IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received February 4, 2021, accepted February 24, 2021, date of publication March 4, 2021, date of current version March 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3063693

A Statistical Inference Attack on Privacy-Preserving Biometric Identification Scheme

DONGMIN KIM[1] AND KEE SUNG KIM[2]
[1]The Affiliated Institute of ETRI, Daejon 34129, South Korea
[2]Daegu Catholic University, Daegu 38430, South Korea

Corresponding author: Kee Sung Kim (kee21@cu.ac.kr)

This work was supported by the National Research Foundation of Korea NRF Grant funded by the Korean Government through MSIT under Grant NRF-2019R1G1A1097540.

ABSTRACT Biometric identification allows people to be identified by their unique physical characteristics. Among such schemes, fingerprinting is well-known for biometric identification. Many studies related to fingerprint-based biometric identification have been proposed; however, they are based purely on heavy cryptographic primitives such as additively homomorphic encryption and oblivious transfer. Therefore, it is difficult to apply them to large databases because of the expense. To resolve this problem, some schemes have been proposed that are based on simple matrix operations rather than heavy cryptographic primitives. Recently, Liu *et al.* proposed an improved matrix-based scheme using the properties of orthogonal matrices. Despite being more efficient when compared to previous systems, it still fails to provide sufficient security against various types of attackers. In this paper, we demonstrate that their scheme is vulnerable to an attacker who operates with a cloud server by introducing statistical-inference attack algorithms. Moreover, we propose concrete identity confirmation parameters that an adversary must always pass, and present experimental results to demonstrate that our algorithms are both feasible and practical.

INDEX TERMS Biometric identification, cloud computing, privacy-preserving, statistical inference attack.

I. INTRODUCTION

Biometric identification is a convenient means of identifying users in a specific group. To identify users, we use biometric traits rather than passwords, identification cards, etc. It is sufficient to substitute them because they are always present on the person, unique, and highly invariant over time. This means that biometric traits satisfy three critical properties: universality, uniqueness, and permanence [1]. For this reason, biometric traits are widely used in identification and authentication systems in various fields. Concurrently, however, the concerns about privacy continue. If a secret key is generated from the biometric data is disclosed, it cannot be reused or replaced in the same system because of its uniqueness. In addition to this problem, there are many concerns about privacy in biometric systems [2], [3], and numerous studies have been conducted to resolve this problem. [4]–[6].

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

Most of the identification systems mentioned above require two basic algorithms, a feature extraction algorithm [7]–[10], and a matching algorithm [11]–[17]. The feature extraction algorithm is used to extract the features of biometric traits such as fingerprints, palm veins, face, and irises. In the case of fingerprints, for example, we first obtain the fingerprint image, and then, generate the n-dimensional vector, called FingerCode [18], from it by applying the relevant feature extraction algorithm. The matching algorithm is used to compare FingerCodes while preserving privacy. A comparison of whether two FingerCodes are similar is easy because it requires computing the Euclidean distance between two vectors to check for similarity. However, it is not simple to compare them while preserving privacy. Because FingerCodes must be encrypted and a general encryption scheme (such as AES or RSA) requires a decryption process to compute the Euclidean distance between them, we have difficulty checking whether two encrypted FingerCodes are identical.

To overcome this difficulty, many works related to privacy-preserving matching algorithms have been proposed. [12]–[15]. Barni *et al.* [12] proposed a privacy-preserving fingerprint authentication scheme using additively homomorphic encryption. The output of their scheme is the set of indices within some threshold, not the index having the minimum distance between the candidate FingerCode and the FingerCode in the database. By this process, the computation cost is linearly increased in proportion to the size of the encrypted data. As a result, their scheme takes 16 s and uses 9.11 MB of bandwidth for each identification request to the database (number of FingerCode=320, length of the FingerCode=16, and component size of the FingerCode=7bits). SCiFI [13] is a secure component-based face identification system that matches images captured by the client to the images stored on the server. They built a secure-hamming distance and secure-minimum algorithm based on additively homomorphic encryption and 1-out-of-N oblivious transfer as cryptographic tools. These cryptographic tools enhance the security of the protocol, but reduce its efficiency. In [13], the authors reported that the identification scheme takes 31 s of online computation for a database of 100 (a list of 100 faces representing a string of 900 bits). Consequently, this scheme is not suitable for practical biometric identification applications, because it is time-consuming to perform the identification algorithm for every request. In a similar work, Huang *et al.* [14] proposed a new protocol that improves the efficiency of the previous biometric identification schemes. They built an improved Euclidean distance protocol based on ciphertext packing techniques and use the encryption circuits to find the closest match. This scheme required 18 s with a 7.6 MB bandwidth cost per identification request on a 1 GB database. While it appears more efficient than the previous schemes, it has the problem wherein the client has to send the entire encrypted database to the server when identification is requested. Yuan and Yu [15] proposed a scheme that resolves the above problem. They use a simple matrix operation instead of heavy cryptographic tools to protect the owners' biometric information. This makes it possible to construct a practical privacy-preserving biometric identification scheme. As a result, the authors report that it takes 4.31 s for an identification request over 10 GB. It is clear that the scheme is efficient but not secure. Zhu *et al.* [19] demonstrated that the scheme is not secure against collusion attacks between query clients and cloud servers. Moreover, they provide an upper bound of the collusion-resistance ability of any accurate secure nearest neighbor (SNN) query scheme, which is a basic scheme used for most of the biometric identification schemes [20]–[22].

To overcome the vulnerability of the scheme, two improved schemes have been proposed [23], [24]. However, Liu *et al.* [25] noted that they are also insecure against known plaintext attacks (KPAs) under their security assumptions, and proposed a new privacy-preserving biometric identification scheme using the properties of orthogonal matrices and additional random numbers. To the best of our knowledge, this is the state-of-the-art scheme in the field and the most efficient. In this paper, we show that their scheme is vulnerable to an attacker who colludes with the cloud server by introducing two statistical inference algorithms. Note that many previous works [12], [26], [27] related to biometric identification using homomorphic encryption schemes [28]–[30] have been proposed; however, for practical reasons, we do not consider these schemes in this paper. Specifically, our contribution can be summarized as follows:

1) We propose statistical inference attack algorithms that can be applied to biometric identification schemes that use matrix operations with random numbers.
2) We highlight the security flaw in Liu *et al.*'s scheme by applying our algorithm. By using this vulnerability, we show that the adversary can impersonate another user with a fake fingerprint.
3) By providing concrete experimental results, we verify that our attack is practical. We also analyze each parameter in the attack algorithm, and then, propose optimized parameters efficiently to generate fake fingerprints.

The remainder of this paper is organized as follows. In Section II, we review some preliminaries to understand the scheme. In Section III, we review the scheme proposed in [25]. In Section IV, we propose the new attack algorithm and show that the scheme in [25] is not secure under the CPA model. We analyze the performance of the attack algorithm in Section V. Finally, we conclude the paper in Section VI.

II. BACKGROUND

A. SYSTEM MODEL

There are three entities defined in our scheme: a data owner ($\mathcal{DO}$), client ($\mathcal{C}$), and cloud service provider ($\mathcal{CS}$), as shown in Figure 1. We use "the server" for "the cloud server" and we use "the owner" for "the data owner" in the rest of the paper. The owner has the clients' original fingerprints in his local database and performs the identification process on the server. The clients request identification when they have to be identified by the owner. Then, the owner delegates the operations needed for identification to the server for computational efficiency. Note that the server only performs the operations requested by the owner. During this process without the results of the operations, the server does not know any sensitive information. To achieve this, most schemes are designed as follows. First, the data owner encrypts the entire database ($\mathcal{DB}$) and sends it to the server. When the clients need to be identified by the owner, they sends their fingerprint ($\mathcal{Q}$) to the owner as a candidate biometric trait. Then, the owner encrypts and sends the fingerprint to the server to determine the closest match. The server computes the Euclidean distance between it and all the stored fingerprints, and then, the server finds the closest match and sends the corresponding index to the owner. Finally, the owner decides whether the client is valid by computing the Euclidean distance between the candidate fingerprint and the fingerprint that corresponds to the index.

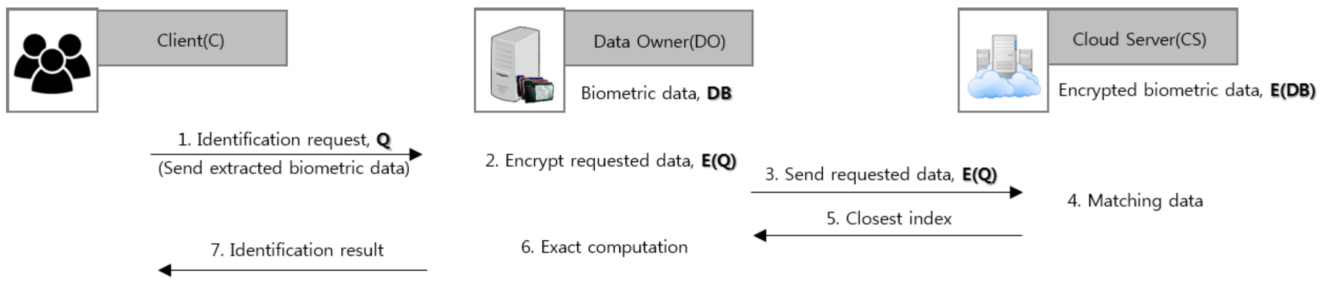

Client(C)
Data Owner(DO)
Biometric data, DB
Cloud Server(CS)
Encrypted biometric data, E(DB)
1. Identification request, Q
(Send extracted biometric data)
2. Encrypt requested data, E(Q)
3. Send requested data, E(Q)
4. Matching data
5. Closest index
6. Exact computation
7. Identification result

FIGURE 1. System model.

B. THREAT MODEL

We consider the server to be "honest-but-curious" that is, the server must follow protocol but can analyze the protocol transcript to infer the private information relevant to the owner or clients. In the paper, we check whether the scheme is secure against following attackers or not, exclude restrictions on system policy(e.g., access control by the limitation of the number of identification request). This model is widely used to search encrypted data in cloud computing. As defined in [20], we classify the attackers $\mathcal{A}$ into the following four levels:

1) LEVEL-1 ATTACKER

The attacker $\mathcal{A}$ can access only the encrypted database $E(\mathcal{DB})$ and encrypted query $E(\mathcal{Q})$. This attack model corresponds to the ciphertext-only attack model in cryptography.

2) LEVEL-2 ATTACKER

The attacker $\mathcal{A}$ can access the encrypted database $E(\mathcal{DB})$ and encrypted query $E(\mathcal{Q})$, as well as some plaintexts $\mathcal{P} \in \mathcal{DB}$ in the database; however, the attacker $\mathcal{A}$ does not know the corresponding encrypted database. This attack model corresponds to the known-sample attack model in cryptography.

3) LEVEL-3 ATTACKER

In this attack model, the attacker $\mathcal{A}$ can access the encrypted database $E(\mathcal{DB})$, encrypted query $E(\mathcal{Q})$, and plaintext with the corresponding encrypted values $\{\mathcal{P}, E(\mathcal{P})\}$. This attack model corresponds to the known-plaintext attack (KPA) model in cryptography

4) LEVEL-4 ATTACKER

In addition to all the abilities in Level-3, the attacker $\mathcal{A}$ can query the forged templates and collude with the cloud server. In other words, the attacker $\mathcal{A}$ can obtain any plaintext with the corresponding encrypted values $\{\mathcal{P}, E(\mathcal{P})\}$ and any query with the corresponding encrypted query $\{\mathcal{Q}, E(\mathcal{Q})\}$. This attack model corresponds to the chosen-plaintext attack (CPA) model in cryptography.

C. EUCLIDEAN DISTANCE

Let $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ and $\mathbf{b} = (b_1, b_2, \ldots, b_n)$ be n-dimensional vectors in Euclidean n-space. A Euclidean distance between vectors $\mathbf{a}$ and $\mathbf{b}$ is the length of the line segment connecting the two vectors. We use it to determine whether the fingerprints are similar. If the calculated Euclidean distance is less than the threshold, we consider that the two fingerprints belong to the same person. Because of the fuzziness property of the biometric traits, the calculated Euclidean distance is not required to be precisely the same. The Euclidean distance between vectors $\mathbf{a}$ and $\mathbf{b}$ is defined as follows:

$$\begin{aligned} dist_{ab} &= \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2} \\ &= \sqrt{\Sigma_{i=1}^{n} (a_i - b_i)^2} \end{aligned}$$

TABLE 1. Definitions and notations.

Symbol	Definition
$\mathbf{b_i}$	the i-th reference template vector
$\mathbf{b_i^{'}}$	the extended vector of the vector $\mathbf{b_i}$
b_{ik}	a k-th element of the vector $\mathbf{b_i}$
$\mathbf{b_c}$	the candidate template vector
$\mathbf{b_c^{'}}$	the extended vector of the vector $\mathbf{b_c}$
n	the dimension of the fingerprint vector
$\mathbf{c_i}$	the ciphertext of the vector $\mathbf{b_i}$
$\mathbf{c_c}$	the ciphertext of the vector $\mathbf{b_c}$
M	an orthogonal matrix as a secret key
$RAND(a,b)$	a random integers on $[a, b]$
$AVG(S)$	an average of the set S
$MAX(a,b)$	a maximum value within a given range $[a, b]$
$MIN(a,b)$	a minimum value within a given range $[a, b]$

III. REVIEW OF LIU *et al.*'s SCHEME

In this section, we review Liu *et al.*'s scheme [25] in detail. Their scheme uses a similar construction technique to the technique used in Zhu *et al.*'s scheme [23], but there are two main differences. First, they use additional random numbers when the users are enrolled. This ensures that the scheme is secure against the KPA and CPA attacks. Second, they apply the concept of orthogonal matrices to key generation. Because of the properties of orthogonal matrices, the scheme can be implemented more efficiently. The details are as follows:

A. PREPARATION STAGE

Let $\mathbf{b_i}$ be the i-th reference template vector extracted from users by the algorithm in FingerCode [7]. When the users are enrolled, the vector $\mathbf{b_i}$ is extended to a $(n+5)-$dimensional vector as $\mathbf{b_i'} = (\alpha_i b_{i1}, \alpha_i b_{i2}, \cdots, \alpha_i b_{in}, -0.5\alpha_i \sum_{j=1}^{n} b_{ij}^2, \alpha_i, 0.5\alpha_i\tau^2, r_i, 0)$. Here, α_i is a random positive real number, b_{ij} is an 8-bit integer, r_i is a random real number, and τ is the presupposed threshold. Then, the data owner generates a random $(n+5)\times(n+5)$-dimensional orthogonal matrix M as a secret key and encrypts the extended template vector $\mathbf{b_i'}$ by calculating $\mathbf{c_i} = \mathbf{b_i'}M$. Finally, the data owner sends $\mathbf{c_i}$ to the cloud server.

B. REQUEST STAGE

At this stage, the data owner extends the candidate template vector $\mathbf{b_c}$ as $\mathbf{b_c} = (\beta_c b_{c1}, \beta_c b_{c2}, \cdots, \beta_c b_{cn}, \beta_c, -0.5\beta_c \sum_{j=1}^{n} b_{cj}^2, \beta_c, 0, r_c)$, where β_c is a random positive real number, and r_c is a random real number. Then, the data owner encrypts the extended candidate template vector $\mathbf{b_c'}$ with the secret key M and sends the encrypted vector $\mathbf{c_c} = \mathbf{b_c'}M$ to the cloud server.

C. IDENTIFICATION STAGE

When the cloud server receives the identification request, it computes $rd_i = \mathbf{c_i}\cdot\mathbf{c_c}$ for all $\mathbf{c_i}$ stored in the server. If $rd_i > 0$, the user is identified and can access the system as an honest user. Otherwise, the user cannot access the system.

D. CORRECTNESS

Let M be an orthogonal matrix, then given two vectors $\mathbf{a}$ and $\mathbf{b}$, the orthogonal matrix M satisfies the following property:

$$\mathbf{a}\cdot\mathbf{b} = (\mathbf{a}M)\cdot(\mathbf{b}M),$$

where $\cdot$ is the inner product. Based on this orthogonal matrix property, value rd_i can be computed as follows:

$$\begin{aligned} rd_i &= \mathbf{c_i}\cdot\mathbf{c_c} \\ &= \mathbf{b_i'}M\cdot\mathbf{b_c'}M \\ &= \mathbf{b_i'}\cdot\mathbf{b_c'} \\ &= \alpha_i\beta_c(\sum_{j=1}^{n} b_{ij}b_{cj} - \frac{1}{2}\sum_{j=1}^{n} b_{ij}^2 - \frac{1}{2}\sum_{j=1}^{n} b_{cj}^2 + \frac{1}{2}\tau^2) \\ &= \frac{1}{2}\alpha_i\beta_c(\tau^2 - \sum_{j=1}^{n}(b_{ij}-b_{cj})^2) \end{aligned}$$

If equation $rd_i > 0$ holds, then equation $\tau^2 > \sum_{j=1}^{n}(b_{ij}-b_{cj})^2$ also holds. Because equation $\sum_{j=1}^{n}(b_{ij}-b_{cj})^2$ is the Euclidean distance between $\mathbf{b_i}$ and $\mathbf{b_c}$, and τ is the presupposed threshold, the above equation holds that the candidate template $\mathbf{b_c}$ is sufficiently close to $\mathbf{b_i}$ to satisfy the identification condition.

IV. STATISTICAL ATTACK ON LIU *et al.*'s SCHEME

A. BASIC IDEA

This attack aims to generate a fake fingerprint vector that can pass the identification process as if it were an authorized user. It can be viewed as a type of impersonation attack. First,

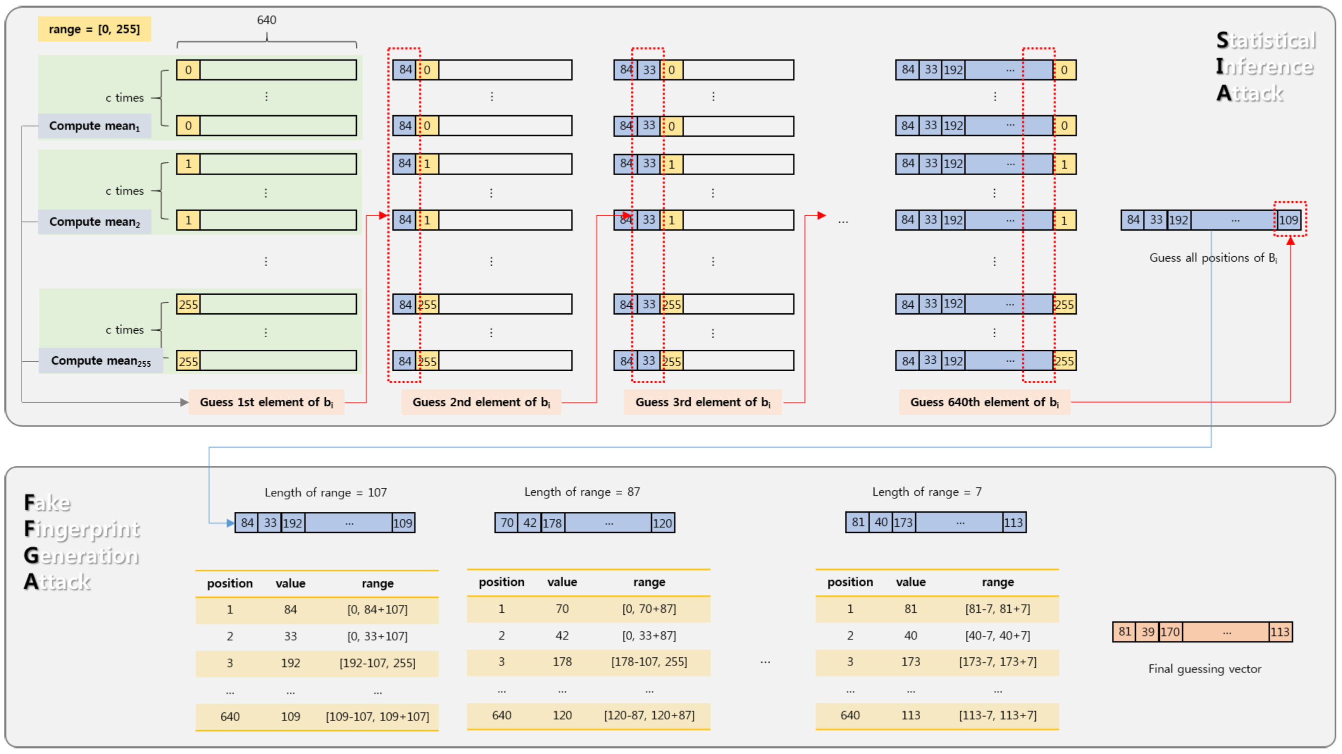
TABLE 2. This algorithm updates $\mathbf{b_c}$ to approximate $\mathbf{b_{i^*}}$ in a specific range of each element using a statistical inference technique.

```
Algorithm 1. SIA (Statistical Inference Attack)
Input: target index $i^*$, a vector $\mathbf{b_c}$, and $range$
 1: for $k$ from 1 to 640 do
 2:     $\delta_0 = MAX(0, b_{ck} - range)$
 3:     $\delta_1 = MIN(b_{ck} + (range+1), 255)$
 4:     for $j$ from $\delta_0$ to $\delta_1$ do
 5:         $b_{ck} = j$
 6:         for $t$ from 1 to $c$ do
 7:             1) $\mathcal{A}$ sends $\mathbf{b_c}$ to the $\mathcal{DO}$ as an identification request.
 8:             2) $\mathcal{CS}$ computes $rd_t$ between $\mathbf{b_{i^*}}$ and $\mathbf{b_c}$.
 9:             3) $\mathcal{A}$ receives $rd_t$ from the $\mathcal{CS}$.
10:         end for
11:         $mean_j = AVG(\{rd_t\}_{t=1}^{c})$
12:     end for
13:     Update $b_{ck}$ to $j^*$ (where $mean_{j^*}$ is maximum)
14: end for
```

TABLE 3. This algorithm generates a forged template $\mathbf{b_c}$ of $\mathbf{b_{i^*}}$ by repeatedly calling SIA. We assume that the template is a 640-dimensional vector, where each element is represented by 1 byte.

```
Algorithm 2. FFGA (Fake Fingerprint Generation Attack)
Input: target index $i^*$ and range parameter $w$
Output: forged template $\mathbf{b_c}$ of $\mathbf{b_{i^*}}$
 1: $range = 127$
 2: $\mathbf{b_c} = (127, 127, \cdots, 127)$
 3:     While ($range > 2$)
 4:         Run SIA($i^*$, $\mathbf{b_c}$, $range$)
 5:         1) $\mathcal{A}$ sends $\mathbf{b_c}$ to the $\mathcal{DO}$ as an identification request.
 6:         2) $\mathcal{CS}$ computes $rd$ between $\mathbf{b_{i^*}}$ and $\mathbf{b_c}$.
 7:         3) $\mathcal{A}$ receives $rd$ from the $\mathcal{CS}$.
 8:         if ($rd > 0$ )
 9:             return $\mathbf{b_c}$ with "Success"
10:         else
11:             $range = range - w$
12:         end if
13:     end while
14: return $\mathbf{b_c}$ with "Fail"
```

an adversarial user (adversary) $\mathcal{A}$ selects the target (fingerprint) vector $\mathbf{b_{i^*}}$ to be forged. If the cloud server colludes with the adversary, they can obtain the relative distance rd_i between the target vector $\mathbf{b_{i^*}}$ and vector $\mathbf{b_c}$ forged by adversary $\mathcal{A}$. Adversary $\mathcal{A}$ changes b_{c1} to 0, and then sends an identification request for vector $\mathbf{b_c}$ to the data owner. Adversary $\mathcal{A}$ repeats the request by increasing b_{c1} individually. For the target vector $\mathbf{b_{i^*}}$ and forged vector $\mathbf{b_c}$, the relative distance rd_i is only influenced by the value of the first element of the vector and random value β. Therefore, if adversary $\mathcal{A}$ removes the randomness affected by the random value β of the relative distance rd_i, adversary $\mathcal{A}$ can guess b_{i1} by using the relative distance rd_i. To remove the randomness of the relative distance rd_i, adversary $\mathcal{A}$ can use the statistical properties of β. Because the random value β is chosen as the uniform distribution, the mean value for the set of repeatedly collected values β converges to the mean value of the distribution with a high probability. By this property, adversary $\mathcal{A}$ can guess b_{i^*1} when the relative distance rd_i is maximal because the maximal relative distance rd_i indicates that the target vector $\mathbf{b_{i^*}}$ and forged vector $\mathbf{b_c}$ are sufficiently close to



position	value	range
1	84	[0, 84+107]
2	33	[0, 33+107]
3	192	[192-107, 255]
...	...	...
640	109	[109-107, 109+107]

position	value	range
1	70	[0, 70+87]
2	42	[0, 33+87]
3	178	[178-87, 255]
...	...	...
640	120	[120-87, 120+87]

position	value	range
1	81	[81-7, 81+7]
2	40	[40-7, 40+7]
3	173	[173-7, 173+7]
...	...	...
640	113	[113-7, 113+7]

FIGURE 2. Example of SIA and FFGA.

pass the identification process. By iterating this process and performing an additional process, adversary $\mathcal{A}$ can estimate the values of all elements of vector $\mathbf{b_{i^*}}$. Concrete parameters such as the iteration count, range of the random values, and distribution are discussed in Section V, and the detailed process is introduced in the next section.

B. OUR ATTACK ALGORITHMS: FFGA WITH SIA

STEP 1. Given target vector $\mathbf{b_{i^*}}$, adversary $\mathcal{A}$ sets the candidate vector $\mathbf{b_c}$ to $(0, 127, \cdots, 127)$ and sends an identification request for vector $\mathbf{b_c}$ to the data owner. Then, adversary $\mathcal{A}$ and the cloud server can obtain the relative distance rd_i between $\mathbf{b_{i^*}}$ and $\mathbf{b_c}$. Adversary $\mathcal{A}$ repeats this process for c times and calculates the mean for a set of rd_i, denoted as $mean_i$.

STEP 2. Adversary $\mathcal{A}$ repeats the process of **STEP 1**, increasing the first element of the candidate vector $\mathbf{b_c}$ by one. By repeating the process, the adversary can obtain 256 $mean_i$ and guess b_{i^*1} as I_1, where $mean_{I_1}$ is the maximum. Adversary $\mathcal{A}$ sets $\mathbf{b_c}$ to $(I_1, 127, \cdots, 127)$, where I_1 is the first element of $\mathbf{b_{i^*}}$.

STEP 3. Adversary $\mathcal{A}$ sets $\mathbf{b_c}$ to $(I_1, 0, 127, \cdots, 127)$, and then repeats the process of **STEP 1** and **STEP 2**. Through the iterative process, adversary $\mathcal{A}$ can guess the full-size fingerprint vector $\mathbf{b_c} = (I_1, I_2, \cdots, I_{640})$, where $\mathbf{b_c}$ is internally updated in the SIA algorithms.

STEP 4. Adversary $\mathcal{A}$ repeats the above process by reducing the range of the inference domain by the range parameter w. In **STEP 2**, adversary $\mathcal{A}$ sets the range of the i-th element from 0 to 255. In this step, adversary $\mathcal{A}$ sets the range of the i-th element from $(I_i - (range - w))$ to $(I_i + (range - w))$, where $range$ is a guess range in our algorithm and w is a range parameter. Adversary $\mathcal{A}$ repeatedly performs the above process with reduced ranges until the range of I_i is less than 2 and then obtains the final guessed vector $\mathbf{b_c} = (I_1, I_2, \cdots I_{640})$. If the relative distance rd_i between the final guessed vector $\mathbf{b_c}$ and target vector $\mathbf{b_{i^*}}$ is larger than 0, the attack is successful.

C. SECURITY ANALYSIS FOR LIU et al.'s SCHEME

In this section, we show that Liu *et al.*'s scheme is not secure against a level-4 attacker using the SIA and FFGA. Using the proposed attack algorithms, the adversary $\mathcal{A}$ can obtain a fake fingerprint that can pass the identification process. Whether the attack is successful depends on the parameters used in the scheme. Considering the conditions for a successful attack, rd_i is greater than 0. That is, the equation $\frac{1}{2}\alpha_i\beta_c[\tau^2 - \sum_{j=1}^{n}(b_{ij} - b_{cj})^2] > 0$ holds. In the equation, because α_i and β_c are random positive real numbers, the values do not affect the sign of the equation. Therefore, we must only find vector $\mathbf{b_c}$ that satisfies $\tau^2 > \sum_{j=1}^{n}(b_{ij} - b_{cj})^2$ for the given vector $\mathbf{b_i}$ and presupposed threshold τ. If $\tau = 25$, for example, we must find vector $\mathbf{b_c}$ that satisfies $625 > \sum_{j=1}^{n}(b_{ij} - b_{cj})^2$. This means that we need to find vector $\mathbf{b_c}$ that the Euclidean distance between $\mathbf{b_i}$ and $\mathbf{b_c}$ is less than 625. If $\tau = 1$, we must find vector $\mathbf{b_c}$ that satisfies $1 > \sum_{j=1}^{n}(b_{ij} - b_{cj})^2$. This means that we must find an identical vector with $\mathbf{b_{i^*}}$. Due to the nature of biometric data, it is difficult to extract the same value whenever the biometric data are extracted. While it is not common, it may be required depending on the environment. We also propose the parameter that the attack is successful when τ is 1.

Given the target fingerprint vector $\mathbf{b_{i^*}}$, it is possible to find vector $\mathbf{b_c}$ close to $\mathbf{b_{i^*}}$ using algorithms SIA and FFGA. The most important parameters affecting attack accuracy are an iteration count c and presupposed threshold τ. The iteration count c affects the attack complexity and can be selected by the attacker. The presupposed threshold τ is deterministically selected by the data owner or the system implements the biometric identification scheme; further, it determines the acceptance rate for the biometric identification scheme. Therefore, we apply various parameters to our attack algorithm to select the most ideal parameter to succeed in the attack. Because of the experiment, it was confirmed that a vector with a Euclidean distance of less than 625 can be generated when τ is 25 and c is set to $30,000$ with $w = 20$. The experimental results for various parameters are explained in the next section.

V. PERFORMANCE ANALYSIS

A. COMPLEXITY ANALYSIS

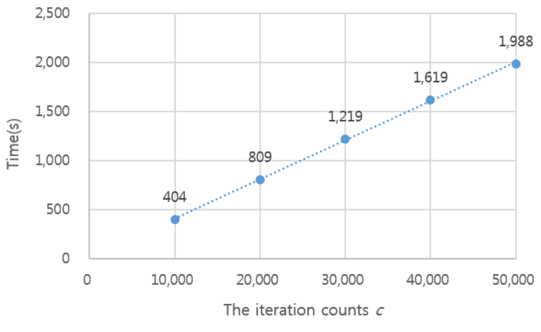
We analyzed the time complexity of our attack algorithm in this section. When we run the SIA algorithm to generate a fake fingerprint vector, the Euclidean distance is calculated by substituting values from 0 to 255 in 640 locations; thus, 256 operations are required. Considering the iteration count c used to eliminate the randomness of β_c, $256 \times 640 \times c$ operations are required, where c is set to a value between $10,000$ and $50,000$. In FFGA, SIA is called 7 times (with $w = 20$) repeatedly to apply the reduced domain to SIA. Thus, the number of total operations is $1.334 \times 2^{33} < 256 \times 640 \times c \times 7 < 1.669 \times 2^{35}$, where c is set to a value ranging from $10,000$ to $50,000$. Table 4 shows that our proposed attack is much faster than some known attacks in the worst-case. It is noteworthy that Liu *et al.*'s scheme is secure against various attacks, but our proposed attack breaks it within a reasonable time.

TABLE 4. Comparison on time complexity of various attacks.

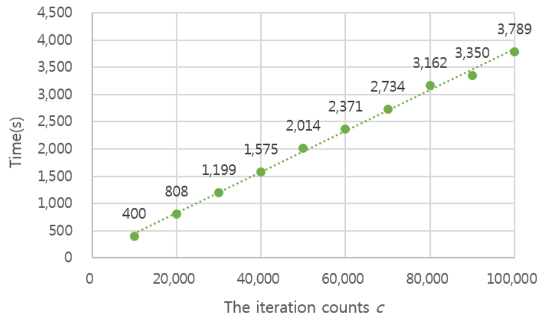
Type of attacks	Time complexity (worst-case)
Brute force attack (key) in [25]	$\approx 2^{207690\lambda}$
The CPA attack in [25]	$\approx 2^{644\lambda}$
Attack using relative distance in [25]	$\approx 2^{641\lambda}$
Brute force attack (fingerprint)	$\approx 2^{5120}$
Our proposed attack (FFGA)	$\approx 2^{36}$

λ is the security parameter which means the length of the secret key's element, where $\lambda > 1$.

Figures 3(a) and 3(b) show that the execution time changes as the iteration count c increases when τ is 25 and τ is 1, respectively. In Figure 3(a), when τ is 25, the attack can be sufficiently successful when c is $30,000$ (we explain this result in the next section); therefore, it takes only 1,219 s for an attacker to generate a fake fingerprint. Note that we only consider the execution time of our attack algorithms, and



(a) Time costs in our attack algorithms (τ=25)



(b) Time costs in our attack algorithms (τ=1)

FIGURE 3. Time costs in our attack algorithms.

not the communication time in the network. As mentioned before, the case where τ is 1 is not common. Nevertheless, we conducted an experiment and measured the time to prove that we can also find the fingerprint vector identical to the target vector through our attack algorithms. When τ is 25, it can be generated in $30,000$ iterations, but when τ is 1, it was experimentally found that $100,000$ iterations were required to generate a fake fingerprint. As shown in Figure 3(b), it takes 3,789 s for an attacker to generate a fake fingerprint vector identical to the target vector.

From Figures 3(a) and 3(b), it can be seen that τ does not significantly affect the execution time, and the execution time increases linearly with the iteration count c. As a result, the attack's success time can be predicted even if the vector size increases or the value of τ changes later. From the results, we can see that our attack has performed well within a realistic time frame.

B. EXPERIMENT ANALYSIS

To determine the exact parameters for a successful attack, we computed the Euclidean distance for various parameters. We implemented the attack algorithms using C language in Windows with a dual-core 3.60 GHz Intel(R) Core(TM) i7-4770 CPU and 8 GB memory. Similar to the previous works [23]–[25], we used the 640-dimensional random vectors with 1-byte elements as the fingerprint vectors. The scheme we are trying to attack is about how to compare the two encrypted fingerprint vectors while preserving privacy. An experiment can be tested with random data, not real-value, since the length of the biometric vector and the number of

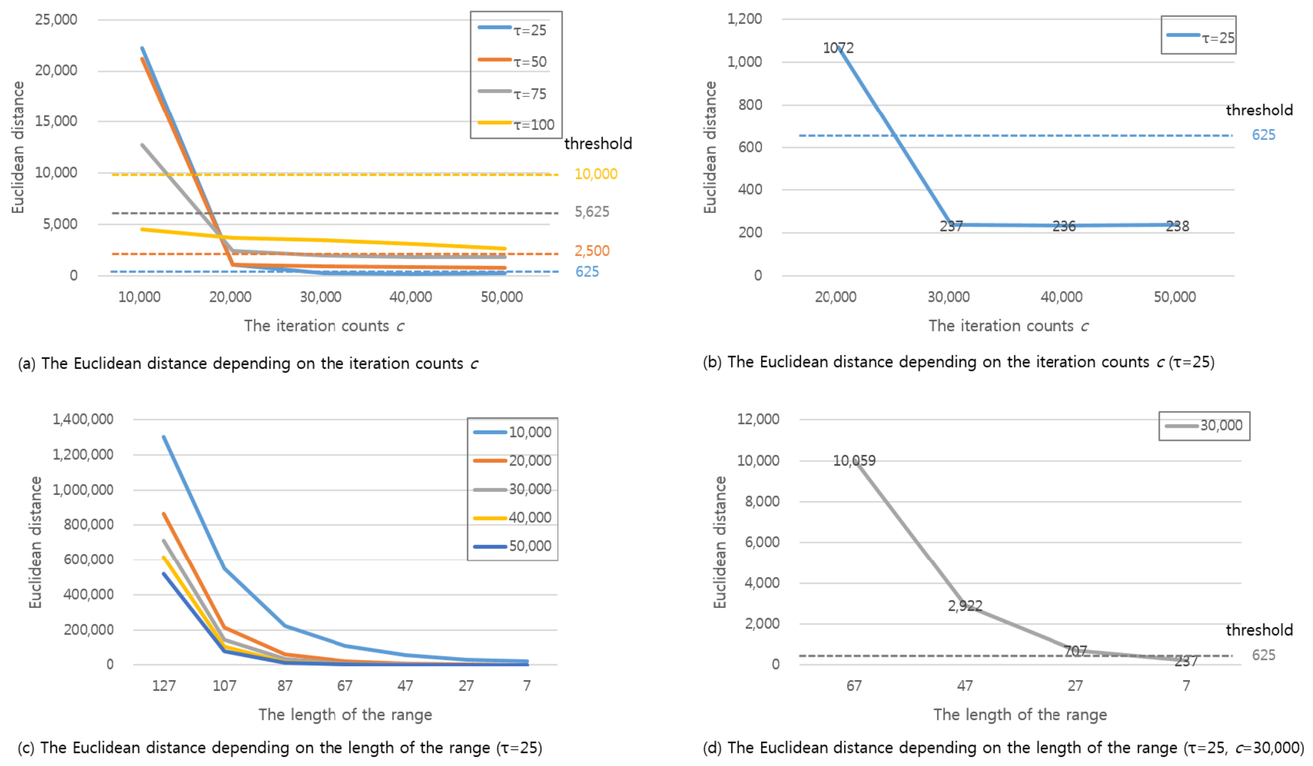


FIGURE 4. **Comparison of the Euclidean distance depending on the various parameters.**

mathematical operations implemented in the scheme have greater effect on the calculation time than the actual value of the vector. In fact, most of the works [23]–[25] assume that the fingerprint vector has already been extracted and use a random-value vector as the fingerprint vector in their experiment. Therefore, there is no big difference between using real-value vector and using random-value vector. As mentioned earlier, we consider only two main parameters, τ and c, because the other parameters are determined by the target vector $\mathbf{b_i}$ and do not affect the identification result. To obtain accurate experimental results, other random values r and α are fixed to 123.13 and 37.25, which are used to generate the extended vector $\mathbf{b}'_{\mathbf{i}}$, respectively.

In the Figures 4(a) and 4(b), we checked the changes in the Euclidean distance according to the iteration count c and the presupposed threshold τ. Figure 4(a) shows that the Euclidean distance decreases as c increases. To pass the identification process, the Euclidean distance should be less than threshold τ^2. For example, when τ is 75, the Euclidean distance between the target vector $\mathbf{b_i}$ and fake vector $\mathbf{b_c}$ should be less than $5,625$. Let us consider two cases: 1) when τ is 75 and c is $10,000$; and 2) when τ is 75 and c is $20,000$. In cases 1) and 2), the Euclidean distances are $12,728$ and $2,426$, respectively. In case 1) because the Euclidean distance is greater than $5,625$, the fake vector $\mathbf{b_c}$ is denied in the identification process. However, because the distance in case 2) is less than $5,625$, the fake vector $\mathbf{b_c}$ can pass the identification process. When τ is 100, there is no need to set c to more than $10,000$ because the values in the graph are all less than $10,000$. As shown in Figure 4(a), if c is greater than $30,000$, all distances are within the threshold value regardless of τ. Notably, the experiments with τ values greater than 100 are meaningless, as c decreases as τ increases. Figure 4(b) shows that the Euclidean distance decreases as c increases when τ is 25. When c is $20,000$ and $30,000$, the Euclidean distances are $1,072$ and 237, respectively. In this case, we know that more than $30,000$ iterations are not required to pass the identification process because the Euclidean distance when c is $30,000$ is much less than the threshold distance 625.

In the Figures 4(c) and 4(d), we checked how the Euclidean distance changes according to the length of the range and c when τ is 25. Figure 4(c) shows that the Euclidean distance decreases as the length of the range decreases. Because FFGA repeats SIA in a reduced domain, we can increase the accuracy and decrease the operation times. We set the interval of the range to 20, which is obtained as an experimental result because we can obtain more accurate results when the interval of the range is 20 in our algorithms. Figure 4(d) shows that the Euclidean distance decreases as the length of the range decreases when τ is 25 and c is $30,000$. In the figure, the distance when the length of the range is 27 exceeds the threshold value, but the distance when the length of the range is 7 is within the threshold value. Therefore, when c is $30,000$ and τ is 25, we can generate a fake fingerprint vector that can pass the identification process.

The results of the experiments verify that Liu *et al.*'s scheme is not secure against a level-4 attack by our proposed algorithms.

VI. CONCLUSION

Privacy-preserving biometric identification schemes enable valid identification without any risk to private biometric information. Although these schemes have been researched extensively, many concerns regarding security and efficiency remain. In this paper, we introduce new algorithms SIA and FFGA that generate fake fingerprint capable of passing the identification processes, and show that Liu *et al.*' scheme is not secure against a level-4 attacker who uses our proposed algorithms. Moreover, we provide an analysis of the attack complexity and experimental results to which concrete parameters were applied. In the future work, we plan to expand our attack and apply it to various biometric identification schemes, and we plan to design a privacy-preserving biometric identification that is secure against our attack.

REFERENCES

[1] R. Bolle and S. Pankanti, *Personal Identification in Networked Society: Personal Identification in Networked Society*, A. K. Jain, Ed. Norwell, MA, USA: Kluwer, 1998.

[2] S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric recognition: Security and privacy concerns," *IEEE Secur. Privacy*, vol. 1, no. 2, pp. 33–42, Mar. 2003.

[3] N.K. Ratha, "Privacy protection in high security biometrics applications," in *Proc. ICEB* (Lecture Notes in Computer Science), vol. 6005, A. Kumar and D. Zhang, Eds. New York, NY, USA: Springer, 2010, pp. 62–69.

[4] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 4, pp. 744–757, Dec. 2007.

[5] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Proc. 9th Int. Symp. Privacy Enhancing Technol. Symp.*, 2009, pp. 235–253.

[6] A. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy-preserving face recognition," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2009, pp. 229–244.

[7] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "Filterbank-based fingerprint matching," *IEEE Trans. Image Process.*, vol. 9, no. 5, pp. 846–859, May 2000.

[8] S. Prabhakar and A. K. Jain, "Decision-level fusion in fingerprint verification," *Pattern Recognit.*, vol. 35, no. 4, pp. 861–874, 2001.

[9] A. M. Bazen and S. H. Gerez, "Systematic methods for the computation of the directional fields and singular points of fingerprints," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 905–919, Jul. 2002.

[10] H. Xu, R. N. J. Veldhuis, A. M. Bazen, T. A. M. Kevenaar, T. A. H. M. Akkermans, and B. Gokberk, "Fingerprint verification using spectral minutiae representations," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 3, pp. 397–409, Sep. 2009.

[11] A. Ross, A. Jain, and J. Reisman, "A hybrid fingerprint matcher," *Pattern Recognit.*, vol. 36, no. 7, pp. 1661–1673, Jul. 2003.

[12] M. Barni, F. Scotti, A. Piva, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, and V. Piuri, "Privacy-preserving fingercode authentication," in *Proc. 12th ACM Workshop Multimedia Secur. (MM&Sec)*, New York, NY, USA, 2010, pp. 231–240.

[13] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, "SCiFI–A system for secure face identification," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, May 2010, pp. 239–254.

[14] Y. Huang, L. Malka, D. Evans, and J. Katz, "Efficient privacy-preserving biometric identification," in *Proc. NDSS*, 2011, pp. 1–40.

[15] J. Yuan and S. Yu, "Efficient privacy-preserving biometric identification in cloud computing," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2652–2660.

[16] N. Cao, Z. Yang, C. Wang, K. Ren, and W. Lou, "Privacy-preserving multi-keyword search ranked search over encrypted cloud data," in *Proc. INFOCOM*, May 2011, pp. 829–837.

[17] R. Li, Z. Xu, W. Kang, K. C. Yow, and C.-Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," *Future Gener. Comput. Syst.*, vol. 30, pp. 179–190, Jan. 2014.

[18] A. K. Jain, S. Prabhakar, and L. Hong, "A multichannel approach to fingerprint classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 4, pp. 348–359, Apr. 1999.

[19] Y. Zhu, T. Takagi, and R. Hu, "Security analysis of collusion-resistant nearest neighbor query scheme on encrypted cloud data," *IEICE Trans. Inf. Syst.*, vol. 97, no. 2, pp. 326–330, 2014.

[20] W. K. Wong, D. W.-L. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proc. 35th SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2009, pp. 139–252.

[21] B. Yao, F. Li, and X. Xiao, "Secure nearest neighbor revisited," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Apr. 2013, pp. 733–744.

[22] Y. Zhu, R. Xu, and T. Takagi, "Secure K-NN computation on encrypted cloud data without sharing key with query users," in *Proc. Int. Workshop Secur. Cloud Comput. (Cloud Comput.)*, 2013, pp. 55–60.

[23] L. Zhu, C. Zhang, C. Xu, X. Liu, and C. Huang, "An efficient and privacy-preserving biometric identification scheme in cloud computing," *IEEE Access*, vol. 6, pp. 19025–19033, 2018.

[24] S. Hu, M. Li, Q. Wang, S. S. M. Chow, and M. Du, "Outsourced biometric identification with privacy," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 10, pp. 2448–2463, Oct. 2018.

[25] C. Liu, X. Hu, Q. Zhang, J. Wei, and W. Liu, "An efficient biometric identification in cloud computing with enhanced privacy security," *IEEE Access*, vol. 7, pp. 105363–105375, 2019.

[26] J.-H. Im, J. Choi, D. Nyang, and M.-K. Lee, "Privacy-preserving palm print authentication using homomorphic encryption," in *Proc. IEEE 14th Int. Conf. Dependable, Autonomic Secure Comput., 14th Int. Conf. Pervas. Intell. Comput., 2nd Int. Conf. Big Data Intell. Comput. Cyber Sci. Technol. Congr. (DASC/PiCom/DataCom/CyberSciTech)*, Aug. 2016, pp. 878–881.

[27] H. Zhu, Q. Wei, X. Yang, R. Lu, and H. Li, "Efficient and privacy-preserving online fingerprint authentication scheme over outsourced data," *IEEE Trans. Cloud Comput.*, early access, Aug. 21, 2018, doi: 10.1109/TCC.2018.2866405.

[28] P. Pailier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1999, pp. 223–238.

[29] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2005, pp. 325–341.

[30] D. Catalano and D. Fiore, "Using linearly-homomorphic encryption to evaluate degree-2 functions on encrypted data," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1518–1529.

DONGMIN KIM received the M.S. and Ph.D. degrees from the Graduate School of Information Security, Korea University, Seoul, South Korea, in 2011 and 2017, respectively. Since 2016, he has been a Senior Researcher with The Affiliated Institute of ETRI, South Korea. His research interests include cryptography, database security, privacy enhancing technology, public auditing protocol, and cryptographic module validation program.

KEE SUNG KIM received the M.S. and Ph.D. degrees from the Graduate School of Information Security, Korea University, Seoul, South Korea, in 2011 and 2015, respectively. From 2014 to 2018, he was a Senior Researcher with The Affiliated Institute of ETRI, South Korea. He is currently an Assistant Professor with the School of Information Technology Engineering, Daegu Catholic University, South Korea. His research interests include cryptography, database security, privacy enhancing technology, and secure protocols.