IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 3 November 2022, accepted 23 November 2022, date of publication 28 November 2022,
date of current version 1 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3225430

RESEARCH ARTICLE

Analysis on Secure Triplet Loss

BORA JEONG, SUNPILL KIM, SEUNGHUN PAIK, AND JAE HONG SEO
Department of Mathematics, Research Institute for Natural Sciences, Hanyang University, Seoul 04763, Republic of Korea

Corresponding author: Jae Hong Seo (jaehongseo@hanyang.ac.kr)

This work was supported in part by the Institute of Information and Communication Technology Planning and Evaluation (IITP) grant funded by the Korea Government [Ministry of Science and ICT (MSIT)] (A Study on Cryptographic Primitives for Succinct Non-Interactive Argument of Knowledge (SNARK), 50%) under Grant 2021000727; and in part by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIT), 50%, under Grant 2020R1C1C1A01006968.

ABSTRACT Major improvements in biometric authentication have been made in recent years due to the advancements in deep learning. Through the use of a deep learning-based facial recognition model with metric learning, more discriminative facial features can be extracted from faces. A large threat to user privacy could result from the disclosure of more discriminatory feature vectors related to biometric information. Among many biometric template protection (BTP) schemes, there have been studies that have attempted to protect feature vectors from the learning process of facial recognition models, while considering security requirements. One of them is secure triplet loss (STL) based BTP, which is an end-to-end BTP scheme using deep learning model that merges an additional layer on a pre-trained facial recognition model. STL-based BTP takes a pre-defined key and an image as inputs, and it is designed to become closer only when both the identity and the key are matched simultaneously. In this paper, we propose an efficient impersonation attack algorithm on STL-based BTP and our impersonation attack algorithm is conducted in a black-box setting using only the similarity scores between a target template and the template from the queried image and key pair. We have succeed in the impersonation attack using approximately 329.59 and 256.57 queries for the two types of black-box target systems. Furthermore, we conduct an analysis of our impersonation attack algorithm along with the implementation code.

INDEX TERMS Authentication, biometrics, face recognition, impersonation attack.

I. INTRODUCTION

Biometric authentication systems have been widely used in applications. Although it offers users convenience, leakage of the original biometric information raises numerous security and privacy concerns. In this paper, we propose an impersonation attack algorithm against one of these biometric authentication systems.

The original biometric information is transformed into a biometric template and entered into the system. The biometric templates must include the features of the subject because the system evaluates the similarity across templates to establish whether they are generated from the same identity; that is, they belong to the same person. Therefore, extracting the more discriminative features from the biometric information is crucial for the performance of biometrics. Recently, deep neural networks (DNNs) have shown good performance as feature extractors and hence DNN-based biometrics are rapidly growing, especially for facial recognition systems [1], [2], [3]. In facial recognition systems, a network preserves similarity between facial images while it maps facial images to feature vectors. The distance between feature vectors can be used to interpret similarity between the feature vectors.

However, registering these feature vectors as biometric templates in the system without any security remains insufficient. Several attack methods [4], [5], [6] were reported to reconstruct the original face images from compromised templates. If an adversary can steal a deep face template then it is quite possible to reconstruct a fake image from the stolen template [4], [5]. In contrast to the former, the attacker's capabilities can be weakened. Reference [6] showed that the adversary can impersonate the target biometric authentication system only with the ability to steal similarity scores. Since the majority of biometric authentication systems use

The associate editor coordinating the review of this manuscript and approving it for publication was Zahid Akhtar.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

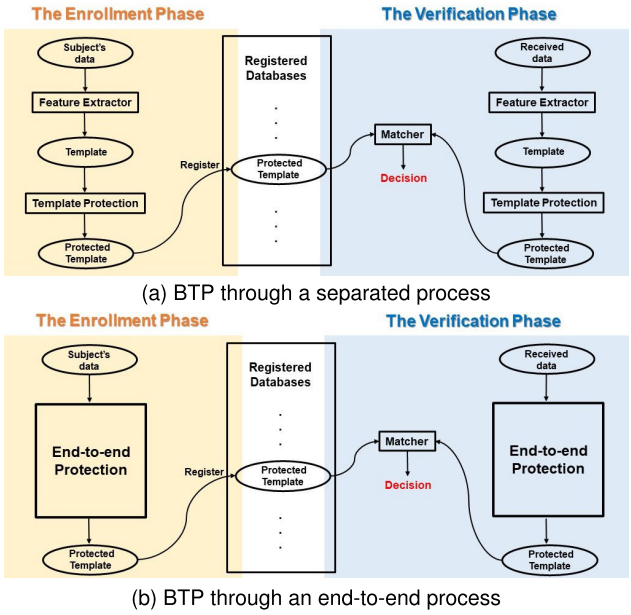


FIGURE 1. Biometric template protection through a separated or end-to-end process.

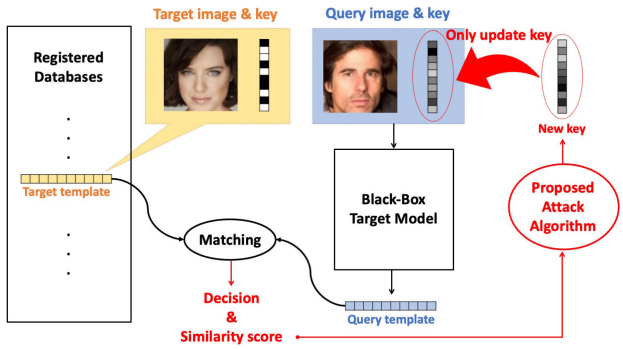


FIGURE 2. Overview of our impersonation attack algorithm that finds a pair of the image and key, which can pass the verification test of the target system with high probability. The attacker can impersonate the subject using the similarity scores received from the black-box target system.

a similarity score between an input and the template database to determine whether they are coming from the same person, we consider an attacker's ability to steal only similarity scores in our attack scenario.

Biometric templates must be highly protected due to the aforementioned security issues, and biometric template protection (BTP) schemes have been studied for decades to alleviate such issues. There are three standard security requirements for BTPs (ISO/IEC 24745:[1])

- **Irreversibility**: It is computationally infeasible to recover original biometric information from compromised templates.
- **Revocability/Cancelability**: The protected templates can be replaced by a new protected template if compromised.
- **Unlinkability**: It is computationally impossible to retrieve the biometric information from the protected template for the same subject when the subject is registered on different systems.

Although DNN-based facial recognition systems have demonstrated good performance to identify people, numerous biometric template protection systems frequently result in performance degradation. It is still challenging to protect biometric templates that satisfy the requirements while preserving their performance. Hence, there are several efforts to make an end-to-end BTP scheme as shown in the Fig. 1, by training DNNs that is directly mapping the facial images to the protected templates [7], [8], [9], [10].

Although the approach in [10] is rather novel, showing good performance, we raise a question about whether neural networks can learn aforementioned security requirements. In particular, it does not employ any cryptographic or information-theoretic tools, although a secret key is used. Thus, careful analysis is necessary for the security argument of such a DNN-based BTP in [10]. In this paper, we propose an impersonation attack algorithm against [10], one of the end-to-end BTP schemes using neural networks without additional cryptographic tools.

A. CONTRIBUTIONS

In our attack scenario, we investigate a DNN-based BTP trained by the secure triplet loss (STL), a recently proposed end-to-end BTP scheme that claims to satisfy all three security requirements with only minor performance degradation [10]. STL-based BTP receives a pair of the facial image and the user-specific key as an input and maps directly to a protected template. The authors of [10] expect relatively higher security due to the additional user-specific key and relatively small performance degradation since the user-specific key is well combined with the feature extraction network without any quantization that often causes severe efficiency loss.

The overview of our impersonation attack algorithm is presented by Fig. 2. We consider the attacker who can only obtain similarity scores between the adversarial queries and the target template in the black-box model. Numerous studies have considered such an attacker [6], [11], [12]. We note that the attacker in our scenario is weaker than typical inversion attackers [4], [5] since the similarity score can always be computed from the compromised template. First of all, the attacker fixes an arbitrary facial image regardless of the target's facial image, and sets a randomly generated key as an initial key. The pair of the fixed facial image and the initial key is entered into the black-box target system, and then the attacker can obtain the similarity score between the target template and the template from the queried pair. For convenience, the template from the queried pair is referred to as a query template in the Fig. 2. We demonstrate that

[1] https://www.iso.org/standard/75302.html

the attacker can successfully impersonate the subject using the similarity scores received from the black-box target system. Precisely, we propose an impersonation attack algorithm that finds a pair of the image and key, which can pass the verification test of the target system with high probability. Our experiments guarantee the practicality of the proposed impersonation attack algorithm.

II. BACKGROUND

A. SIMILARITY SCORE IN FACIAL RECOGNITION SYSTEMS

Facial recognition systems measure the similarity score between templates. The system decides acceptance by comparing the similarity scores with a pre-determined threshold in the verification phase. Numerous facial recognition systems have been proposed in various ways, but in general, they transform the given facial images into templates, called feature vectors.

The similarity can be measured by the distance of such feature vectors. It can be expected that the two input images come from the same subject if the distance between two feature vectors is closer than the pre-determined threshold. If the similarity score is defined as the distance and the similarity score between the two feature vectors is smaller than the pre-determined threshold, then the systems become to decide that the two feature vectors were extracted from the images of the same subject. To accomplish this, the loss function should be designed to minimize the intra-class distance and maximize inter-class discrepancy. Metric learning is one of the notable approaches, which has led to significant performance growth in facial recognition systems [1], [2], [3].

B. TRIPLET LOSS

The triplet loss [1] is the most prominent example of metric learning and has been widely used in practice. The basic idea of the triplet loss is the following; facial images are sampled as a triplet, which consists of three facial image called anchor, positive sample, and negative sample. From this, the objective of the triplet loss function is to minimize the Euclidean distance between anchor & positive samples and to maximize the distance between anchor & negative sample. Formal description of the triplet loss is given as follow:

$$l = max\{0, \alpha + |f(x_A) - f(x_P)| - |f(x_A) - f(x_N)|\} \quad (1)$$

where x_A, x_P and x_N are an anchor, a positive sample, and a negative sample, respectively, and α is a margin to facilitate the desired objective.

C. SECURE TRIPLET LOSS

Inspired by the triplet loss, the STL-based BTP is proposed by [10]. STL is a loss function designed to train an end-to-end BTP using deep learning model with metric learning. STL-based BTP is a two-factor DNN-based authentication system since it takes not only the user's biometric data but also the user-specific key as input and then the DNN outputs the protected template. The architecture of STL-based BTP is provided in Fig. 3.

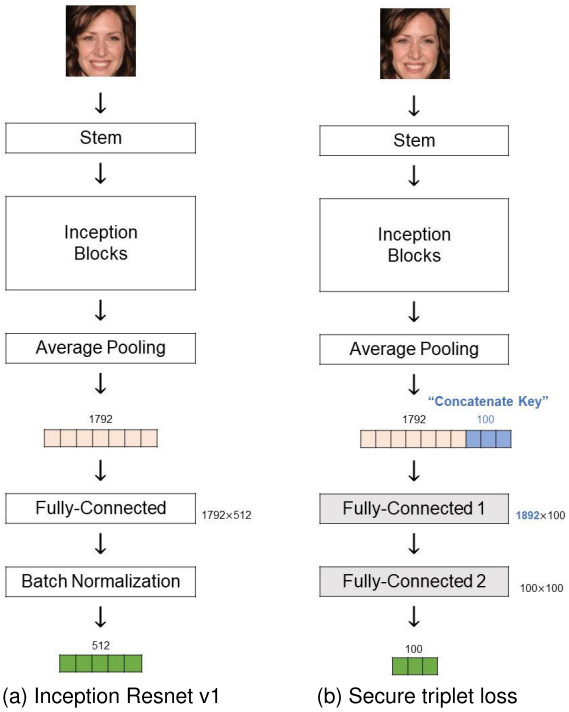


FIGURE 3. Network architecture of STL-based BTP (b) and its baseline network Inception-Resnet-v1 (a).

The objective of the STL is to reduce the distance between outputs when not only the identity of the biometric data but also the user-specific key from each input pairs is matched. There are four types of distances, both positive identity and key, positive identity but negative key, negative identity but positive key, and both negatives, which is summarized in Table 1. For each distance, the first one d_{SP} (both positives) must be minimized, and the remainders must be maximized. From this, the authors of [10] proposed two loss functions l_C and l_L for cancelability and unlinkability, respectively.

TABLE 1. Distances in the secure triplet loss [10].

	Same Key	Different Key
Positive ID	d_{SP}	d_{DP}
Negative ID	d_{SN}	d_{DN}

They contended that cancelability is attained by minimizing d_{SP} when both identity and key are matched, and by maximizing remainders. The loss function for cancelability, l_C, is the following.

$$l_C = max[0, \alpha + d_{SP} - min(d_{DP}, d_{SN}, d_{DN})] \quad (2)$$

where α is a margin to facilitate the desired objective. In addition, they added a loss component, l_L, to enhance unlinkability. They calculated the mean and standard deviation of distances when keys are different since it should be challenging to differentiate between d_{DP} and d_{DN}.

$$l_L = |\mu(d_{DP}) - \mu(d_{DN})| + |\sigma(d_{DP}) - \sigma(d_{DN})|$$

where μ is a mean of distances and σ is a standard deviation of distances. The final loss function to enhance unlinkability, l_{STL}, includes above two loss components l_C and l_L with balance parameter γ.

$$l_{STL} = \gamma l_C + (1-\gamma) l_L \tag{3}$$

where γ is a hyperparameter to balance the loss components l_C and l_L.

III. PROPOSED ATTACK ALGORITHM

A. TARGET MODEL

In this paper, we propose an impersonation attack algorithm against STL-based BTP. The structure of the network is divided into two parts. The first part is a pre-trained feature extractor, which outputs the feature vector from the input image. The second part is additional layers to bind the key with the feature vector, which consists of two fully connected layers. In detail, the feature vector from the first part is concatenated with the user-specific key. The keys are binary vectors in $\{0,1\}^{100}$ and normalized before concatenation. After this, passing through the second part, STL-based BTP finally outputs the protected template. Figure 3 illustrates the overall structure of the network.

Following the loss functions proposed by [10], our target models are two types; STL_C and STL_L. The two target models have the same structure, but they are trained by the loss function with different hypherparameter γ in (3). For STL_C, γ is set to 1 and it indicates that the loss function has only containing a loss component l_C. For STL_L, γ is set to 0.9, which is recommended to achieve best performance by the author of [10]. Comparing the two target models, STL_C has better performance than STL_L. Therefore, we set up the two target models expecting that the attack performance would be different.

B. ATTACK SCENARIO

We present our attack scenario in this section, and the following section presents the details of our impersonation attack algorithms. The attacker's ability and main goal are as follows:

- **Attacker's ability:** The attacker has access to the target model as a black-box. That is, the attacker can query pairs of image and key to black-box target model and obtain similarity scores between the target template and the templates which are extracted from the queries.
- **Attacker's goal:** The goal of the attacker is impersonation. This indicates that a query is accepted by the target model even when one of the image or the key does not match to target template's one.

The strategy of our impersonation attack is the following. The attacker starts by selecting any facial image and creating a random key. The attacker then query this pair of image and key to the system, obtaining the similarity score between the query and the target template. Note that the chosen image is fixed during our attack while updating the key through our proposed algorithms. The process of updating the key is repeated until the attacker finds a pair of the image and key that causes impersonation for the target template.

C. ATTACK ALGORITHM

We assume that the target template t from the subject whom the attacker attempt to impersonate is stored at the black-box target model $\mathbf{T}$, and the attacker can obtain the similarity score s between the target template t and the template from the queried pair (img, k) of image and key. The proposed impersonation attack algorithm is presented in Algorithm 1.

Algorithm 1 Attack Algorithm for the Real-Valued Key

Require: target model $\mathbf{T}$, image img, scale factor $\alpha \in \mathbb{R}$
Ensure: Key $k \in \mathbb{R}^d$

```
Initialize k ←$ [0, 1]^d
Query (img, k) to T and get score s ← T(img, k)
while T rejects the query do
    Set noise ← N(0⃗, I_d)
    Set k' ← k + α × noise
    Query (img, k') to T and get score s' ← T(img, k')
    if s' < s then
        Update s ← s' and k ← k'
    end if
end while
Return k
```

In Algorithm 1, an initial key k is set to a real-valued vector uniformly sampled from $[0,1]^d$. The attacker queries (img, k) to the black-box target model $\mathbf{T}$ and obtains the decision and a score s. If the decision is rejection, then the attacker randomly samples a noise from Gaussian distribution[2] $\mathcal{N}(\vec{0}, I_d)$ and scaled the noise by a factor of α to make a new key k' by adding the scaled noise to the previous key k. The attacker queries (img, k') to the $\mathbf{T}$ and obtain the decision and a new score s'. If the decision is rejection and the new score s' is smaller than the previous score s, the attacker updates the key k to the new key k'. Note that, in this case, the similarity score is defined as the distance between the templates. The attacker repeats the process until the $\mathbf{T}$ accepts the queried pair. Finally, the attacker can obtain the key to impersonate the target subject with an arbitrary facial image regardless of the target subject.

As the results will be discussed in Section IV, Algorithm 1 is successful to impersonate the target subject. But the target model can be made to reject the real-valued vectors to defend against such impersonation attacks which uses Algorithm 1. Note that the STL-based BTP generates binary vectors as keys and the keys are converted into real-valued vectors by the normalization.

For this reason, we add the assumption that the attacker can query the target model only with binary keys and

[2] Although the impersonation attack is successful using other distributions, we will only describe the Gaussian distribution because we experimentally confirm that the Gaussian distribution succeeds in the impersonation attack with the least queries.

Algorithm 2 Attack Algorithm for the Binary Key

Require: target model $\mathbf{T}$, image img, number of flip $n \in \{1, \ldots, d\}$
Ensure: Key $k \in \{0, 1\}^d$

```
Initialize k <-$ {0,1}^d
Query (img, k) to T and get score s <- T(img, k)
while T rejects the query do
    Set k' <- RFlip(k, n)
    Query (img, k') to T and get score s' <- T(img, k')
    if s' < s then
        Update s <- s' and k <- k'
    end if
end while
Return k
```

Algorithm 3 RFlip Algorithm

Require: $k = (k_1, \ldots, k_d) \in \{0, 1\}^d$, $n \in \{1, \ldots, d\}$
Ensure: $k' = (k'_1, \ldots, k'_d) \in \{0, 1\}^d$

```
Randomly sample n indices i_1, ..., i_n from {1, ..., d}
and set I := {i_1, ..., i_n}
for i ∈ {1, ..., d} do
    if i ∈ I then
        Set k'_i <- k_i + 1 (mod 2)
    else
        Set k'_i <- k_i
    end if
end for
Return k'
```

propose another Algorithm 2 which only exploits binary keys. In Algorithm 2, an initial key k is set to a binary vector uniformly sampled from $\{0, 1\}^d$. The attacker queries (img, k) to the black-box target model $\mathbf{T}$ and obtains a score s in the same manner of Algorithm 1. Then, the attacker makes a new key by randomly flipping bits of the key through Algorithm 3 instead of sampling noise. The remaining processes are identical to those of Algorithm 1. The overall algorithm for the binary key is given in Algorithm 2 and 3.

IV. ATTACK RESULTS

A. EXPERIMENTAL SETTINGS

To validate our impersonation attack algorithm, we experiment with the same circumstance as that in [13]. Precisely, for the recognition backbone architecture, we employed Facenet with the structure of Inception-Resnet-v1 pretrained[3] on vggface2 [14] and fine-tuned each networks in YTF-Databases [15] in the same manner as in [10]. Then, based on this architecture, STL_C attains 13.44% equal error rate (EER) with a 0.0261 threshold, and STL_L attains 14.95% EER with 0.0174 threshold.[4]

[3]https://github.com/timesler/facenet-pytorch

[4]Note that, for simplicity, we used a pre-determined threshold and our impersonation attack can be easily generalized for the hidden threshold scheme since our impersonation attack does not rely on the knowledge of the threshold at all.

TABLE 2. Average of Queries required and Success rates in Algorithm 1.

α	STL_C		STL_L	
0.01	952.66 queries	97.4%	662.54 queries	98.6%
0.02	523.33 queries	99.0%	416.80 queries	99.4%
0.03	435.89 queries	99.4%	326.13 queries	99.4%
0.04	362.54 queries	99.8%	283.25 queries	99.8%
0.05	**329.59** queries	99.8%	**256.57** queries	99.8%

TABLE 3. Average of queries required and success rates in Algorithm 2.

n	STL_C		STL_L	
1	720.38 queries	97.0%	565.58 queries	98.4%
2	689.68 queries	96.4%	**490.79** queries	97.6%
3	**649.17** queries	98.2%	498.56 queries	98.2%
4	719.17 queries	97.2%	569.81 queries	97.4%
5	822.25 queries	97.4%	565.76 queries	98.0%

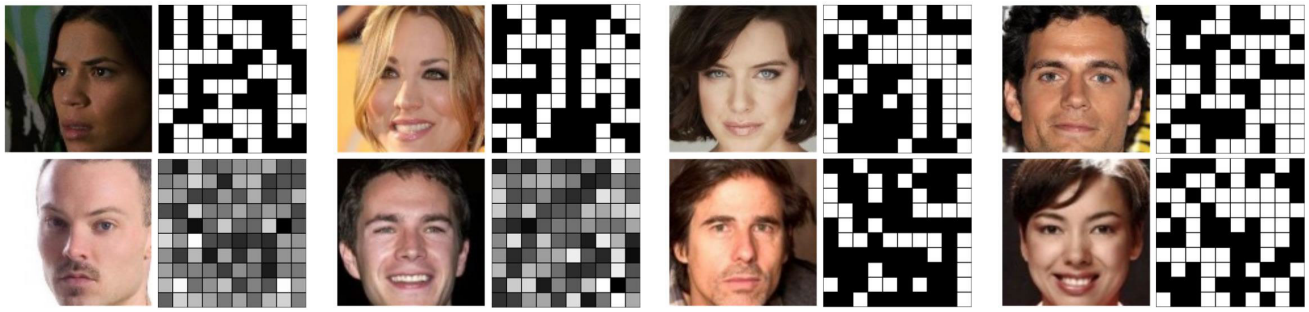
We attempted to impersonate the 500 target subjects. Each subject has a pair of the facial image and the user-specific key. The target facial images were selected from the well-known face database, CASIA-Webface [16]. Each user-specific keys was uniformly sampled from $\{0, 1\}^{100}$ and normalized, following the setting in their work [13]. In addition, we experimented with increasing the scale factors α from 0.01 to 0.05 in Algorithm 1 and increasing the number of flip n from 1 to 5 in Algorithm 3.

B. EXPERIMENTAL RESULTS

For the two target models, we calculated the average number of queries required from 500 trials of each impersonation attack. The results are presented in Tables 2 and 3. We chose any facial image independent of the target identity to demonstrate that false acceptance happens when both the identity and key are unmatched. As a result, we found pairs that impersonated the system. The obtained pairs are shown in Fig. 4.

In Algorithm 1, we experimented on the different scale factors α and achieved the best results when $\alpha = 0.05$. We discovered that a pair of the image and key is accepted by the target model for the target template after approximately 329.59 and 256.57 queries on STL_C and STL_L, respectively, when $\alpha = 0.05$. The target model STL_C requires more queries than STL_L. Since the STL_C has better verification performance than the STL_L, the protected templates from the former has a low tendency to rely on the key than the latter. Note that STL-based BTPs take a pair of an image and a key and our impersonation attack algorithm only uses the key update.

In Algorithm 2, we experimented on the different numbers of flip n for the binary vectors. According to the choice of n, we achieved the best results, which take approximately 649.17 queries on STL_C for $n = 3$ and 490.79 queries on STL_L for $n = 2$. In both algorithms, the final key for



(a) Algorithm 1 / STL_C (b) Algorithm 1 / STL_L (c) Algorithm 2 / STL_C (d) Algorithm 2 / STL_L

FIGURE 4. For each subfigure, the pairs of facial image and key are recognized as the same identity and key pair on the target model. the above pair is the target's facial image and the key, and the below one is the pair of the query image and retrieved key via our impersonation attack algorithms. For visualization, we reshaped each 100-dimensional key into a 10 × 10 matrix.

successful impersonation was entirely different from that of the target template. Remark that neither the identity nor the key were matched since we fixed the negative identity with the target, as shown in Fig. 4.

Sparsely, the queries were required much more than the average since the our impersonation attack algorithm can be interpreted as an optimization problem to minimize the score obtained from the target model. The choice of starting point is crucial for such an optimization problem. If a starting point, an initial image and key pair in our context, lies nearby a local minimum, then the attacker would fail to impersonate the target model. To alleviate this problem, the attacker may try other starting points when the algorithm does not halt within some fixed number of iterations. In our experiments, we limited the number of iteration to 1000 for choosing other starting point. We decided that the impersonation attack is successful when the number of updates in starting point is less than 10. As we showed in the Table 2 and 3, our impersonation attack achieves the attack success rate 99.8% and 98.4% in Algorithm 1 and 2, respectively.

C. ANALYSIS ON THE STL

From the results of our impersonation attack experiments, it was possible to occur the false acceptance by changing only the key. This implies that protected templates are influenced by the 100 dimensional keys rather than the facial images. We found the reason in the last two layers of the target models.

The keys in the target models are only impacted by the last two layers. Two fully connected layers are added following the concatenation of the feature vector and key, and then ReLU functions are used (FC-ReLU-FC-ReLU). We call the first fully connected layer FC_1, the second one FC_2. Then, the final protected template t can be represented through the following equation:

$$\mathbf{t} = R \circ FC_2 \circ R \circ FC_1(X) \tag{4}$$

where $R(x) = max\{0, x\}$ is the ReLU function and $X \in \mathbb{R}^{1892}$ is the input vector of FC_1.

Let $FC_1(x) = Wx + \mathbf{b}$ and $FC_2(x) = \overline{W}x + \overline{\mathbf{b}}$, where the weight matrices $W \in \mathbb{R}^{100\times1892}$, $\overline{W} \in \mathbb{R}^{100\times100}$ and biases $\mathbf{b}, \overline{\mathbf{b}} \in \mathbb{R}^{100}$. Let $\mathbf{x} \in \mathbb{R}^{1792}$ is the feature vector from the image and $\mathbf{k} \in \mathbb{R}^{100}$ is the normalized key. Before the FC_1, $\mathbf{x}$ and $\mathbf{k}$ are concatenated to the 1892-dimensional vector $X = (\mathbf{x}, \mathbf{k})$. Then, (4) is represented as follows:

$$\begin{aligned}\mathbf{t} &= R(\overline{W} \times R(WX + \mathbf{b}) + \overline{\mathbf{b}}) \\ &= R(\overline{W} \times R(W \times (\mathbf{x}, \mathbf{k}) + \mathbf{b}) + \overline{\mathbf{b}})\end{aligned} \tag{5}$$

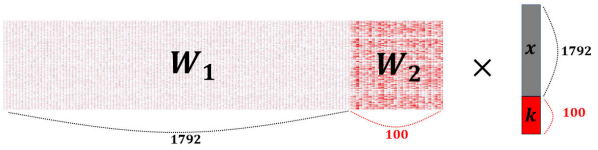


FIGURE 5. Weight matrix W of the FC_1 and vector multiplied by the matrix. The entries of the matrix W were colored in darker red as the absolute value was larger so that W can be separated by the left 1792 columns W_1 and right 100 columns W_2. The gray part of the vector is the feature vector x extracted from the image, and the red part of the vector is the normalized key k.

We printed out the elements of the weight matrix W in (5) and confirmed the weights with large absolute values are concentrated in the last 100 units. For the absolute values of weights, the mean weights of W is approximately 0.009. However, the mean value is approximately 0.06 in the last 100 units, while approximately 0.006 in the front 1792 units. In the front 1792 units, the weights are close to zero, and this consequently reduces the influence of the preceding 1792-dimensional vector $\mathbf{x}$ extracted from the image. This implies that the network becomes more focused on the last 100 vector $\mathbf{k}$, key. We visualize the absolute values of weight for easy understanding in Fig. 5. The larger the absolute value of the element, the darker the red color. The elements colored red are concentrated in the last 100 units; that is, the absolute values of weights are large in the last 100 units.

In addition, both of the two target models STL_C and STL_L have a tendency to rely on the 100 dimensional key $\mathbf{k}$ rather than the feature vector $\mathbf{x}$. However, STL_C has better verification performance than STL_L and requires more queries in the attack results. This indicates that the influence of the feature vector $\mathbf{x}$ is relatively large in STL_C. We found the

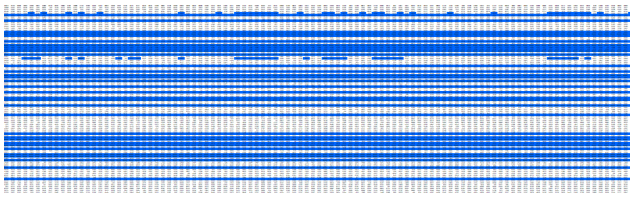


FIGURE 6. Positions of the negative numbers after the FC_2. Each column is a 100-dimensional vector obtained from FC_2 before the ReLU activation layer. All negative entries are colored blue, and they will become zero after the ReLU function.

reason in the norm of $\mathbf{x}$. The norm of $\mathbf{k}$ is fixed to 1 since the $\mathbf{k}$ is normalized, but the norm of the feature vector $\mathbf{x}$ is approximately 4.51 in STL_C and 3.58 in STL_L. This is why the queries required for the impersonation attack are different in the two target models.

As shown in Fig. 5, the matrix W in (5) can be separated into the first 1792 columns and last 100 columns. Let matrix W_1 be the first 1792 columns and the matrix W_2 be the last 100 columns as shown in Fig. 5, and then $W = [W_1||W_2]$. Since the entries of W_1 are close to zero, the term of W_1 does not significantly affect the templates. Therefore, we omit the term of W_1 and obtain the following approximation.

$$\begin{aligned} \mathbf{t} &= R(\overline{W} \times R(W_1\mathbf{x} + W_2\mathbf{k} + \mathbf{b}) + \overline{\mathbf{b}}) \\ \mathbf{t} &\approx R(\overline{W} \times R(W_2\mathbf{k} + \mathbf{b}) + \overline{\mathbf{b}}) \end{aligned} \tag{6}$$

Since the ReLU function affects only negative numbers, we present the intermediate vectors between the fully connected layer and ReLU activation layer in Fig. 6. Each column is a 100 dimensional vector, the intermediate vector obtained from FC_2. For the 500 different facial images and keys, the negative numbers were mostly observed in similar locations. Similar results were found in the case of FC_1. The position of zero is already decided by the fully connected layers prior to the ReLU activation layers since the positions of the negative integers are largely comparable before the ReLU function. Hence, we omit the ReLU function in (6).

$$\mathbf{t} \approx \overline{W}W_2\mathbf{k} + \overline{W}\mathbf{b} + \overline{\mathbf{b}} \tag{7}$$

$$\overline{W}W_2\mathbf{k} \approx \mathbf{t} - \overline{W}\mathbf{b} - \overline{\mathbf{b}} \tag{8}$$

This indicates that the protected templates $\mathbf{t}$ depends on the normalized key $\mathbf{k}$. Simplifying (7) with respect to $\mathbf{k}$, it becomes (8), which is the form of $Ax \approx b$ where $A \in \mathbb{R}^{100\times100}$, $b \in \mathbb{R}^{100}$ and $x \in \mathbb{R}$ is an unit vector whose entries consist of 0 or $1/\sqrt{\alpha}$ for some positive integer $\alpha < 100$. It is reduced to the optimization problem by finding a subset of column vectors of A so that the sum of column vectors is close to vector b. Such x can be easily found with a greedy algorithm in general, such as Algorithm 4. The optimization problem to find the unit vector $\mathbf{k}$ satisfying (8) is the case when $m = n$ in Algorithm 4.

For a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $b \in \mathbb{R}^m$, the goal of Algorithm 4 is to find an unit vector x satisfying Ax is closest to b. First, calculate the inner product of each column of A with b and select the column with the largest value of inner product. Then, except for the selected column, calculate the inner product of the remaining each column with b and repeat the process. Since the greedy algorithm find the local minimum of $\|Ax - b\|_2$, such x is not the optimal solution of $Ax \approx b$. If the local minimum is less than the predefined threshold of the target model, then such x will be accepted by the target model with an arbitrary image. This indicates the attack success rates and we confirmed experimentally that the attack success rate is 96.4% in the worst case.

Algorithm 4 Algorithm for the Unit Vector x

Input: $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $\mathcal{N} = \{1, \cdots, n\}$
Output: unit vector $x \in \mathbb{R}^n$

Parse A as $[a_1||\cdots||a_n]$ where $a_i \in \mathbb{R}^m$ for $i \in \mathcal{N}$
Set $\mathcal{A} \leftarrow \mathcal{N}$, $\mathcal{B} \leftarrow \emptyset$, and $v \leftarrow \{0\}^m$
for $j \in \mathcal{N}$ **do**
  Set $k' \leftarrow \text{argmax}_{k\in\mathcal{A}}\langle a_k, b\rangle$
  Set $\mathcal{A} \leftarrow \mathcal{A} \setminus \{k'\}$ and $v' \leftarrow a_{k'}$
  Set $v' \leftarrow v + \frac{v'}{\sqrt{j}}$
  if $\langle v', b\rangle > \langle v, b\rangle$ **then**
    Set $v \leftarrow \frac{\sqrt{j}}{\sqrt{j+1}}v'$ and $\mathcal{B} \leftarrow \mathcal{B} \cup \{k'\}$
    if $j = n$ **then**
      Set $x \leftarrow \{1\}^n$
      break
    end if
  else
    Set $x \leftarrow \{0, 1\}^n$ where $x_i = 1$ for $i \in \mathcal{B}$ and $x_i = 0$ for $i \notin \mathcal{B}$
    break
  end if
end for
Set $x \leftarrow \frac{x}{\|x\|}$
Return x

V. CONCLUSION

We proposed an impersonation attack algorithm against the STL-based BTP [10], one of the end-to-end BTP schemes. Although the STL-based BTP maps pairs of a facial image and the user-specific key to protected templates, the key was only impacted by the last two layers. We printed out the weight of the two layers; the weight is concentrated on the columns corresponding to the key rather than the image. This implies that the model can be under threat from the key-only attack. Hence, we proposed the two types of impersonation attack algorithm for a real-valued key and a binary key. We succeeded in the impersonation attack for a real-valued key using approximately 329.59 and 256.57 queries on the target model STL_C and STL_L, respectively. Further, we succeeded in the impersonation attack for a binary key through approximately 649.17 and 490.79 queries on the target model STL_C and STL_L, respectively.

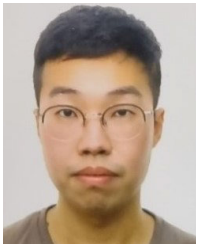
REFERENCES

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.

[2] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 212–220.
[3] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4690–4699.
[4] G. Mai, K. Cao, P. C. Yuen, and A. K. Jain, "On the reconstruction of face images from deep face templates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 5, pp. 1188–1202, May 2019.
[5] C. N. Duong, T.-D. Truong, K. Luu, K. G. Quach, H. Bui, and K. Roy, "Vec2Face: Unveil human faces from their blackbox features in face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6132–6141.
[6] A. Razzhigaev, K. Kireev, I. Udovichenko, and A. Petiushko, "Darker than black-box: Face reconstruction from similarity queries," 2021, *arXiv:2106.14290*.
[7] R. K. Pandey, Y. Zhou, B. U. Kota, and V. Govindaraju, "Deep secure encoding for face template protection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2016, pp. 9–15.
[8] A. K. Jindal, S. R. Chalamala, and S. K. Jami, "Securing face templates using deep convolutional neural network and random projection," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2019, pp. 1–6.
[9] L. Chen, G. Zhao, J. Zhou, A. T. S. Ho, and L. Cheng, "Face template protection using deep LDPC codes learning," *IET Biometrics*, vol. 8, no. 3, pp. 190–197, May 2019.
[10] J. R. Pinto, M. V. Correia, and J. S. Cardoso, "Secure triplet loss: Achieving cancelability and non-linkability in end-to-end deep biometrics," *IEEE Trans. Biometrics, Behav., Identity Sci.*, vol. 3, no. 2, pp. 180–189, Apr. 2021.
[11] A. Razzhigaev, K. Kireev, E. Kaziakhmedov, N. Tursynbek, and A. Petiushko, "Black-box face recovery from identity features," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 462–475.
[12] G. Chen, S. Chenb, L. Fan, X. Du, Z. Zhao, F. Song, and Y. Liu, "Who is real bob? Adversarial attacks on speaker recognition systems," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 694–711.
[13] J. R. Pinto. (2020). *Github Repository of the First Author of 10.* [Online]. Available: https://github.com/jtrpinto/SecureTL
[14] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age," in *Proc. 13th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG)*, May 2018, pp. 67–74.
[15] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proc. CVPR*, Jun. 2011, pp. 529–534.
[16] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," 2014, *arXiv:1411.7923*.

BORA JEONG received the B.S. degree in mathematics from Hanyang University, Seoul, Republic of Korea, in 2022, where she is planning to pursue the Ph.D. degree in applied mathematics. Her research interests include privacy-preserving machine learning, including deep learning-based biometrics.

SUNPILL KIM received the B.S. degree in mathematics from Hanyang University, Seoul, Republic of Korea, in 2020, where he is currently pursuing the Ph.D. degree in applied mathematics. His main research interests include cryptography, especially biometrics and zero-knowledge proof. He is also interested in artificial intelligence using deep learning.



SEUNGHUN PAIK is currently pursuing the B.S. degree in mathematics with Hanyang University, Seoul, Republic of Korea.

JAE HONG SEO received the B.S. degree in mathematics from Korea University, in 2004, and the Ph.D. degree from the Department of Mathematical Science, Seoul National University, in 2011.

He worked with the National Institute of Communications and Technology, Japan, from 2011 to 2013. He worked with Myongji University, South Korea, from 2013 to 2018. He has been an Associate Professor with the Mathematics Department, Hanyang University, South Korea, since 2018. His research interests include computational algorithms, cryptology, and their applications in deep learning and blockchain. He has served as a Program Committee Member for IACR conferences and workshops. He received the Sangsan Young Mathematician Award, Korean Mathematical Society, in 2012, and the Young Scientist Award for Excellence, Elsevier and NRF, in 2018. He was the Program Committee Co-Chair of APKC 2018 and ICISC 2019.