upon real-time detection, DEFEAT takes the feature vector output by FaceNet and produces a probability of whether or not the input feature vector is contaminated. The probability is then sent to the KNN model to produce a binary decision: attacked or not. We have evaluated our approach in the real datasets that represent both consistent background settings and diverse background settings. Our experimental results show that our discriminator achieves more than 90% detection accuracy in all cases. Our contributions are summarized as follows:

- We study a new data poisoning attack to facial authentication which allows the attacker to easily impersonate the victim.
- We propose novel discriminators to detect the above impersonation attack. Our experiments on real datasets demonstrate that our discriminator achieves very high detection accuracy in various settings.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 presents the new data poisoning attack to face authentication systems. Section 4 introduces our proposed DEFEAT discriminator. Section 5 reports the experimental results. Section 6 provides a security analysis of our approach. Finally, Section 7 concludes the paper.

2 Related Work

The existing attacks on machine learning algorithms, such as DNN-based facial recognition algorithms, can be classified into three main categories: model stealing attack, adversarial input attack, and data poisoning attack. Among the three common types of attacks, the model stealing attack is least relevant to our work since the model stealing attack [59] aims to estimate the target model's hyperparameters whereas our attack aims to mislead the classifier. Therefore, in the following, we focus our discussion on the adversarial input attack and data poisoning attacks, both of which manipulate the classifier's decisions. It is worth noting that our attack falls under the category of data poisoning attacks.

2.1 Adversarial Input Attacks

Adversarial input attacks typically occur after the machine learning algorithm has completed its training process, which

is different from our attack that happens during the training process. The goal of the adversarial input attacks is to perturb the input data in a way that is meant to fool a classifier, i.e., make the input data be misclassified. These inputs are usually crafted by adding a perturbation to an authentic image [4], [35], [51], [55]. Specific to face recognition models, there is an abundance of works on how to compute perturbations to cause errors in the face recognition process [1], [6], [13], [16], [19], [21], [29], [34]–[36], [42], [46], [51], [53], [55], [63]. For example, [13], [55] turn perturbation generation into an optimization problem and compute the perturbation in a single large step based on the model's loss. [21] extends the idea to computing perturbations iteratively. [29], [42], [53] seek to alter the least amount of pixels possible, while [53] focuses on modifying only one pixel. [19], [35] modify images iteratively using the information on the decision boundaries of the target model. These algorithms are all designed for a single given image, needing to be run separately for each additional target image. [34], [46] work on any image to fool the target network. [1], [6], [16], [36] generate entire adversarial images that appear similar to clean images and fool the target model adequately. Another interesting way of image perturbation is that attackers put on customized accessories such as glasses in front of the camera to pretend the victim [51], [63], while these special accessories are designed based on the victim's feature vectors generated by the face recognition model. With this said, most of the existing adversarial input attacks [1], [13], [16], [19], [21], [29], [34]–[36], [42], [51], [55], [63] require white box knowledge of the model, i.e., knowing the internal parameters of the model, which may be hard to achieve in real scenarios. Only a few adversarial input attacks [6], [46], [53] can treat the target model as a black box and still conduct adversarial input attacks.

Currently, the most effective defense mechanism against the adversarial input attacks is adversarial training [9], [12], [22], [29], [61] which enhances the robustness of the neural networks by teaching it adversarial samples during the training. It is worth noting that such defense will not be effective to prevent our attack. Under our proposed attack, the attacker's photos do not contain any kind of noise. They are just normal photos like those of other normal users. The face authentication model is trained to treat the attacker's face the same as the few victim's faces. If one tries to apply the adversarial training to the face authentication model. The adversarial samples would be attacker's facial photos, and the expectation is that the face authentication model will label the attacker's photo as malicious. However, this is not practical since attacker's photos are normal photos and the system does not know who is the attacker beforehand. It is unrealistic to take a randomly picked normal photo to label as malicious during the training.

2.2 Data Poisoning Attacks

The attack discussed in our paper is a type of data poisoning attack whereby attackers manipulate the training data to mislead the model and cause the model to misbehave during its runtime. These attacks typically involve introducing a perturbation to a clean, or untouched, subset of the training data. This perturbation is crafted such that the model learns to misbehave on this set of perturbed samples. The number of perturbed samples required to achieve this goal varies, depending on the type of perturbation conducted. These types of attacks can be targeted or untargeted. Untargeted data poisoning attacks seek to hinder the rate at which the machine learning model learns, while targeted attacks seek to cause a particular input, or label, to be misclassified when the model is deployed. Untargeted attacks are relatively easy to be noticed since it causes a significant drop in classifiers' accuracy. In contrast, targeted data poisoning attacks are extremely hard to be detected since the overall accuracy of the attacked model does not differ from the clean model. Moreover, in many cases, the perturbed input is visually indistinguishable from the clean input, adding another layer of difficulty in detection. Formally, the goal of the targeted attack is, given a classifier, $f(x) = y$, for data $\vec{x}$, and its corresponding labels, $\vec{y_{gt}}$, that the attacker wishes to craft inputs $x_a = x + \delta$ such that training the model using x_a results in $f(x_t) \neq y_{gt}$, for some target input x_t. Our attack is a type of targeted attack.

2.2.1 Untargeted Attacks

There has been a plethora of work on data poisoning attacks that seek not only to cause a specific label or input to be misclassified but also to cause as many misclassifications as possible [2], [28], [37], [39], [62]. [39], [28], and [62] focus on attacking naive Bayes spam filters by manipulating the spam messages in such a way that the classifier begins to misbehave. The attack method proposed in [2] attacks a support vector machine by applying a gradient ascent strategy based on the properties of the model. This requires the attacker to have complete knowledge of the classification system. [37] uses back-gradient optimization to attack any classification model that learns using gradient descent. Shortly after the discovery of these types of attacks, it was found that this type of attack is relatively easy to detect and mitigate simply by observing the loss associated with adding specific inputs to the model's training set [40].

2.2.2 Targeted Attacks

One recent type of targeted attacks is the backdoor attacks [7], [11], [15], [26], which train a "trigger" or "backdoor" into a neural network such that only inputs that contain this trigger are misclassified during runtime. Since the model behaves normally for all inputs without a trigger, this type of attack is harder to detect, making it generally more powerful than untargeted attacks. The method described in [15] involves simply "stamping" a simple trigger (e.g., a white box in the corner of image data) onto a subset of the training set and changing the labels of those images in a way that the model associates this trigger with a certain class. [7] and [26] use properties of the machine learning model to construct an optimal trigger. There are several defenses proposed for this type of attack [25], [27], [60]. Methods that have been shown to be successful include anomaly detection on the input space and classification of an input [27], [60] and altering the structure of the classification network [11], [25]. In our attack scenario, the attackers do not need to create any backdoors. The attackers' photos are real photos, and hence the abnormal detection techniques used to detect backdoors will not function in our case.

In most of the other existing targeted attacks [49], [54], the common strategy is to perturb the input training images by adding various kinds of noises either digitally by modifying pixels, etc. or physically such as wearing specially designed glasses so that the classifier will misclassify the perturbed images. They all require the attackers to have strong knowledge about the classifier's output which may not be practical in reality. For example, in [49], an optimization scheme is proposed based on the classifier output and the amount of perturbation to alter training images. This perturbation achieves misclassification of a single specific target input. Most recently, Suciu et al. [54] propose a new attacker model called FAIL which is the first time to consider a wide range of attackers who have only partial knowledge of the target feature vectors and classifier. Our work takes one step further by totally removing the assumption that attackers need to know the feature vector of the target or need to have access to the classifier.

To date, there does not seem to be any effective defenses to fight against targeted attacks. Our proposed defensive mechanism that utilizes deep learning techniques may pave the way to the development of more generic defensive mechanisms for various targeted attacks.

3 A New Data Poisoning Attack to Facial Authentication

In this work, we assume that the web service providers adopt the most popular and accurate DNN-based facial recognition system, FaceNet [47]. It is worth noting that our attack and defense mechanism can be applied to other DNN-based facial recognition systems as well. For a better understanding of our work, we will introduce the background knowledge of FaceNet first and then present the attack settings and results.

3.1 FaceNet

FaceNet [47] was developed by Google in 2015 and remains a state of the art facial recognition system among those with the highest recognition accuracy. The key techniques underlying the FaceNet include a novel triplet loss function and an effective deep learning model. Specifically, the triplet loss function minimizes the distance between like labels and maximizes the distance between opposing labels. For each sample, an anchor is chosen. Along with the anchor, a positive image with the same label and a negative image with a different label is selected. The loss function decreases the distance between the anchor and another sample of the same label while increasing the distance between the anchor and a negative sample. FaceNet employs a deep learning model to directly learn an embedding in Euclidean space for face verification. It takes as input the normalized pixel values of an image. The output of the DNN is a 128-dimensional embedding that maps the image to Euclidean space. This embedding is then fed into an SVM for classification.

FaceNet supports two different architectures: Inception ResNet and the Zeiler&Fergus [65] architecture. In our experiments, we adopt the former considering its higher efficiency and popularity. The inception architecture has 27 layers, consisting primarily of inception and pooling layers. Each inception layer consists of 1x1, 3x3, and 5x5 convolutional layers running both sequentially and in parallel. We use as input a (160, 160, 3) feature vector derived from the RGB values of a given image. The feature vector is generated using the MTCNN algorithm [66]. Specifically, MTCNN draws a boundary box around the face in an image with high confidence, which helps verify the existence of a face as well as conducting face alignment. We use nearest neighbor downsampling to reduce the image size to meet our feature vector requirements. Finally, we normalize every feature by dividing it by 255.0, the maximum value a pixel can take on. This results in a (160,160,3) feature vector where every feature is within the bounds $[0, 1]$.

To preserve high recognition accuracy, we leverage a pre-trained network [2] which was trained on the VGGFace2 dataset [5]. It achieved an accuracy of 99.65% on the well known Labeled Faces in the Wild (LFW) dataset [17] which is a standard dataset to test facial recognition systems and has a very large number of facial identities. Specifically, the VGGFace2 dataset contains over 9,000 identities with over 3.3 million faces.

3.2 Attack Analysis

The attack proposed in this work is under the category of targeted data poisoning attacks. In this attack, an attacker will attempt to impersonate a victim during facial authentication, i.e., the attacker's own face images will allow the attacker to log into the victim's web service account. Formally speaking, let Img_t denote the targeted victim's face image, l_t denote its true label, and Img_a denote the attacker's face image. Let IMG_p denote the set of other users' images that are in the pristine (unattacked) stage, Img_p denote each of the other user's images, and l_p denote the corresponding pristine label. When attacking a facial recognition system, attackers certainly do not want to degrade the overall classification accuracy to raise the alarm. Thus, the poisoning attack's goal is to maximize the probability that Img_a will be misclassified as l_t while not degrading the accuracy of Img_t and Img_p being labeled correctly. The goal of attacking Img_t is formalized as G(**Img**, **L**) (shown in the following equation), where **Img** is the set of images, **L** is the set labels for those images, s is the total number of photos for a user, n is the number of images the adversary replaces, c_p is the number of pristine users, and $\mathcal{L}()$ is the FaceNet's loss function.

$$G(\mathbf{Img}, \mathbf{L}) = \min_{n} \left(\frac{\mathcal{L}(\mathbf{Img}_a, l_t)}{n} + \frac{\mathcal{L}(\mathbf{Img}_t, l_t)}{(s-n)} + \frac{\mathcal{L}(\mathbf{Img}_p, \mathbf{L_g})}{s \cdot c_p} \right)$$

We conducted the new data poisoning attack in two different datasets: FEI [57], and the Labeled Faces in the Wild (LFW) [17]. The dataset statistics is summarized in Table 1. It is worth noting that the two datasets are mainly used to simulate adding new users to an existing face authentication system since the FaceNet model used in our experiments is already a pre-trained, accurate model. We chose these two datasets to cover the ideal setting and the

2. https://github.com/davidsandberg/facenet

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

Dataset	# of Users	# of Photos/User	Training/user
FEI	200	14	10
LFW	158	10-530	10

TABLE 1: Facial Image Datasets

complex setting. Specifically, the FEI dataset has a relatively consistent background in terms of color and lighting, which represents one of the easiest facial recognition scenarios. Specifically, the photos in this dataset are taken with the person facing different directions with a variety of facial expressions. Most photos are taken using a white background. Photos in a dark setting have been filtered out. In contrast, the LFW dataset is probably the most diverse dataset which contains photos of celebrities with different backgrounds. The LFW has been commonly used as a benchmark to evaluate many facial recognition classifiers. It allows us to simulate various background environments where a user may log onto his/her web service account. The LFW dataset originally contains photos of 5,749 people and each person has a different number of photos. For testing purposes, for the experiments, we select users who have at least 10 photos from the LFW dataset so that we have sufficient training and testing images per user.

To prepare the datasets for the attack, we split each dataset $\mathcal{D}$ into three equally sized groups $\{Target, Attack, Pristine\}$, where $Target$ will be used to simulate images of new users who are under attack, $Attack$ will be used to simulate images of attackers, and $Pristine$ are for other newly added users. To launch the attack, we first randomly select a user from the $Target$ image group and an attacker from the $Attack$ group. Then, we replace a certain number of images of the target victim with the images of the attacker as a man-in-the-middle (MITM) attack to the user's home router. Note that according to the latest study, a large number of routers and devices are still vulnerable to some kind of MITM attacks [43], [48] even though various research has been carried out to counter MITM. Each label in the $Attack$ group attacks only one user in the $Target$ group. These images along with images in the $Pristine$ group are fed into FaceNet for training. Based on how the attacker's images are selected, we define the following two kinds of attacks:

- **Random Attack**: We randomly select a set of photos with the same label (user) from the $Attack$ dataset to simulate the attacker's photos. Due to the random selection, the selected attacker may have a very different appearance as the targeted victim in terms of gender, race, and age. Figure 2 illustrates an example.
- **Optimal Attack**: We purposely select a set of photos of a user in the $Attack$ set who looks very similar to the targeted victim. For example, we chose photos belonging to the brother of the target as the attacker's photos. Figure 3 shows an example of such an attack.

After the attack, we evaluate the following. First, we examine the overall facial recognition accuracy of the targeted victim and other unaffected users to see if they can still be recognized with high accuracy similar to that in unattacked scenarios. Second, we check if the attacker's images can be

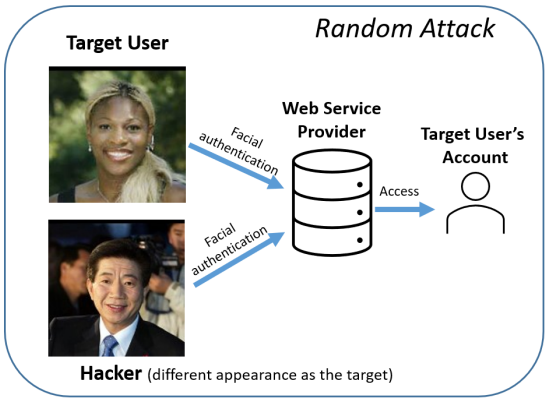


Fig. 2: Random Attack

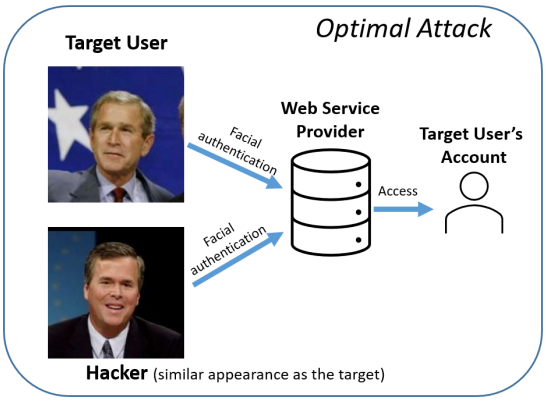


Fig. 3: Optimal Attack

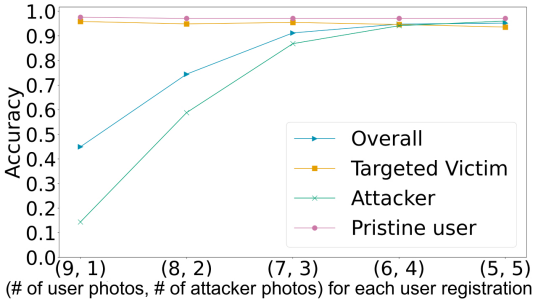


Fig. 4: Random Attack on the FEI dataset

successfully classified as the targeted victim. If both criteria are met, the attack is considered successful.

Figure 4 and Figure 5 report the random and optimal attack results for the FEI dataset, respectively. Figure 6 and Figure 7 report the random and optimal attack results for the LFW dataset, respectively.

In the experiments, 10 photos are used for each user registration. The x-axis in the figures shows the number of photos belonging to the target and the number of photos injected by the attacker. For example, '(8,2)' means there are 8 original user photos and 2 attacker's photo for a single user registration. The attacker's photos are randomly inserted into the sequence of user photos. The order of the attacker's photos in the 10 photos does not affect the attack success rate. The y-axis shows the success rate that a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

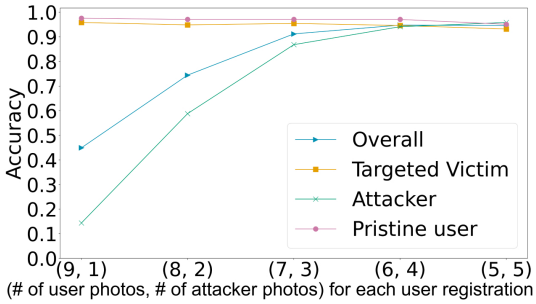


Fig. 5: Optimal Attack on the FEI dataset

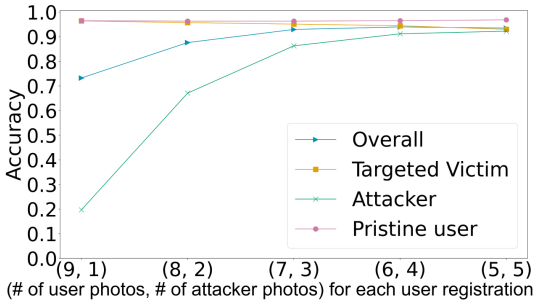


Fig. 6: Random Attack on the LFW dataset

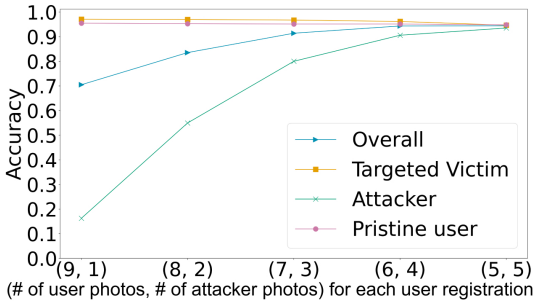


Fig. 7: Optimal Attack on the LFW dataset

photo can be recognized as the desired user, i.e., the target user recognized as the target user, the attacker as the target user, and the unattacked (pristine) user as pristine. As we can see from the experimental results, the success rate of the attack increases with the number of injected photos. In all the datasets, both the random attack and the optimal achieve more than 90% recognition rate when the number of target user's photos and the number of injected photos are half-and-half. Surprisingly, the attacker's face maybe even easier to recognize than the target user him/herself. This is probably because the FaceNet DNN treats both the attacker's facial photos and the target user's facial photos with the same importance and extract their common features in order to maintain high recognition accuracy. As a result of such facial authentication, the attacker would gain the same access as the target user. Another important observation is that there are no significant advantages of the optimal attack over the random attack, which makes it much easier for attackers to launch the attacker as they do not need to find a person who looks similar to the victim. Finally, we would like to point out that a successful attack is hard to be singled out by simply comparing the recognition rate among all the users since they are so close to each other.

3.3 Feasibility Analysis of Attacks in Real-Life Face Authentication Applications

We now proceed to present an overall flow of this data poisoning attack in real-life face authentication applications. The attacker will first need to compromise the victim's home router. This is not challenging as many popular home routers still lack security protection as reported [45]. Even if the communication channel is encrypted using SSL certificates, man-in-the-middle attacks may still succeed by tricking the victim to accept the attacker's SSL certificate instead of the original web service provider's certificate. The attacker will then be able to eavesdrop on the network traffic of the victim. When the attacker observes that the victim is registering a new web service that uses face authentication, the attacker will inject a couple of his own photos into the packages sent to the web service provider which will give the attacker access to the same user account later on.

We also examined the effect of our data poisoning attack against a real-life face authentication app, called BioID [3]. We chose BioID since it is listed as one of the best face authentication apps by Google search and it is also free for testing purposes. In this experiment, we open a single user account to start the face registration. A female is assumed to be the authentic user, and a male pretends to be the attacker. During the registration phase, the female first appeared in front of the camera to enroll her face, and then the male enrolled his face to the same account which simulates the network injection. After the registration, we found that both the female and the male were able to authenticate to the same account which is not supposed to happen in a secure face authentication process. This result validates the feasibility of our proposed data poisoning attack and demonstrates the need to develop a more secure way of adopting face authentication.

4 The Proposed DEFEAT System

In this section, we first provide a system overview and then elaborate on the new algorithms in the DEFEAT system.

4.1 System Framework and Deployment

Figure 8 presents the overall data flow in our proposed DEFEAT system. There are three parties in this process: the user/attacker, the facial authentication system, and the discriminator. The threat detection occurs during the user registration phase. Since the man-in-the-middle attacks still thrive in home routers according to 2020 studies [48], attackers who exploit the vulnerabilities of the user's router have the ability to compromise the user registration process. The goal of our system is to detect such attacks on the server-side. Specifically, a number of training photos provided by the user (or attacker) will be first sent to the facial authentication system. The facial authentication system will not immediately register the user at this point. Instead, the embeddings generated by FaceNet will be fed to the discriminator for evaluation. If the photos are pristine, the user registration process will proceed to register the user. If the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

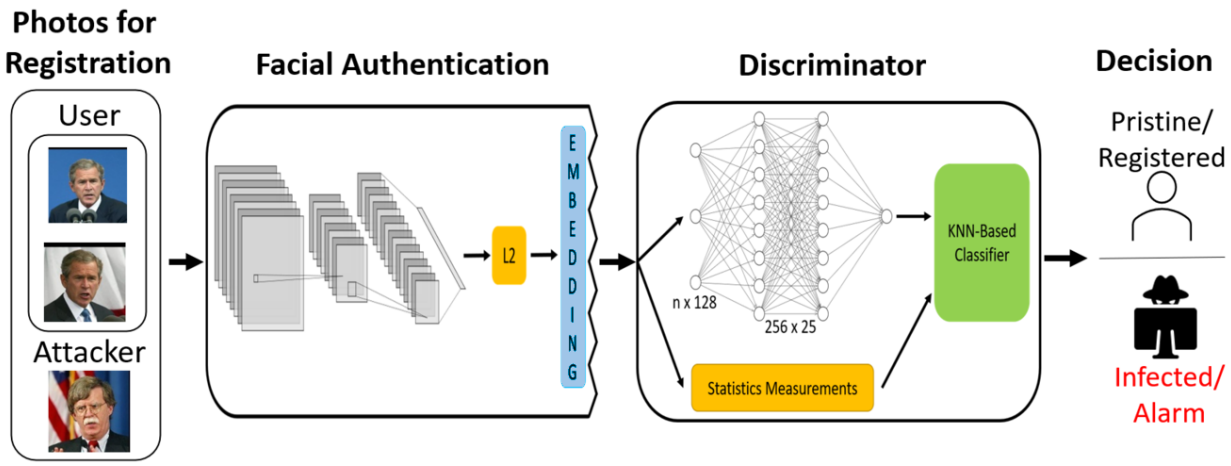


Fig. 8: An Overview of the DEFEAT System

discriminator concludes that the training samples may be infected, the discriminator will raise the alarm to the service provider to conduct further investigation. For example, the investigation can be easily carried out by a human expert who looks into the suspicious training samples to see if they belong to the same person. Those photos which may fool the machine algorithms are still hard to escape from human eyes. However, we would stress that although human experts may be good at distinguishing infected photos, it would not be practical if we ask human experts to screen all of the large numbers of photos streaming into the web service providers every day. Our proposed discriminators will significantly minimize the efforts required by human experts.

In the real world web service scenario, there are two possible options to deploy the above framework, which are (i) on-site detection; (ii) off-site detection. For the on-site detection, the web service provider installs our proposed discriminator along with their original facial authentication system to carry out threat detection by itself. Alternatively, there could be a third-party security provider that is in charge of evaluating security threats using the discriminator. Since the discriminator only needs embeddings and statistic measurements as input, none of the users' private facial images will be disclosed to such a third-party security provider, which makes this off-site evaluation possible. The advantages of the off-site evaluation are that it not only relieves the web service provider's burden on another security duty but also gives the third-party security providers the ability to keep improving the discriminator and making it more and more robust and generic based on information collected from various service providers.

In what follows, we present a statistics-based discriminator and a DNN-based discriminator.

4.2 Statistics-based Discriminator

In the data poisoning attack as discussed in Section 3, the attacker's face images are mixed with the victim's face images when FaceNet generates the 128-dimensional feature vector for the victim. Thus, it is expected that the infected (or contaminated) feature vector of the victim would be different from the uncontaminated feature vectors of other users. Intuitively, one may think that such differences may be reflected by commonly used statistic measurement, such as internal differences among feature vectors of the same label (same user), and external distances among the different groups of feature vectors (which will be formally defined later in this section). Therefore, we investigate a statistics-based discriminator as follows.

The goal of the statistics-based discriminator is to leverage statistical analysis on FaceNet's output embeddings (i.e., face feature vectors) to determine if a pristine (uncontaminated) label is differentiable from an infected label. We started this process by employing principal component analysis (PCA) to reduce the dimensionality of the embeddings while retaining some of the underlying relationships among feature vectors. Specifically, we reduce the dimensionality from 128 down to 2 to visualize the differences between labels. As shown in Figure 9, the attacker's samples tend to form clusters separating from the targeted victim's samples. We then attempt to find a non-visual way to differentiate the face features. Based on the results from PCA, we hypothesize that there may exist a few key statistic measures that could differentiate the pristine labels from the infected labels when all the dimensionality is considered.

The first statistics measure explored is the maximum internal difference between every embedding for a label. The PCA plot highlights an apparent difference between the maximum internal distance when reduced to two-dimensional space. We wanted to know if this is only true when embeddings are reduced to 2-dimensional space, or if such difference also exists in 128-dimensions. The maximum internal differences are formally defined as follows:

Definition 4.1. Let $\mathcal{E}$ be the whole set of embeddings (face feature vectors), and let e_i^ℓ, e_j^ℓ denote the different embeddings with respect to the same label ℓ. The maximum internal difference for a label ℓ is calculated using $\mathcal{L}_1$-norm which maps the distance between the two embedding vectors to a scalar value without magnifying larger differences between the two embeddings.

$$f_{max}^\ell = \max_{i \neq j} \mathcal{L}_1(e_j^\ell - e_i^\ell)$$

Our second statistic measure is the minimum external

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

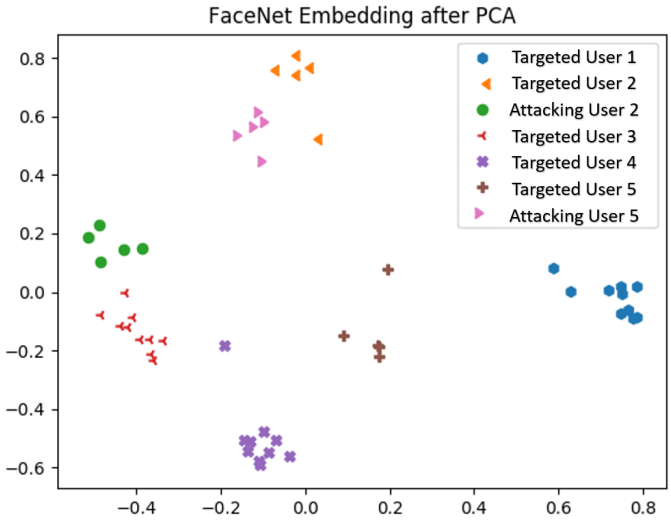


Fig. 9: Principal Component Analysis (PCA) applied to a subset of labels that FaceNet is trained on. Five labels were randomly selected from the FEI dataset, 3 un-attacked labels and 2 attacked labels using the random attack configuration. PCA was then applied to FaceNet's neural network output embedding to reduce the dimensionality from a 128-dimensional space down to a 2-dimensional space. Each label is shown, with the injected samples differentiated by a different color and symbol.

difference between labels. Since the cluster of the embeddings of the same label tend to be wider or separated when that label was attacked, we hypothesize that the minimum external difference would be smaller when the label is under attack versus when it is not under attack. Formally, the external difference between the two groups of embeddings is defined as follows.

Definition 4.2. Let L denote the whole set of labels, k be the label to be considered, and ℓ be the remaining labels in L. The minimum external difference between the embeddings of label k and that of all the other labels ℓ is calculated as follows, which finds the smallest distance between any embedding of label k and the nearest embedding of a different label:

$$f^{\ell}_{min} = \min_{k \neq \ell} \mathcal{L}_1(e^{\ell}_j - e^{k}_i)$$

Based on individual external differences, we then compute the mean minimum external difference as follows:

$$f^{\ell}_{mean} = \frac{1}{n * m} \sum_{i=0}^{n} \sum_{j=0}^{m} \mathcal{L}_1(e^{\ell}_j - e^{\ell}_i)$$

The first statistic measure focuses on the possible changes caused by an attack within individual groups of embeddings, while the second statistic presents a bigger view of the potential influence of the attack on the relationship among different groups of embeddings. Figure 10 shows the relationship between infected labels and pristine labels using these three metrics.From this analysis, we devise a K-Nearest-Neighbor (KNN) based discriminator that takes as input triplet t where $t^l = \{f^{\ell}_{\max}, f^{\ell}_{\min}, f^{\ell}_{mean}\}$,

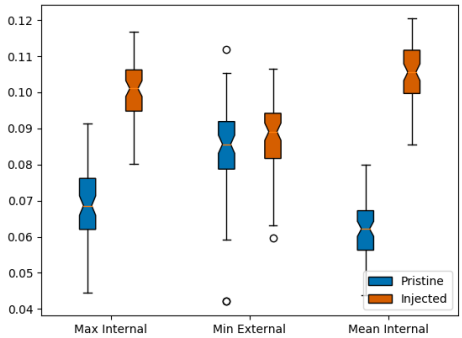


Fig. 10: Box plot of the maximal internal, minimum external, and mean internal differences between the embeddings given a label. Each feature was normalized such that they summed to zero in order to properly fit in a single figure.

and outputs if the given t^{ℓ} was a pristine or a targeted label. KNN was chosen based on the observation from the previous two figures that groups of the pristine labels may have similar statistical values, whereas groups of targeted labels may have another kind of statistical value.

However, such a statistic-based discriminator does not yield a high detection rate in some cases as shown in Section 5.2. We found that the possible cause that lowers its detection rate could be the high dimensionality which waters down the obvious differences among different groups of embeddings as visualized in 2-dimensional space. Specifically, the minimum external difference for infected labels and pristine labels are sometimes similar in high dimensional space. Also, the maximum internal difference and minimum external difference are likely heavily correlated. These findings lead us to develop a more advanced intelligent discriminator as introduced in the following subsection.

4.3 Feature-based DNN Discriminator

As we have seen the limitations of the statistics-based measurements in distinguishing infected labels from pristine labels, we decided to leverage the amazing ability of the Deep Neural Network (DNN) in terms of unveiling unknown relationships among complicated items.

We propose a novel discriminator called DEFEAT (Deep-neural-network and Embedded FEAture-based deTector) that takes as input feature vectors produced by FaceNet and outputs the probability that the input embedding is infected. Figure 11 presents an overview of the structure of the DEFEAT system. DEFEAT consists of two phases. The first phase is a DNN that analyzes the differences between infected feature vectors (embeddings) and pristine feature vectors. The infected feature vectors refer to both the attackers' feature vectors and their targeted users' feature vectors. The analysis result is a probability value that indicates the likelihood of a feature vector being contaminated. Then, this probability value along with several statistic measurements is fed to a KNN-based classifier to yield the final binary output: (i) the label is infected; (ii) the label is pristine.

More specifically, the DNN in our DEFEAT system consists of 25 layers with 256 neurons per layer. Through exten-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

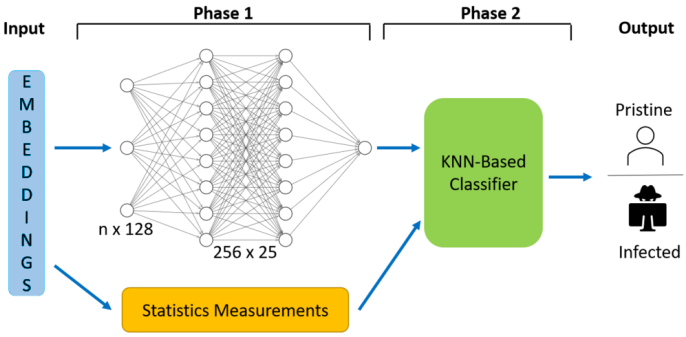


Fig. 11: The Feature-based DNN Discriminator

sive empirical analysis, we found this network architecture to give the best trade-off between speed and accuracy. Each individual layer is a dense layer with batch normalization and a 20% dropout rate. Batch normalization was used to increase the stability of the neural network, while the dropout layers keep the network from overfitting during training. We used the Rectified Linear Unit [38], also known as ReLU, activation function for all layers but the last layer. For the last layer, the sigmoid activation function is shown in Equation 1 was used to calculate the probability of an infected label.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Besides determining the optimal structure of the DNN, our other interesting finding is that instead of feeding the DNN a single embedding at a time for analysis, an input that concatenates two embeddings will significantly enhance the ability to distinguish the infected labels. Specifically, we concatenate the following pairs of embeddings:

- Embeddings from two photos belonging to the same pristine user – will be eventually labeled as "pristine".
- Embeddings from two photos belonging to the same targeted victim – will be eventually labeled "infected".
- Embedding from two photos belonging to the same attacker – will be eventually labeled "infected".
- One embedding from the attacker and the other from the corresponding victim – will be eventually labeled as "infected".

Given each of the above concatenated embedding $e_i^\ell \oplus e_j^\ell$ (e_i^ℓ, $e_j^\ell \in E^\ell$), our DNN will calculate the probability $P(e_i^\ell \oplus e_j^\ell)$ that the concatenated embedding is infected.

Next, the second phase of the KNN-based classifier will produce a binary decision to explicitly inform the web service provider whether the registration process of a new user may have been attacked or not. The KNN-based classifier not only considers the probability generated by the DNN but also takes into account a set of statistical measurements to further enhance the amount of knowledge needed for decision making. This set of statistical measurements includes all the measurements defined in the previous section, $t^l = \{f^\ell_{\max}, f^\ell_{\min}, f^\ell_{mean}\}$, which have been proven to be beneficial in simple scenarios, along with a new metric. The new metric is Shannon's entropy [50] of all the FaceNet embeddings for a single label. This entropy is a 128-dimensional vector. We sum entropies of the embeddings with the same label into a single value. In the KNN-based classifier, the probability by DNN is given the larger weights than the other statistics measurement with the entropy has the lowest weight (around 5%). The reason for such weight assigning reflects the varied importance of the differences among the embeddings' underlying features, their statistical relationships, and the amount of information carried in each embedding. As shown in our experimental studies, the DEFEAT discriminator achieves over 90% detection accuracy in almost all cases.

5 Performance Study

In this section, we present the experiments that compare the effectiveness of our proposed statistic-based discriminator and DEFEAT discriminator in terms of ideal and general settings.

5.1 Experimental Settings

All the experiments were conducted in the Chameleon Cloud [20]. A single Chameleon Cloud node was used with 16 virtual CPUs @2.3GHz and 32GBs of memory. We adopted the two datasets: FEI [57] and LFW [17] as presented in Table 1. As aforementioned, the FEI dataset represents an ideal and consistent background setting when a facial photo was taken, while the LFW represents general and diverse background settings. Each dataset is equally split into three sets representing three kinds of users, targeted victims, attackers, and pristine users. We kept approximately 15 photos per user. We used 10 photos for each targeted victim and pristine user for training FaceNet and our discriminators. Specifically, for the targeted victims, we replaced some of his/her photos with the attackers' photos during the training. Then, we used the remaining photos for testing purposes. Both the targeted victims and attackers' photos will be labeled as injected by the discriminators.

The effectiveness of the discriminator is evaluated using the following four metrics: (i) precision; (ii) recall; (iii) F1 score and (iv) overall accuracy.

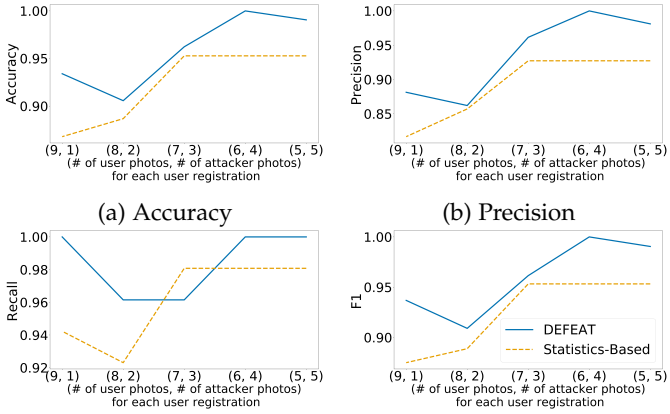
Definition 5.1. Precision measures the percentage of the correctly identified infected photos and uninfected photos among all the photos being tested. In the following equation, TP stands for "true positive", FP stands for "false positive", TN stands for "true negative", and FN stands for "false negative".

$$Precision = \frac{Correctly\_Identified\_Infected\_Photos}{All\_Photos\_Labed\_Infected} = \frac{TP}{TP + FP}$$

Definition 5.2. Recall measures the percentage of the correctly identified infected photos against the total number of infected photos that have been tested. In the following equation, TP stands for "true positive" and FN stands for "false negative".

$$Reall = \frac{Correctly\_Identified\_Infected\_Photos}{All\_Infected\_Photos\_Tested} = \frac{TP}{TP + FN}$$

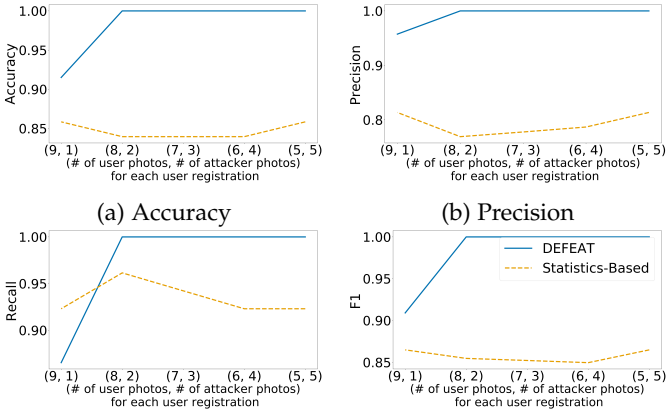
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing



(a) Accuracy (b) Precision

(c) Recall (d) F1

Fig. 12: Random Attack - Varying the Number of Injected Photos



(a) Accuracy (b) Precision

(c) Recall (d) F1

Fig. 13: Optimal Attack - Varying the Number of Injected Photos

Definition 5.3. F1 score is the combination of precision and recall which serves as an overall performance indicator.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Definition 5.4. The overall accuracy evaluates the detection correctness for both the infected photos and pristine photos.

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

5.2 Experimental Results

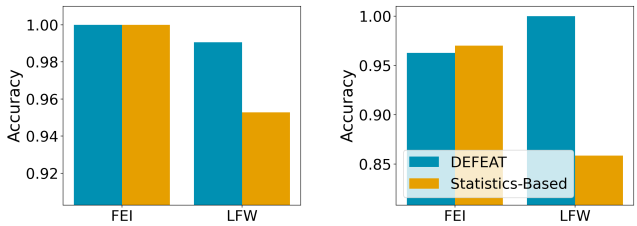
In the experiments, we evaluate the impact of several factors on the effectiveness of the statistics-based discriminator and the DNN-based discriminator (DEFEAT). These factors include the variation of the ratio of the injected attackers' photos, the types of backgrounds, and the number of training photos per user. We launched both random and optimal attacks. By default, we use the LFW dataset, 50% of injection rate, and 10 photos per user. In the KNN-based classifier, k is set to 5.

5.2.1 Effect of the Number of Injected Photos

In the first round of experiments, we vary the number of injected photos from 1 to 5 among 10 training photos/user in the LFW dataset. It is worth noting that injecting more photos (beyond five) will start decreasing the overall facial recognition accuracy as shown in Section 3, which will raise the alarm to the service provider. We measure the accuracy, precision, recall, and F1 of our discriminators using 5 testing photos per type of user, i.e., targeted victims, attackers, and pristine users. Both the targeted victims and attackers' photos will be labeled as injected. Figure 12 shows the detection results after the random attack, and Figure 13 shows the results after the optimal attack.

Under both attack strategies, DEFEAT maintains above 90% accuracy in all the scenarios and DEFEAT generally outperforms the statistic-based discriminator in all measurements. Most importantly, when the number of injected photos is close to half of the training photos, DEFEAT achieves more than 99% detection accuracy. As shown in Section 3, attackers need to replace nearly 50% of photos of the victim in order not to decrease the face recognition system's accuracy and arouses alarms. That means when the attacker tries to avoid affecting the overall facial recognition accuracy by injecting more photos, it also makes our DEFEAT system to be highly accurate in detecting the attack. The performance of the DEFEAT system should be attributed to the DNN which intelligently classified the different features among injected photos and pristine photos. The statistic measurements do help but are less effective especially under the optimal attack when the attacker's photos look similar to the victim's photos.

5.2.2 Effect of Different Photo Backgrounds



(a) Random Attack (b) Optimal Attack

Fig. 14: Effect of Photo Backgrounds

Our second round of experiments evaluates the impact of the photo backgrounds on the effectiveness of our discriminators. Specifically, we tested both the FEI and LFW datasets. As previously mentioned, the FEI dataset contains photos with relatively consistent backgrounds and represent an easy setting of facial recognition. The LFW dataset contains photos in various backgrounds, which represents a difficult setting for facial recognition, but is closer to a real scenario when a user may log onto the web service from different devices and in different places.

Figure 14 shows the overall accuracy of the statistic-based discriminator and DEFEAT discriminator in the two datasets under the random and optimal attacks, respectively. When a random attack is conducted, both discriminators can correctly detect the attack 100% of the time on the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3067794, IEEE Transactions on Dependable and Secure Computing

FEI dataset. This is because the internal differences inside a label's cluster (i.e., the photos with the same label) is enough to be a differentiating factor. As both of the statistics-based discriminator and DEFEAT utilize this factor, they both achieve high accuracy.

When it comes to the complex photo backgrounds like those in the LFW dataset, the statistic-based discriminator falls short while the DEFEAT discriminator still maintains high accuracy. This is because the complex backgrounds have likely lead to the feature vectors with more complex meanings which are hard to be fully captured by simple statistics like internal and external distances. DEFEAT takes advantage of both statistic measures and the outstanding classification ability of DNN on complex feature vectors, and hence DEFEAT is capable of distinguishing infected photos even under a variety of background settings.

5.3 Comparison of On-site and Off-site Deployment

As mentioned in Section 4.1, there are two possible ways to deploy the proposed DEFEAT system: the on-site deployment and the off-site deployment. Here, we compare the response time of these two types of deployments. Specifically, we utilize two computers to simulate the web service provider and the security provider, respectively. For the on-site deployment, FaceNet and DEFEAT are installed on the same computer. For the off-site deployment, FaceNet and DEFEAT are installed in separate computers, whereby the feature vectors generated by FaceNet in one computer will be sent to DEFEAT in the other computer to conduct the attack detection. Once the detection is completed, the detection result will be sent back to the first computer that mimics the web service provider. In both types of deployments, we vary the number of photos received by the service provider from 2,000 to 10,000. Note that we use only a single thread in this test. The number of photos can be easily scaled up when multiple server nodes are adopted since they can use the same detection model for authentication and attack detection after the training is completed.

Figure 15 reports the average response time per face authentication, i.e., photo validation. As expected, the off-site deployment needs a little longer response time because of the network delay caused by the transmission of the feature vectors and decisions between the web service provider and the security provider. However, the network delay adds only 0.1% more response time compared to that of the on-site deployment. The main reason is that the sizes of feature vectors and detection results are only a few bytes per user. We also observe that the response time per photo stays relatively constant when the total number of photos increases. Note that the seemly big fluctuation of the curve is mainly due to the zoom-in effect used to show the slight gap between the two deployment methods. The average response time in different sizes of datasets is around 70ms. This is because the size of the DEFEAT model is not determined by the total number of photos, and hence the individual photo validation time is not affected by the data size. The result also demonstrates the potential scalability of our approach.

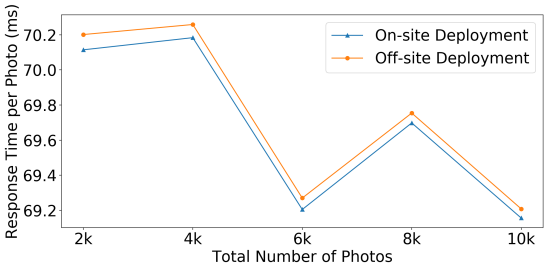


Fig. 15: Comparison of On-site and Off-site Deployment

6 Security Analysis

In this section, we analyze the security assumptions and features of our proposed discriminators.

First, we evaluate the practicability of the data poisoning attack as presented in Section 3. This attack does not require the attacker to compromise the server at the service provider side which is typically much better protected than home computers and home networks. The attacker only needs to be able to inject his/her own photos into the targets. No other knowledge is required to have a successful attack. This attack could be accomplished by the man-in-the-middle attack which is still a critical security problem in many home routers as of 2020 [48]. Once the MITM attack succeeds, the attacker can replace the target's photos with his/her own without being noticed by the target or the service provider. It is worth noting that the attack can happen not just during the new user registration phase but also during the user update phases since facial authentication systems need to be re-trained from time to time in order to function with facial changes due to age or facial hair. As we have seen from our experiments, once the injection is completed, the attackers can easily impersonate the targeted victim in future facial authentication. Since the victim's access to his/her account is not affected by the attacker's injection, the victim may not notice this until harm is done.

Even if the web applications return photos for users to validate, there are still several scenarios that the attacker may be unnoticed by the user. Recall that the training phase takes a series of photos while the attacker only needs to inject a couple of photos. The first scenario is simply a careless user who does not carefully look through the bunch of photos returned by the web applications to identify the one or two wrong photos and easily clicked the approval button. Considering that many security breaches are actually caused by human negligence, such a scenario is very likely to happen. The second scenario will almost guarantee the success of the attack. The attacker is intercepting the communication channel during the training phase. As the attacker is able to inject photos, he is also able to drop the packages containing his own photos returned by the web applications, which means the user will not see the attacker's photos during the validation. Moreover, most of the existing face recognition services do not store the actual images but only face features for privacy protection. For example, Microsoft face service clearly stated "No image will be stored. Only the extracted face feature(s) will be stored on server" [33]. It is thus hard for the users to verify their registered information later on.

Next, we discuss the security guarantees offered by our proposed discriminator. From the empirical studies on real datasets, we have seen that our discriminator achieved more than 90% detection accuracy. Although it is still not perfect, our discriminator does capture a significant amount of potential threats without requiring any prior knowledge of attackers' information. We would also like to mention that the false positives reported by our discriminators are very low. Our DEFEAT discriminator has between a 0% and 1.11% false-positive rate considering both datasets and attack strategies. Our statistic-based discriminator has between 0% and 2.04% false-positive rate. This indicates one advantage of our discriminators in that we will not raise too many false alarms to affect the normal usage of unattacked users.

Another important security advantage of our proposed DEFEAT discriminator is that it would still be robust against attackers who know the mechanism of the DEFEAT. Our DEFEAT system does not need to be a black box to the adversary. This is because it is already hard for an attacker to attack a DNN. As our DEFEAT is another DNN followed by the FaceNet DNN, it makes it even harder for the attacker to craft photos which need to satisfy two DNNs. First, the attacker needs to modify photos so that FaceNet DNN will label them as the target user's label. Second, the same set of photos needs to be able to fool the DEFEAT DNN to let it produce a low probability of infection while the photos also need to guarantee correct statistical measurements as noninfected photos. To sum up, we expect that attacking two concatenated DNNs would be a very challenging if not impossible task to attackers. This is also why we adopt DNN as the key structure of our discriminator.

7 Conclusion

In this paper, we discuss a new potential threat that can compromise facial authentication systems at web service providers. The attack can be easily implemented to impersonate a user and gain full access to the user's web service account without raising alarms to either the user or the service provider. There is no known defense mechanism against such an attack. Therefore, we propose novel detection mechanisms that leverage both statistic measurements and deep neural networks. The extensive experimental results have demonstrated that our proposed discriminators achieve very high detection accuracy in real datasets.

References

[1] Shumeet Baluja and Ian Fischer. Learning to attack: Adversarial transformation networks, 2018.

[2] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1467–1474, 2012.

[3] BioID. https://www.bioid.com/.

[4] Avishek Joey Bose and Parham Aarabi. Adversarial attacks on face detectors using neural net based constrained optimization. *CoRR*, abs/1805.12302, 2018.

[5] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018.

[6] Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 6977–6987. Curran Associates, Inc., 2017.

[7] J. Clements and Y. Lao. Backdoor attacks on neural network operations. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1154–1158, 2018.

[8] Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[9] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020.

[10] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[11] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C. Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 113–125, New York, NY, USA, 2019. Association for Computing Machinery.

[12] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[13] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2015.

[14] Gaurav Goswami, Nalini Ratha, Akshay Agarwal, Richa Singh, and Mayank Vatsa. Unravelling robustness of deep learning based face recognition against adversarial attacks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017.

[16] Jamie Hayes and George Danezis. Machine learning as an adversarial service: Learning black-box adversarial examples. 08 2017.

[17] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[18] M. Juuti, S. Szyller, S. Marchal, and N. Asokan. Prada: Protecting against dnn model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 512–527, 2019.

[19] C. Kanbak, S. Moosavi-Dezfooli, and P. Frossard. Geometric robustness of deep networks: Analysis and improvement. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4441–4449, 2018.

[20] Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S. Gunawi, Cody Hammock, Joe Mambretti, Alexander Barnes, François Halbach, Alex Rocha, and Joe Stubbs. Lessons learned from the chameleon testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*. 2020.

[21] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016.

[22] Alexey Kurakin, J. Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *international conference on learning representations*, 2017.

[23] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. Defending against model stealing attacks using deceptive perturbations. *ArXiv*, abs/1806.00054, 2018.

[24] Dan Lin, Nicholas Hilbert, Christian Storer, Wei Jiang, and Jianping Fan. Uface: Your universal password that no one can see. *Computers & Security*, 77:627–641, 2018.

[25] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In Michael Bailey, Thorsten Holz, Manolis Stamatogiannakis, and Sotiris Ioannidis, editors, *Research in Attacks, Intrusions, and Defenses*, pages 273–294, Cham, 2018. Springer International Publishing.

[26] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *NDSS*, 2018.

[27] Yuntao Liu, Yang Xie, and Ankur Srivastava. Neural trojans. *CoRR*, abs/1710.00942, 2017.

[28] Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 641–647, 2005.

[29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations*, 2018.

[30] I. Masi, Y. Wu, T. Hassner, and P. Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 471–478, 2018.

[31] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 135–147, 2017.

[32] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *ArXiv*, abs/1702.04267, 2017.

[33] Microsoft. Face api - v1.0. *https://westus.dev.cognitive.microsoft.com/docs / services/ 563879b61984550e40cbbe8d/ operations/ 563879b61984550f30395236*.

[34] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 86–94, 2017.

[35] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. *CoRR*, abs/1511.04599, 2015.

[36] Konda Reddy Mopuri, Utsav Garg, and R. Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. *CoRR*, abs/1707.05572, 2017.

[37] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. *CoRR*, abs/1708.08689, 2017.

[38] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[39] Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I. P. Rubinstein, Udam Saini, Charles Sutton, J. D. Tygar, and Kai Xia. Exploiting machine learning to subvert your spam filter. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, 2008.

[40] Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I. P. Rubinstein, Udam Saini, Charles Sutton, J. D. Tygar, and Kai Xia. *Misleading Learners: Co-opting Your Spam Filter*, pages 17–51. Springer US, Boston, MA, 2009.

[41] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[42] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.

[43] Luana Pascu. Acronis reports critical flaws in geovision biometric devices, man-in-the-middle attack risks. *BiometricUpdate*, 2020.

[44] Andrea Paudice, Luis Muñoz-González, Andras Gyorgy, and Emil C Lupu. Detection of adversarial training examples in poisoning attacks through anomaly detection. *arXiv preprint arXiv:1802.03041*, 2018.

[45] Alison DeNisco Rayome. Why do so many wireless routers lack basic security protections? *TechRepublic. https://threatpost.com/asus-home-router-bugs-snooping-attacks/157682/*, 2019.

[46] Sayantan Sarkar, Ankan Bansal, Upal Mahbub, and Rama Chellappa. UPSET and ANGRI : Breaking high performance image classifiers. *CoRR*, abs/1707.01159, 2017.

[47] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[48] Tara Seals. Asus home router bugs open consumers to snooping attacks. *ThreatPost*, 2020.

[49] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *CoRR*, abs/1804.00792, 2018.

[50] Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[51] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540, New York, NY, USA, 2016. Association for Computing Machinery.

[52] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[53] J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[54] Octavian Suciu, Radu Mărginean, Yiğitcan Kaya, Hal Daumé, and Tudor Dumitraş. When does machine learning fail? generalized transferability for evasion and poisoning attacks. In *Proceedings of the 27th USENIX Conference on Security Symposium*, pages 1299–1316, USA, 2018. USENIX Association.

[55] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.

[56] Anurag Tagat. Online fraud: too many accounts, too few passwords.

[57] Carlos Eduardo Thomaz and Gilson Antonio Giraldi. A new ranking method for principal components analysis and its application to face image analysis. *Image and Vision Computing*, 28(6):902–913, 2010.

[58] Alex Walling. Top 10 facial recognition apis & software of 2020.

[59] B. Wang and N. Z. Gong. Stealing hyperparameters in machine learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 36–52, 2018.

[60] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[61] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 6586–6595, 2019.

[62] Gregory Wittel and Shyhtsun Wu. On attacking statistical spam filters. 01 2004.

[63] Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. Evading real-time person detectors by adversarial t-shirt. *CoRR*, abs/1910.11099, 2019.

[64] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *CoRR*, abs/1704.01155, 2017.

[65] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[66] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.