The Korean Institute of Communications and Information Sciences
CrossMark
Available online at www.sciencedirect.com
ScienceDirect
ICT Express 2 (2016) 135–139
www.elsevier.com/locate/icte

Efficient and privacy-preserving biometric identification in cloud☆

Changhee Hahn, Junbeom Hur*

Department of Computer Science and Engineering, Korea University, Republic of Korea

Received 28 July 2016; accepted 14 August 2016
Available online 24 August 2016

Abstract

With the rapid growth in the development of smart devices equipped with biometric sensors, client identification system using biometric traits are widely adopted across various applications. Among many biometric traits, fingerprint-based identification systems have been extensively studied and deployed. However, to adopt biometric identification systems in practical applications, two main obstacles in terms of efficiency and client privacy must be resolved simultaneously. That is, identification should be performed at an acceptable time, and only a client should have access to his/her biometric traits, which are not revocable if leaked. Until now, multiple studies have demonstrated successful protection of client biometric data; however, such systems lack efficiency that leads to excessive time utilization for identification. The most recently researched scheme shows efficiency improvements but reveals client biometric traits to other entities such as biometric database server. This violates client privacy. In this paper, we propose an efficient and privacy-preserving fingerprint identification scheme by using cloud systems. The proposed scheme extensively exploits the computation power of a cloud so that most of the laborious computations are performed by the cloud service provider. According to our experimental results on an Amazon EC2 cloud, the proposed scheme is faster than the existing schemes and guarantees client privacy by exploiting symmetric homomorphic encryption. Our security analysis shows that during identification, the client fingerprint data is not disclosed to the cloud service provider or fingerprint database server.

Keywords: Privacy; Biometrics; Identification; Cloud

1. Introduction

Biometric identification is one of the most prominent methods for identifying an individual. All biometric traits, such as fingerprint, iris, and retina, share the important factors of universality (people have their own fingerprint), uniqueness (the probability that two persons have the same fingerprint is negligible), and permanence (biometric traits usually do not change over time) [1]. Such properties have certain pros and cons. Although they make the usage of biometric traits easy and client identification precise, they raise concerns for client privacy. For example, suppose Alice identifies herself using her fingerprint to access some web services, such as a health-care service and social network service (SNS), the service providers may track the transmission of her fingerprint and discern her private information including health condition and registered SNS identity. This severely violates client privacy. Furthermore, if Alice's fingerprint data is revealed to the public, anyone can masquerade as Alice by simply submitting her fingerprint data, thereby invalidating the entire identification system [2,3]. As biometric traits are unique and cannot be changed during the lifetime, once leaked, they cannot be revoked and re-generated.

Recently, several studies [4,5] have proposed privacy-preserving fingerprint identification systems, which use an asymmetric homomorphic encryption algorithm to encrypt the fingerprint data so that only key owners can access their fingerprints. Although the systems guarantee privacy-preserving identification, the computation cost of the encryption algorithm is considerable. Thus, they are not scalable considering the growing number of clients.

Yuan et al. [6] introduced an efficiency-improved fingerprint identification scheme that exploits matrix operations to encrypt fingerprint data, thus avoiding heavy computations compared with the schemes using the asymmetric encryption algorithm.

* Corresponding author.
E-mail addresses: hahn850514@korea.ac.kr (C. Hahn), jbhur@korea.ac.kr (J. Hur).

Peer review under responsibility of The Korean Institute of Communications Information Sciences.

☆ This paper is part of a special issue entitled ICT Convergence in the Internet of Things (IoT) guest edited by Yacine Ghamri-Doudane, Yeong Min Jang, Daeyoung Kim, Hossam Hassanein and JaeSeung Song.

http://dx.doi.org/10.1016/j.icte.2016.08.006
2405-9595/© 2016 The Korean Institute of Communications Information Sciences. Publishing Services by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

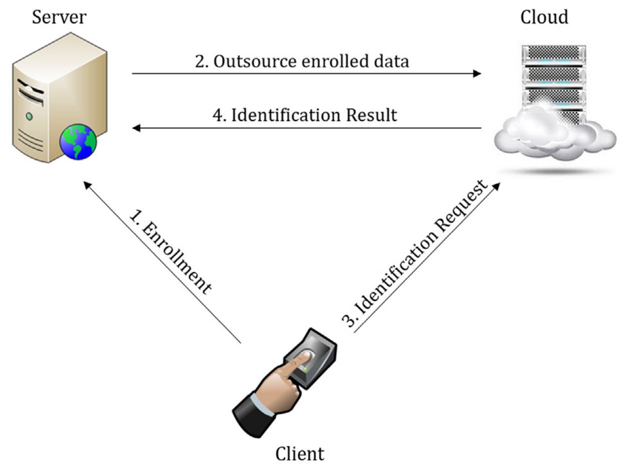

Fig. 1. The system model of the proposed scheme.

Further, most computations are shifted from a server onto a third-party entity (e.g., a cloud) to exploit their resources. Nevertheless, there is a trade-off between efficiency and privacy, that is, they must assume that the server has the authority to access the fingerprint database. We stress that such assumption must be alleviated considering irrevocability of biometric data. For example, a malicious employee with access to the fingerprint database may sell a copy of the data to someone, or the server could be compromised, thus rendering the data unrecoverable.

In this paper, we propose an efficient privacy-preserving fingerprint-based identification scheme. Compared with Yuan et al.'s scheme [6], our scheme exploits a symmetric homomorphic encryption algorithm in biometric identification, thus achieving both security and efficiency. We allow the server to outsource most computations to a cloud to save storage cost and improve efficiency so that fingerprints are secure.

2. System description

2.1. System model

The proposed scheme consists of three entities: a client, data server (a server for short), and cloud (see Fig. 1). The client encrypts and enrolls his/her fingerprint. For identification, the client encrypts and sends a newly scanned fingerprint to the cloud. Note that the previous key used to enroll is not re-used but a fresh key is generated at every identification to encrypt the fingerprint.

We adopt a filterbank-based fingerprint matching system [7], which is also used in other biometric identification schemes. This system [7] promises high accuracy by using FingerCode: a chain of N-independent feature codes that are typically 8-bit integers. This is used to measure the Euclidean distance between two fingerprints.

2.2. Threat model

We assume that attackers reside outside the system and attempt to eavesdrop on the data sent from a client. The goal of these attackers is to obtain a client's raw biometric data, in this case, the fingerprint. The attackers can then bypass the identification process and successfully access the data server. As mentioned earlier, biometric traits are incapable of being revoked when leaked. Therefore, it is important that the biometric data is secured from attackers.

We define the cloud as an honest-but-curious entity, implying that it behaves properly in most cases but attempts to harvest biometric information. In addition, we postulate that the cloud may collude with an outside adversary to recover a client's fingerprint data to gain illegal profits. We assume that the data server is also curious about fingerprint data. A data server providing service to a client does not necessarily imply that it is allowed to access the client's fingerprint data.

2.3. Design goal

Our goal is threefold. First, during enrollment and identification, fingerprint data should not be revealed to any entities including the server and the cloud. Next, the proposed scheme should be able to filter out malicious clients who submit random values similar to legitimate clients' FingerCodes. Lastly, the identification regarding computation and communication should be efficient.

3. The proposed scheme

3.1. Preliminaries

Let $Enc(\cdot)$ be a homomorphic encryption function. Then, for any given encryption key k, the encryption function satisfies $Enc_k(m_1 \lhd m_2) \longleftarrow Enc_k(m_1) \rhd Enc_k(m_2)$, for some operators $\lhd$ and $\rhd$ on input messages m_1 and m_2. The encryption scheme is said to be additively homomorphic if the following equation holds, given two encrypted messages, $Enc_{k_1}(m_1)$ and $Enc_{k_2}(m_2)$:

$$Enc_{(k_1+k_2)}(m_1 + m_2) = Enc_{k_1}(m_1) + Enc_{k_2}(m_2). \tag{1}$$

We use a secret key $k = [k_1, \ldots, k_N]$ to encrypt the FingerCode $m = [x_1, \ldots, x_N]$ as follows:

$$Enc_{k_i}(x_1) = x_1 + k_1 \bmod M, \tag{2}$$

$$\vdots$$

$$Enc_{k_N}(x_N) = x_N + k_N \bmod M, \tag{3}$$

where M is a randomly-chosen large integer satisfying $0 \le x_i < M$.

3.2. Initial client enrollment

The first client has FingerCode $m_1 = [x_{11}, \ldots, x_{1N}]$ and a random key $k_1 = [k_{11}, \ldots, k_{1N}]$. He encrypts m_1 such that $Enc_{k_1}(m_1) = [x_{11} + k_{11} \bmod M, \ldots, x_{1N} + k_{1N} \bmod M]$. The client then sends the encrypted file to the server, which re-encrypts the file by using the key pair (k_{s1}, k'_{s1}). Note that neither key is permanent, instead they are used only for the enrollment of the first client. The server computes $Enc_{k_{s1}+k'_{s1}}(0)$ and re-encrypts $Enc_{k_1}(m_1)$ to generate $Enc_{k_1+k_{s1}+k'_{s1}}(m_1)$,

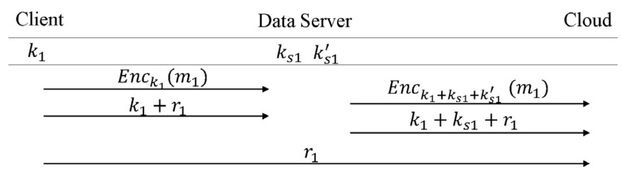

Fig. 2. Initial client enrollment process.

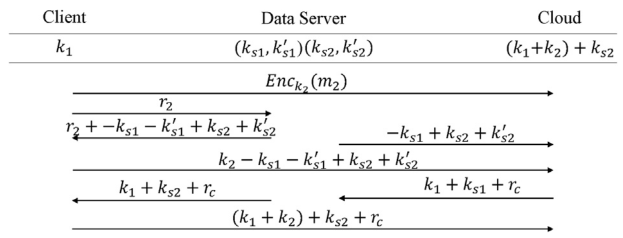
Client
Data Server
Cloud
k_1
$(k_{s1}, k'_{s1})(k_{s2}, k'_{s2})$
$(k_1+k_2) + k_{s2}$
$Enc_{k_2}(m_2)$
r_2
$r_2 + -k_{s1} - k'_{s1} + k_{s2} + k'_{s2}$
$-k_{s1} + k_{s2} + k'_{s2}$
$k_2 - k_{s1} - k'_{s1} + k_{s2} + k'_{s2}$
$k_1 + k_{s2} + r_c$
$k_1 + k_{s1} + r_c$
$(k_1 + k_2) + k_{s2} + r_c$

Fig. 3. An example of subsequent client enrollment process when $n = 2$.

which the server then sends to the cloud. Next, the client, server, and cloud establish a secure channel among them to run a simple key-exchange protocol. First, the client sends $k_1 + r_1$, where $r_1 = [r_{11}, \ldots, r_{1N}]$ is a random number vector, to the server. The server adds k_{s1} to $k_1 + r_1$ and sends the result to the cloud. Next, the client sends r_1 directly to the cloud. By subtracting r_1, the cloud acquires $k_1 + k_{s1}$ without knowing individual keys. After the first client registration is complete, the cloud contains $k_1 + k_{s1}$, $Enc_{k_1+k_{s1}+k'_{s1}}(m_1)$, $Enc_{k_{s1}+k'_{s1}}(0)$. The initial client enrollment process is depicted in Fig. 2.

3.3. Subsequent client enrollment

We first show the second client enrollment process and then generalize it for the nth client. The second client generates a randomly chosen key k_2 and encrypts m_2. The encrypted data $Enc_{k_2}(m_2)$ is sent directly to the cloud. Next, the client chooses and sends a random number vector r_2 to the server, which first generates a new key pair (k_{s2}, k'_{s2}), computes $r_2 - \left(k_{s1} + k'_{s1}\right) + (k_{s2} + k'_{s2})$ and $\left(k_{s2} + k'_{s2}\right) - k_{s1}$, and sends them to the second client and the cloud, respectively. The client then computes and sends $k_2 - \left(k_{s1} + k'_{s1}\right) + (k_{s2} + k'_{s2})$ to the cloud. By using both values sent by the client and server, the cloud updates the biometric database as follows:

$$Enc_{\sum_{i=1}^{2} k_i + k_{s2} + k'_{s2}}(m_1) = Enc_{k_1+k_{s1}+k'_{s1}}(m_1) + Enc_{k_2-(k_{s1}+k'_{s1})+(k_{s2}+k'_{s2})}(0), \quad (4)$$

$$Enc_{\sum_{i=1}^{2} k_i + k_{s2} + k'_{s2}}(m_2) = Enc_{k_2}(m_2) + Enc_{k_{s1}+k'_{s1}}(0) + Enc_{(k_{s2}+k'_{s2})-k_{s1}}(0). \quad (5)$$

Note that biometric data m_1 and m_2 were earlier encrypted using different keys. However, they are now encrypted with the same key. After updating the database, the cloud initiates a secure channel to update its key. First, the cloud generates a random number vector r_c and sends $k_1 + k_{s1} + r_c$ to the server, which deletes k_{s1} and adds k_{s2}. Next, $k_1 + k_{s2} + r_c$ is sent to the second client. The client adds his key k_2 and finally sends the result to the cloud. By subtracting r$_c$, the cloud updates its key. Simultaneously, the server revokes the key pair (k_{s2}, k'_{s2}). The cloud then uses $\sum_{i=1}^{2} k_i + k_{s2} + k'_{s2}$ as a key. By iterating this process for n number of clients, the cloud has encrypted FingerCodes as follows:

$$Enc_{\sum_{i=1}^{n} k_i + k_{sn} + k'_{sn}}(m_1),$$
$$\vdots$$
$$Enc_{\sum_{i=1}^{n} k_i + k_{sn} + k'_{sn}}(m_n).$$

The second client enrollment process is shown in Fig. 3.

3.4. Secure biometric identification

To make an identification request, the client generates a new random key k_c and encrypts $-m_c$. Then, $Enc_{k_c}(-m_c)$ is sent directly to the cloud, which then adds a new random number vector r to it. The server obtains the result and computes $Enc_{k_c-k'_{sn}}(r - m_c)$ by simply adding the negative value of its key k'_{sn}. The client computes $Enc_{-k'_{sn}}(r - m_c)$ by subtracting his key k_c, and sends the result to the cloud. The cloud first subtracts r and computes Euclidean distances between the biometric database and candidate data.

$$Enc_{\sum_{i=1}^{n} k_i + k_{sn} + k'_{sn}}(m_1) + Enc_{-k'_{sn}}(-m_c) = Enc_{\sum_{i=1}^{n} k_i + k_{sn}}(m_1 - m_c), \quad (6)$$
$$\vdots$$
$$Enc_{\sum_{i=1}^{n} k_i + k_{sn} + k'_{sn}}(m_n) + Enc_{-k'_{sn}}(-m_c) = Enc_{\sum_{i=1}^{n} k_i + k_{sn}}(m_n - m_c). \quad (7)$$

As shown in Eqs. (6) and (7), the result is now encrypted by the cloud's secret key. By decrypting the result, the cloud obtains $\left(m_j - m_c\right)$, where $1 \le j \le n$. To compare the Euclidean distances between two FingerCodes, the cloud computes the distance as follows:

$$dist\left(m_j, m_c\right) = \sqrt{\left(x_{j1} - x_{c1}\right)^2 + \cdots + (x_{kN} - x_{cN})^2}. \quad (8)$$

By computing all distances, the cloud locates the smallest distance $dist_{min}$, and sends it to the server to verify whether the client is legitimate by using some threshold, that is, if $dist_{min} \le threshold$, the client is identified successfully.

4. Analysis

4.1. Security analysis

In this subsection, we demonstrate that the proposed scheme is secure. First, we assume that the attacker can be a valid client and send an identification request to the server and cloud. In

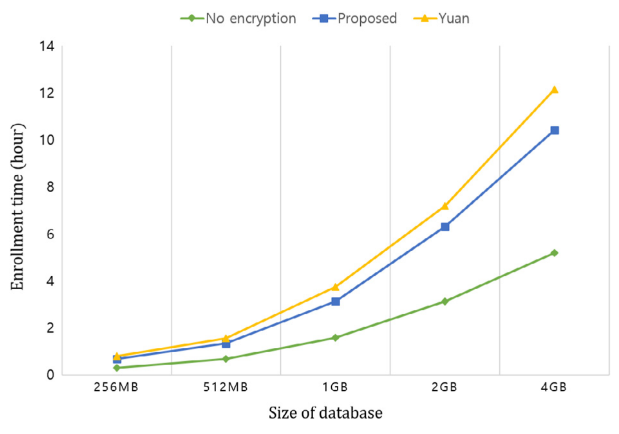

Fig. 4. Time cost for enrollment phase for different database sizes.

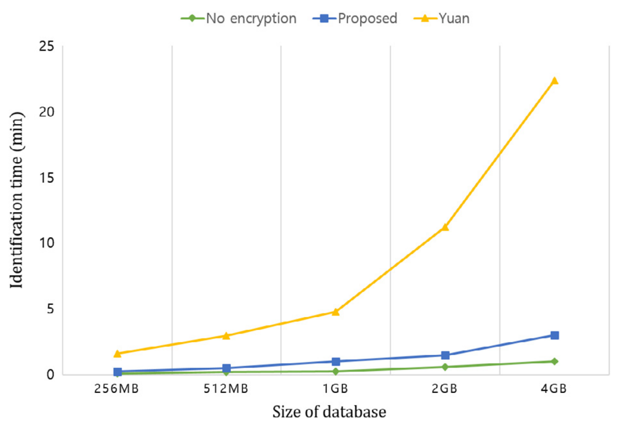

Fig. 5. Time cost for identification time for different database sizes.

addition, the cloud may collude with clients to harvest the raw FingerCode data in the database.

To harvest a client's FingerCode m_k, the cloud should determine the value r_j, where $1 \leq j \leq n$. Specifically, we have an encrypted form $c_k = [(x_{k1} + r_1 \mod M), \ldots, (x_{kN} + r_N \mod M)]$; then, the cloud has the following N equations:

$$x_{k1} + r_1 = c_{k1}, \tag{9}$$

$$\vdots$$

$$x_{kN} + r_N = c_{kN}. \tag{10}$$

The cloud cannot learn the FingerCode because there are $2N$ unknown variables, and it only knows N equations.

In case of collusion between the cloud and a set of malicious clients, the database encryption key is the sum of all clients' keys $\sum_{i=1}^{n} k_i$ and server's key pair (k_{sn}, k'_{sn}). Even if a number of clients collude with the cloud, the partial sum of the malicious clients' key cannot facilitate the cloud in guessing the value $\sum_{i=1}^{n} k_i + k_{sn} + k'_{sn}$. To harvest the client's private data, the key k'_{sn} is required, which cannot be obtained by the cloud. Therefore, the proposed scheme is secure against malicious entities.

4.2. *Performance analysis*

We implemented the proposed scheme to evaluate the actual performance. We programmed our proposed system by using the Java language on a Linux instance in the Amazon EC2 cloud [8] to analyze practicality of the proposed scheme in cloud computing. In order to construct the practical biometric identification scheme, we set up five databases with different sizes. The Amazon EC2 cloud consists of one instance node with 2.5 GHz Intel E5-2670v2 CPU and 1GiB memory. The cloud instance runs Linux with low I/O performance. We generated five databases with 640-elements contained in FingerCode data. To the best of our knowledge, the scheme by Yuan et al. [6] is the most recent and fastest among the existing schemes. Therefore, we implemented this biometric identification scheme to compare with our scheme.

As shown in Fig. 4, the enrollment time reaches 40 min for a 256 MB database, and increases linearly with the size of the database. With a 4 GB database, 10.4 h are needed for database enrollment. Meanwhile, the scheme by Yuan et al. [6] requires 48 min at 256 MB and 12.15 h at 4 GB. Both schemes show relatively similar performance results in the enrollment phase even though we exploited the additively homomorphic encryption scheme that uses only simple addition and modulation, while Yuan et al. used matrix operations which cause heavy computational overhead. This is because our scheme requires scanning of an entire database for each client enrollment, thus requiring more computation in the cloud. Nevertheless, the enrollment process is performed only once during the service.

When the client sends an identification request, our scheme shows a great advantage over that by Yuan et al. [6], as shown in Fig. 5. In a 256 MB database, our scheme needs 7.8 s, whereas the scheme by Yuan et al. requires 96 s. This difference increases with the size of the database. For example, for 256 MB and 4 GB databases, our scheme needs only 15 s and 2 min, respectively. In contrast, the scheme by Yuan et al. requires 96 s and 22 min for 256 MB and 4 GB databases. Such performance gap is caused because we only use $O(N)$ operations over an entire database with N-element FingerCode data, whereas the scheme by Yuan et al. needs $O(N^2)$ computations because of the utilization of the matrix operation. Thus, the proposed scheme outperforms that of Yuan et al. during client identification, demonstrating that the proposed scheme is approximately 11 times faster.

5. Conclusion

In this paper, we proposed an efficient and privacy-preserving biometric identification scheme in cloud computing. Unlike in previous studies, we allow no entities except the client to access the client's biometric data. The security analysis shows that the biometric data is not disclosed to the server and cloud. By leveraging the additively homomorphic encryption, we securely outsource the client biometric identification to the cloud. Moreover, the performance analysis and experimental results show that our scheme outperforms the previous schemes in terms of computation and communication.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2016R1A2A2A05005402).

References

[1] R. Bolle, S. Pankanti, Biometrics: Personal Identification in Networked Society, Kluwer Academic Publishers, 1998.

[2] S. Li, A.C. Kot, An improved scheme for full fingerprint reconstruction, IEEE Trans. Inf. Forensics Secur. (2012).

[3] K. Cao, J. Anil, Learning fingerprint reconstruction: from minutiae to image, IEEE Trans. Inf. Forensics Secur. (2015).

[4] M. Barni, T. Bianchi, D. Catalano, M.D. Raimondo, R.D. Labati, P. Faillia, Privacy-preserving fingercode authentication, in: MM&Sec', 2010.

[5] Y. Huang, L. Malka, D. Evans, J. Katz, Efficient privacy-preserving biometric identification, in: NDSS, 2011.

[6] J. Yuan, S. Yu, Efficient privacy-preserving biometric identification in cloud computing, in: Proc. Of IEEE INFOCOM, 2013.

[7] A.K. Jain, S. Prabhakar, L. Hong, S. pankanti, Filterbank-based fingerprint matching, IEEE Trans. Image Process. (2000).

[8] Amazon Inc. Amazon Elastic Compute Cloud (Amazon EC2), https://aws.amazon.com/ec2.