J Inf Process Syst, Vol.16, No.3, pp.599~611, June 2020
https://doi.org/10.3745/JIPS.03.0138

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Iris Ciphertext Authentication System Based on Fully Homomorphic Encryption

Xinxia Song*, Zhigang Chen**,***, and Dechao Sun**

Abstract

With the application and promotion of biometric technology, biometrics has become more and more important to identity authentication. In order to ensure the privacy of the user, the biometrics cannot be stored or manipulated in plaintext. Aiming at this problem, this paper analyzes and summarizes the scheme and performance of the existing biometric authentication system, and proposes an iris-based ciphertext authentication system based on fully homomorphic encryption using the FV scheme. The implementation of the system is partly powered by Microsoft's SEAL (Simple Encrypted Arithmetic Library). The entire system can complete iris authentication without decrypting the iris feature template, and the database stores the homomorphic ciphertext of the iris feature template. Thus, there is no need to worry about the leakage of the iris feature template. At the same time, the system does not require a trusted center for authentication, and the authentication is completed on the server side directly using the one-time MAC authentication method. Tests have shown that when the system adopts an iris algorithm with a low depth of calculation circuit such as the Hamming distance comparison algorithm, it has good performance, which basically meets the requirements of real application scenarios.

Keywords

Fully Homomorphic Encryption, Iris Authentication, One-Time MAC, SEAL

1. Introduction

In recent years, the biometric authentication has gradually changed the way people live, the first of which is mobile payment. Compared with traditional password authentication, biological features are not easily lost or forgotten, while ensuring the uniqueness of authentication users. On the other hand, due to the high security of biological characteristics, once biometric data is stolen, it will have more serious consequences than other authentication methods. Therefore, it is very important to develop a solution to the biometric data with stronger protection force for the authentication system.

At present, researchers have put forward many different biological feature template protection schemes, which fall into two broad categories:

(1) Feature transformation method. This method transforms biometric data into random data using client-specific keys or passwords. Typical examples of this method are the Biohashing [1] and

Manuscript received December 7, 2018; first revision March 8, 2019; accepted April 18, 2019.
Corresponding Author: Zhigang Chen (chenzhigang@zwu.edu.cn)
* College of Junior, Zhejiang Wanli University, Ningbo, China (sxxx@zwu.edu.cn)
** College of Electronic and Computer, Zhejiang Wanli University, Ningbo, China (chenzhigang@zwu.edu.cn, ssundechao123@163.com)
***State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

Robust Hashing [2]. This method is performance-wise, but it is no longer safe if the client-specific key is compromised.

(2) Biometric encryption method based on ECC. The method includes fuzzy vault (FV) [3] and fuzzy commitment [4]. Since this method requires strong restrictions on the accuracy of the authentication, there are some practical and security issues.

Most of the existing authentication systems are based on the improvement of the above biometric template protection scheme, and there are few authentication systems based on fully homomorphic encryption. Because the fully homomorphic encryption can calculate the ciphertext arbitrarily without knowing the key, the decryption result is the same as the corresponding calculation for the plaintext. Therefore, it is necessary to guarantee the security of the biometric authentication system by combining the protection scheme of biometric template with fully homomorphic encryption technology. Blanton and Gasti [5] proposed a security protocol for iris and fingerprint, homomorphic encryption algorithm using DGK scheme [6], their scheme takes only 150 ms to complete 2048-bit binary iris feature template Hamming distance, but the homomorphic encryption ciphertext is shorter, it is difficult to guarantee the security. Kulkarni et al. [7] proposed an iris authentication scheme based on the SHE scheme [8]. For the 2048 iris, the execution time of the server was 58 seconds, but their agreement was only three rounds. Karabat and Namboodiri [9] proposed an authentication protocol that uses a pure threshold homing encryption, whose scheme runs on the server side for 6.1 seconds, and the client runs for 2.1 seconds. Therefore, the application of fully homomorphic encryption technology still needs to consider the actual problems such as performance and encryption data size. One might consider a simple way to store a hash template in a server through a one-way hash function, such as SHA3 [10]. However, because of the scanning noise, the biological characteristics are not identical in each capture, so it is impossible to have the same hash value, which is not feasible. The authors [11] proposed a biometric authentication method based on fully homomorphic encryption using Helib library. They exploit SIMD operations to split biometric plaintext to small size segments. However, we find the segmented ciphertext modulus decreases little and SIMD (single instruction multiple data) operations are cost. Therefore, the segmentation does not significantly improve the efficiency and safety of the system.

In this paper, the scheme of FV and SEAL library implements a design based on the iris features of fully homomorphic encryption ciphertext authentication system, it does not depend on reliable hardware, using one-time MAC authentication, can very good protect user iris feature template and complete the corresponding iris certification, greatly enhance the system security.

2. Certification Scheme Design

2.1 FV Scheme

The FV scheme [12,13] is based on the polynomial ring R. It is used $R = Z[x] / (x^n + 1)$. Not only that the scheme also involves several parameters. n is the degree of the polynomial and it is always a power of two. λ is the safety parameter. q is a ciphertext modulus, which is used to reduce the coefficients of a ciphertext polynomial. t is a plaintext modulus, used to reduce the coefficients of a plaintext polynomial. Sample $\alpha \leftarrow S$ means that S is randomly selected uniformly from the finite set α and χ is an error

probability distribution on R. ω is the basis of decomposition of the whole coefficient. $\ell = \lfloor \log_\omega d \rfloor$ means dividing the integer d into ℓ parts. The concrete algorithm of the scheme is as follows:

KeyGen(λ): Sample $s \leftarrow R_2$ and output $sk = s$. sample $a_1 \leftarrow R_q$ and $e \leftarrow \chi$, output (sk, $\boldsymbol{pk}$) = (s, (a_0, a_1)).

EvKeyGen(sk, ω): For $i \in \{0,...,\ell\}$, sample $a_i \leftarrow R_q$ and $e_i \leftarrow \chi$, output $\boldsymbol{evk}$ = (($-(a_i s + e_i) + \omega^i s_2$) *mod* q, a_i).

Encrypt($\boldsymbol{pk}$, m): For $m \in R_t$, let $\boldsymbol{pk}$ = (a_0, a_1), sample $u \leftarrow R_2$ and e_1, $e_2 \leftarrow \chi$. Compute $c_0 = (\Delta m + a_0 u + e_1)$ *mod* q and $c_1 = (a_1 u + e_2)$ *mod* q. Output $\boldsymbol{ct}$ = (c_0, c_1).

Decrypt(sk, $\boldsymbol{ct}$): Let $\boldsymbol{ct}$ = (c_0, c_1) and $sk = s$, output $m' = ((t / q(c_0 + c_1 s))$ *mod* q) *mod* t.

Add($\boldsymbol{ct}_0$, $\boldsymbol{ct}_1$): Input $\boldsymbol{ct}_0$, $\boldsymbol{ct}_1$, output $\boldsymbol{ct}_0 + \boldsymbol{ct}_1$.

Mul($\boldsymbol{ct}_0$, $\boldsymbol{ct}_1$): Input $\boldsymbol{ct}_0$, $\boldsymbol{ct}_1$, output $\boldsymbol{ct}_0 \times \boldsymbol{ct}_1$.

2.2 Batching and Automorphism

Batching technique refers to the use of Chinese remainder theorem (CRT) and SIMD [14,15] to pack n numbers into a plaintext polynomial at a time, and to operate the polynomial is equivalent to a combination of the n number of the same operation (i.e., parallel).

The use of batching is conditional: plaintext mode t is prime and $t = 1 \pmod{2n}$. This means where $t > 2n$, which sometimes affects efficiency.

This condition ensures that the multiplicative group of the integer modulus t contains a subgroup with the number of elements $2n$. That is: $\zeta \in Z_t$ make $\zeta^{2n} = 1(mod\ t)$, and for all $0 < m < 2n$. The root of the $2n$ sub-primitive unit called the modulus t.

It is important to have such a primitive unit root, because the polynomial modulus $x^n + 1$ can be decomposed into the product of the following factors at modulus t:

$$x^n + 1 = (x - \zeta)(x - \zeta^3)...(x - \zeta^{2n-1})(\mathrm{mod}\, t) \tag{1}$$

According to the CRT, the ring R_t can be decomposed into:

$$R_t = \frac{Z_t[x]}{(x^n + 1)} = \frac{Z_t[x]}{\prod_{i=0}^{n-1}(x - \zeta^{2i+1})} \overset{CRT}{\cong} \prod_{i=0}^{n-1} \frac{Z_t[x]}{(x - \zeta^{2i+1})}$$

$$\cong \prod_{i=0}^{n-1} Z_t[\zeta^{2i+1}] \cong \prod_{i=0}^{n-1} Z_t \tag{2}$$

All of the isomorphisms are isomorphic on the ring, which means that both the addition and the multiplication structures are kept on both sides of the equation. The $\prod_{i=0}^{n-1} Z_t$ on the right, which can be expressed as $Z_t \times Z_t \times \ldots \times Z_t$, can also be regarded as an n-dimensional vector. So the addition of the two elements on the right needs to execute the addition of the n corresponding components. According to the isomorphism, the correspondence and the left are only the addition of the two polynomials on the ring R_t. The same multiplication is the same. Let $\alpha_i = \zeta^{2i+1}$ have Decompose:

$$R_t \to \sum_{i=0}^{n-1} Z_t, m(\alpha) \to [m(\alpha_0)m(\alpha_1),...,m(\alpha_{n-1})] \tag{3}$$

The same is the same in the opposite direction, which is called Compose. Compose and Decompose can be called packing and unpacking.

Automorphisms technique is a way in which the plaintext corresponding to each plaintext slot can be replaced. When the plaintext is $m(\alpha)$, the plaintext corresponding to each plaintext slot is $m(\alpha_0)$, $m(\alpha_1)$,…, $m(\alpha_{n-1})$. By using Frobenius automorphism, $m(\alpha) \rightarrow m(\alpha^{2^i})$ means that the plaintext slot moves i plaintext slots. For example, when $i = 1$, the plaintext slot of $m(\alpha)$ is cyclically moved by one position, and the plaintext corresponding to each plaintext slot becomes $m(\alpha_1)$, $m(\alpha_2)$,…, $m(\alpha_{n-1})$, $m(\alpha_0)$ [16].

Therefore, batching technique and automorphisms technique can be used to complete the cyclic movement operation of the plaintext in the corresponding plaintext slot in the case of ciphertext.

2.3 OTM Authentication

The message authentication code (MAC) generally uses the hash function of the portable key to verify the integrity of the transmitted data. Commonly used hash functions are MD5, SHA-2, and SHA-3. The storage of the iris feature ciphertext in this paper can use the above method to compress the ciphertext, and then the database only needs to store the ciphertext compressed summary. However, the authentication strategy designed in this paper is that the cloud server decrypts the result after the ciphertext's homomorphic operation, and the decryption operation can only be completed by the user. Therefore, the use of the hash function does not meet the needs of this design. Therefore, this paper designs a one-time MAC (OTM) authentication method, that is, the key generated by the message key algorithm in the MAC scheme can only be used once, and the user is secret. After decryption, the cloud server can authenticate the decryption result. The specific plan is as follows:

MKGen(Z_l): Set $\boldsymbol{mk} = (r_0, r_1)$, r_0 and r_1 are randomly selected from the Z_l, where Z_l is a set of l-bit integers.

MACGen($\boldsymbol{mk}$, m): The information authentication code m_c of the information m is obtained by calculating $m_c = m \times r_0 + r_1$.

Verification($\boldsymbol{mk}$, m, m_c): Input $\boldsymbol{mk}$, x and m_c, verify m is equal to $(m_c - r_0) / r_1$, if so, b is 1, otherwise b is 0.

2.4 Iris Ciphertext Recognition Method

At present, there are few iris recognition products on the market, mainly because the recognition accuracy of iris is largely dependent on the hardware facilities and recognition algorithms of iris acquisition [17,18]. In order to simulate the real iris recognition scene, this paper selects the public iris database, CASIA-Iris [18], to replace the iris acquisition process, using the public MATRIB code provided by the School of Computer Science and Software Engineering of the University of Western Australia [19] to preprocess the iris data, and in order to weigh the performance of the homomorphic calculation, the final use of 2048 The binary vector of bits represents the iris feature template.

The iris recognition method in this paper uses Hamming distance calculation to compare the encoded iris feature templates. It is to measure the distance between the two templates by counting the number of corresponding codes on the two templates [20]. The smaller the distance, the more the two templates match.

Suppose $\boldsymbol{A} = (a_0, a_1, \ldots, a_{n-1})$ and $\boldsymbol{B} = (b_0, b_1, \ldots, b_{n-1})$ represent two binary vectors of length n, whose Hamming distance refers to this The sum of two vector exclusive ORs. Therefore, it is defined as:

$$\mathrm{HD}(A,B) = \sum_{i=0}^{n-1}(a_i - b_i)^2 \tag{4}$$

In order to ensure the security of the iris feature template, this paper designs a recognition method based on iris ciphertext using the characteristics of fully homomorphic encryption technology. First, the purpose of this paper is to test homomorphic performance, transforming XOR into a combination of subtraction and multiplication when calculating Hamming distance. Secondly, since the FV scheme is built on the ring R, the iris feature template must be encoded into a polynomial. The usual practice is to use integer encoding. However, the iris initial template used in this paper is a binary vector of length n. It takes at least n multiplications to calculate the Hamming distance between two iris feature templates. However, the multiplication time between the ciphertexts of the iris feature template after fully homomorphic encryption is very slow, and the integer coding results in low computational efficiency, which is difficult to meet the actual system requirements.

Therefore, this paper uses the batching technique to pack the binary vector of length n into a polynomial, so that the XOR calculation of the vector is completed by one subtraction and one multiplication. At the same time, using the characteristics of automorphisms, only the $\log_2 n$ time shifts and $\log_2 n$ time additions to complete the calculation of the sum of the elements in the homomorphic ciphertext slot, that is, the ciphertext distance is calculated. As shown in Figs. 1 and 2, assuming that the vector $\boldsymbol{V} = (1, 2, 3, 4)$, its corresponding homomorphic ciphertext is $\boldsymbol{V'} = (v_1, v_2, v_3, v_4)$. Since the ciphertext slot is 4, only two-time shifts and two-time additions are required, where $(k_1, k_2) = (2^0, 2^1)$.

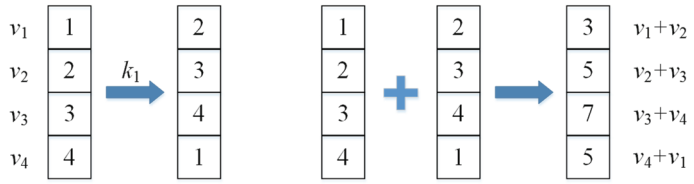


Fig. 1. Step 1 of calculating the sum of the elements in the homomorphic ciphertext slot.

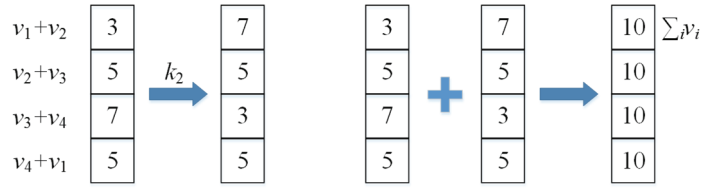


Fig. 2. Step 2 of calculating the sum of the elements in the homomorphic ciphertext slot.

In summary, the iris ciphertext recognition process in this paper is shown in Fig. 3. First, the client U packs the binary iris feature templates $\boldsymbol{A}$ and $\boldsymbol{B}$ into plaintext polynomials BP_A, $BP_B \in R_t$; then it encrypts BP_A, BP_B, output ciphertext polynomial $\boldsymbol{ct}_A$, $\boldsymbol{ct}_B \in R_q \times R_q$ and send it to the server S; the server S calculates the Hamming distance between the ciphertext polynomial $\boldsymbol{ct}_A$ and $\boldsymbol{ct}_B$, and outputs $\mathrm{HD}(\boldsymbol{ct}_A, \boldsymbol{ct}_B)$.

1. $[U]$ $(BP_A, BP_B) \leftarrow (Compose(A), Compose(B))$
2. $[U]$ $(ct_A, ct_B) \leftarrow (Encrypt(BP_A), Encrypt(BP_B))$
3. $[U \rightarrow S]$ (ct_A, ct_B)
4. $[S]$ $\mathrm{HD}(ct_A, ct_B) = \sum_{i=0}^{n-1} (ct_A - ct_B)^2$

Fig. 3. Iris ciphertext recognition method.

2.5 Iris Ciphertext Authentication Protocol

This paper encrypts biometrics templates and stores ciphertext through fully homomorphic encryption, and then measures similarity between two ciphertext templates. Finally, it is authenticated by OTM authentication. As shown in Fig. 4, the overall protocol of the system is only three rounds.

The first round, from step 1 to step 3 in the protocol diagram, the client U generates the secret key and public key through the *KeyGen* algorithms; then the client U acquires the original iris *Bio*, encodes the *Bio* using the CRT encoding, generates the polynomial BP_{Bio}, and then uses the *Encrypt* algorithm encrypts BP_{Bio} and generates ciphertext $\boldsymbol{ct}_{Bio}$. Finally, the client U sends a registration request (U_{id}, $\boldsymbol{ct}_{Bio}$) to the server S (i.e. user registration information and iris ciphertext). The server S stores (U_{id}, $\boldsymbol{ct}_{Bio}$) in the database and returns a success or failure message to the client.

1. $[U]$ $(sk, pk) \leftarrow (SecretKeyGen(\lambda), PublicKeyGen(sk))$
   $ct_{Bio} \leftarrow Encrypt(BP_{Bio} \leftarrow Compose(Bio))$
2. $[U \rightarrow S]$ (U_{id}, ct_{Bio})
3. $[U \leftarrow S]$ storage success or storage failure
4. $[U]$ $ct_x \leftarrow Encrypt(BP_x \leftarrow Compose(x))$
5. $[U \rightarrow S]$ (U_{id}, ct_x)
6. $[S]$ $ct_d = \mathrm{HD}(ct_{Bio}, ct_x)$
   $mk \leftarrow MessageKGen(Z_I)$
   $ct_T \leftarrow MACGen(mk, ct_d)$
7. $[U \leftarrow S]$ (ct_d, ct_T)
8. $[U]$ $d \leftarrow Decompose(BP_d \leftarrow Decrypt(ct_d))$
   $T \leftarrow Decompose(BP_T \leftarrow Decrypt(ct_T))$
9. $[U \rightarrow S]$ (d, T)
10. $[S]$ $b \leftarrow Verification(mk, m, m_c)$
11. $[U \leftarrow S]$ $b \in \{0,1\}$

Fig. 4. Iris authentication protocol.

The second round, from step 4 to step 7 in the protocol diagram, the client U obtains the current iris x, uses the CRT encoding to encode the x, generates the polynomial BP_x, and then uses the *Encrypt* algorithm to encrypt BP_x to generate the ciphertext $\boldsymbol{ct}_x$; then the client sends the authentication Request (U_{id}, $\boldsymbol{ct}_x$) to the server S; the server S calculates the Hamming distance between $\boldsymbol{ct}_{Bio}$ and $\boldsymbol{ct}_x$, output $\boldsymbol{ct}_d$ = HD($\boldsymbol{ct}_{Bio}$, $\boldsymbol{ct}_x$), then the server randomly selects (r_0, r_1) from Z_I, and outputs the message key $\boldsymbol{mk}$ = (r_0, r_1), and calculates the Hamming distance message authentication code $\boldsymbol{ct}_T$ through the $\boldsymbol{ct}_T = \boldsymbol{ct}_d \times r_0 + r_1$; The

last server S sends ($\boldsymbol{ct}_d$, $\boldsymbol{ct}_T$) to the client U.

Third, from step 8 to step 11 in the protocol diagram, the client U decrypts ($\boldsymbol{ct}_d$, $\boldsymbol{ct}_T$) using the *Decrypt* algorithm, and then performs CRT decoding on the decrypted result to generate plaintext (d, T); then sends it to the server S, and the server S verifies whether d is equal to $(T - r_0) / r_1$, the authentication result b is output and sent to the client. If the received b is 1 by the client U, the iris recognition result is not tampered, otherwise the iris recognition result has been tampered. Note: (d, T) is two integers at this time, they do not affect the security of the protocol, so the transmission process does not choose to encrypt it.

3. System Design and Implementation

3.1 System Model and Participants

This system mainly designs the verification server and uses one to one way to verify each user. The whole system consists of two participants, that is, the user and the authentication server. The user U has a binary feature template extracted from his iris feature. The server S has abundant computing resources and storage space, so it can complete any function calculation of homomorphic ciphertext, but cannot decrypt its generated ciphertext and user's ciphertext.

3.2 System Architecture

As shown in Fig. 5, the system adopts the C/S architecture. The main function of the client is to provide registration and authentication services for users, while the cloud server provides homomorphic computing and authentication services. The overall function is described below.

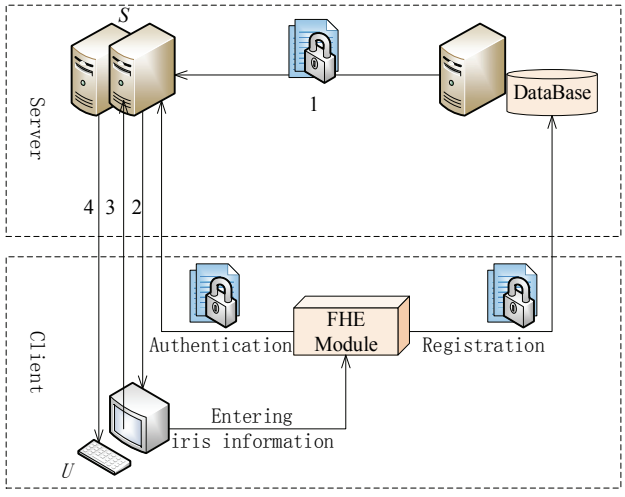


Fig. 5. System architecture diagram.

First, the client encrypts the user's iris feature information. If the client sends a registration request to the cloud server, the cloud server responds and stores the user iris feature ciphertext in the cloud database. If the client sends an authentication request to the cloud server, the cloud server responds and calls the

user iris ciphertext stored in the database, and then performs identification under the iris ciphertext with the current user iris ciphertext, and then calculates the ciphertext identification. The resulting MAC code is sent to the client. After the client responds, the user decrypts the ciphertext identification result and the corresponding MAC code with the secret key that he has mastered and sends it to the cloud server. The cloud server authenticates and returns. The result of the authentication is given to the client.

3.3 System Function Module

As shown in Fig. 6, the overall function of the system is divided into three modules. The first is the login module. When the user inputs the basic information that has been registered, the user can enter the authentication interface. Otherwise, you need to click the registration link to enter the registration interface. Followed by the registration module, its main functions are to generate the public key and secret key, save the key to the mobile memory and encrypt the iris information and store it with the user's basic information in the database; finally the authentication module, its function is encrypted the current iris information and sent to the server. After the server completes the iris authentication, the client decrypts the authentication result by the secret key and prompts the user.

3.4 Development Environment

This system was developed on Windows 10 using the C# programming language. The test used an HP notebook with an Intel Core i5-6200U processor. The homomorphic encryption algorithm library uses SEAL, which is a homomorphic encryption library developed by Microsoft Research using a C++ programming language. It has no external dependencies, so it is easy to compile in many different environments. At present, the homomorphic operations that have been implemented in the SEAL library mainly include Negate, Add, AddMany, Sub, Multiply, MultiplyMany, Square, Exponentiate, AddPlain, and MultiplyPlain.

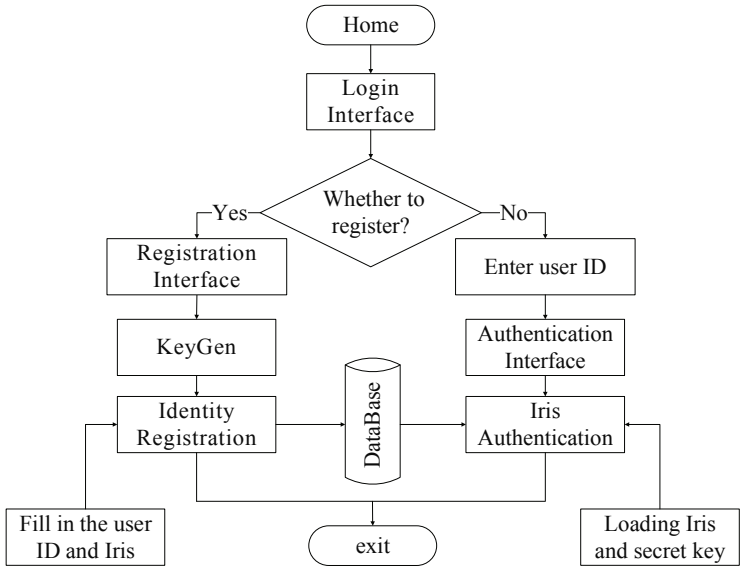


Fig. 6. System function module.

3.5 Parameter Optimization

The noise ceiling for each ciphertext in the SEAL library is fixed and is called the noise budget. Whenever a homomorphic operation is performed, the noise in the ciphertext increases. When the total amount of noise exceeds the noise budget, the ciphertext cannot be decrypted successfully. Therefore, while improving system efficiency and ensuring safety, it is also necessary to ensure that ciphertext can be decrypted successfully.

In order to encrypt a binary vector of 2048 bits, the polynomial number must be taken as $n \geq 2048$. However, when the parameter n is too large, the ciphertext calculation time is too long and the system efficiency is extremely low. Therefore, we study the 2048-bit binary vector segmentation. Table 1 shows the minimum value of $\log_2 q$ under the calculation of the ciphertext Hamming distance at different segments.

Table 1. Different segments of (n, $\log_2 q$) value comparison

Segment	n	$(\log_2 q)_{min}$
16	128	64
8	256	65
4	512	65
2	1024	67
1	2048	68

From the data in Table 1, it shows that the segmented ciphertext modulus q decreases little, so the segmentation does not significantly improve the efficiency and safety of the system. Also, Table 2 shows the maximum value of $\log_2 q$ for n = 1024, 2048, 4096, 8192, 16384 with a security level of 80 bits [20,21]. Based on the comprehensive analysis of the data in Tables 1 and 2, the final choice of $n = 2048$, $q = 2^{76} - 2^{22} + 1$, at this time to ensure that the security level of 80 or more at the same time, the ciphertext to complete the Hamming distance calculation noise growth does not exceed the noise budget.

Table 2. Security level 80 bit under the (n, $\log_2 q$) value comparison

Security level (bit)	n	$(\log_2 q)_{max}$
80	1024	47.5
80	2048	95.4
80	4096	192
80	8192	392.1
80	16384	799.6

4. System Analysis

4.1 Security Analysis

Network transmission security: An attacker can obtain only the homomorphic encrypted iris feature data or the data generated by the cloud server through random number calculation. Therefore, the network attacker can't use the acquired data to decipher the original iris feature plaintext data, and also can resist the replay attack.

Server security: Even an attacker can access the server's database. Because the iris templates stored in the server database are encrypted. These ciphertext templates do not reveal any information about the user's iris characteristics. If the user registers in multiple authentication servers based on this protocol, the keys obtained by the user are definitely different, so the user does not leak information when storing multiple templates in multiple server databases. If you suspect that a template has been corrupted, you can generate a new template with a different key.

Client security: Even if an attacker has access to the client system, he cannot obtain the iris feature template and secret key and cannot authenticate. If an attacker uses brute force, the effort required is equal to randomly assigning a bit vector. If the attacker wants to obtain the iris feature information in the server database by modifying the bits to be sent to the server, the OTM authentication method adopted in this paper can solve this problem well. Even an attacker can send a modification (d, T) to pass the authentication test. However, it is basically impossible for an attacker to achieve both iris recognition success and authentication success.

4.2 Homomorphic Performance

The following is the time taken for the system to test the fully homomorphic calculation process. The registration process is shown in Table 3.

Table 3. Registration part (unit: ms)

Operations	TN 1	TN 2	TN 3
CRT Compose	13.3	12.3	12.2
Encrypt	105.5	107.2	104.5
Total	1047.2	985.4	1000.8

The authentication part is used as shown in Table 4.

Table 4. Authentication part (unit: ms)

Operations	TN 1	TN 2	TN 3
Square	275.7	292.1	285.4
Sub	2.8	3.0	2.9
CRT Compose (r0, r1)	28.9	29.8	23.3
ctd * r0 + r1	194.3	202.7	188.9
Decrypt (ctd, ctT)	245.1	273.5	231.6
Decompose (ctd, ctT)	24.1	29.0	26.0
Total	2446.5	2544.8	2439.1

The test results are as follows:

In the registration part, the average registration time is 1011.1 ms, while the iris template encryption only takes 105.7 ms on average, accounting for 10.5% of the total registration time.

In the authentication part, the total time of authentication is 2476.8 ms on average, while the average of homomorphism is 482.6 ms, accounting for 19.5% of the total authentication time, and the average time of encrypting and decrypting ciphertexts is 355.8 ms, accounting for 14.4% of the total authentication time.

Since the tests are conducted locally, the performance of the system depends on the processing power of the notebook CPU. Through analysis, the system loads and communicates part of the graphical interface with the longest time, while the homomorphic encryption and decryption and the total time spent less than 40%. Therefore, the iris ciphertext full homomorphic efficiency is still good.

Below we compare the results with paper [11]. The length of biometric is 630 bits in [11], while the length is 2048 in our scheme. The normal size of irises is 2048 bits. We study the 2048-bit binary vector segmentation. Table 1 shows that the segmented ciphertext modulus q decreases little, so the segmentation does not significantly improve the efficiency and safety of the system. Therefore the 2048-bit binary vector segmentation is a good choice. In order to compare the performance with [11] at the same level, we also take 630 bits biometric of irises in our scheme. Table 5 shows that our results are better than the scheme [11].

Table 5. Comparison of two schemes (unit: ms)

Operations	Our scheme	Ghostshell [11]
Encryption	13.3	16.16
Decryption	105.5	163.72
Addition	0.05	0.05
Multiplication	10.2	14.32

5. Conclusion

With the rapid development of information technology, information security has become the most concerned point. The system combines homomorphic encryption with biometrics to ensure the security and integrity of user feature templates. In real life, such as online payment, account login, etc., biometrics can be used for identity verification, and the system can perform ciphertext calculation in the cloud, which greatly improves the security of data processing. It can be seen from the performance analysis that the efficiency of the system is good when the circuit depth of the fully homomorphic encryption calculation is not high. Despite this, the system is still far from the actual complex application requirements, and further research is needed.

Acknowledgement

This paper is supported by the Natural Science Foundation of Zhejiang Province of China (No. LY17F020002), Public Projects of Zhejiang Province (No. 2017C33079, LGG18F020001), Ningbo Natural Science Foundation (No. 2017A610120, 2018A610159), and the State Key Laboratory of Cryptology (No. 2017-MS-18).

References

[1] R. Belguechi, V. Alimi, E. Cherrier, P. Lacharme, and C. Rosenberger, “An overview on privacy preserving biometrics” in *Recent Application in Biometrics*. Rijeka, Croatia: InTech, 2011, p. 65-84.

[2] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM Systems Journal*, vol. 40, no. 3, pp. 614-634, 2001.

[3] A. Juels and M. Sudan, "A fuzzy vault scheme," *Designs, Codes and Cryptography*, vol. 38, no. 2, pp. 237-257, 2006.

[4] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in *Proceedings of the 6th ACM Conference on Computer and Communications Security*, Singapore, 1999, pp. 28-36.

[5] M. Blanton and P. Gasti, "Secure and efficient protocols for iris and fingerprint identification," in *European Symposium on Research in Computer Security*. Heidelberg: Springer, 2011, pp. 190-209.

[6] I. Damgard, M. Geisler, and M. Kroigard, "Homomorphic encryption and secure comparison," *International Journal of Applied Cryptography*, vol. 1, no. 1, pp. 22-31, 2008.

[7] R. Kulkarni and A. Namboodiri, "Secure hamming distance based biometric authentication," in *Proceedings of 2013 International Conference on Biometrics (ICB)*, Madrid, Spain, 2013, pp. 1-6.

[8] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, Bethesda, MD, 2009, pp. 169-178.

[9] C. Karabat, M. S. Kiraz, H. Erdogan, and E. Savas, "THRIVE: threshold homomorphic encryption based secure and privacy preserving biometric verification system," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, article no. 71, 2015.

[10] M. J. Dworkin, "SHA-3 Standard: permutation-based hash and extendable-output functions (NIST FIPS-202)," National Institute of Standards and Technology, Gaithersburg, MD, 2015.

[11] J. H. Cheon, H. Chung, M. Kim, and K. W. Lee, "Ghostshell: secure biometric authentication using Integrity-based homomorphic evaluations," *IACR Cryptology ePrint Archive*, vol. 2016, article no. 484, 2016.

[12] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *IACR Cryptology ePrint Archive*, vol. 2012, article no. 144, 2012.

[13] H. Chen, K. Laine, and R. Player, "Simple encrypted arithmetic library-SEAL v2.1," in *Financial Cryptography and Data Security*. Cham: Springer, 2017, pp. 3-18

[14] Z. Brakerski, C. Gentry, and S. Halevi, "Packed ciphertexts in LWE-based homomorphic encryption," in *Public Key Cryptography – PKC 2013*. Heidelberg: Springer, 2013, pp. 1-13.

[15] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations," *Designs, Codes and Cryptography*, vol. 71, no. 1, pp. 57-81, 2014.

[16] J. Deng, C. Xu, and H. Yang, "A secure computation scheme of inner product based on fully homomorphic encryption," *Journal of University of Electronic Science and Technology of China*, vol. 45, no. 5, pp. 808-811, 2016.

[17] S. Thavalengal, P. Bigioi, and P. Corcoran, "Iris authentication in handheld devices-considerations for constraint-free acquisition," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 2, pp. 245-253, 2015.

[18] CASIA iris database [Online]. Available: http://biometrics.idealtest.org.

[19] L. Masek and P. Kovesi, "MATLAB source code for a biometric identification system based on iris patterns," School of Computer Science and Software Engineering, University of Western Australia, 2003.

[20] Q. Tian and Z. Liu. "Survey of iris recognition," *Application Research of Computers*, vol. 25, no. 5, pp. 1295-1300, 2008.

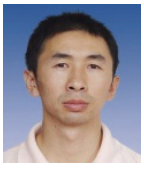
[21] M. R. Albrecht, "On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL," in *Advanced in Cryptology – EUROCRYPT 2017*. Cham: Springer, 2017, pp. 103-129.

Xinxia Song **https://orcid.org/0000-0003-4973-8295**

She received the B.Sc. degree in mathematics from Kashgar University in 1995, and the M.Sc. degree in mathematics from the Zhejiang Normal University in 2005. She is an associate professor at Zhejiang Wanli University. Her research interests include algebra and cryptography.

Zhigang Chen **https://orcid.org/0000-0001-5140-7319**

He received the B.Sc. degree in mathematics from Kashgar University in 1995, the M.Sc. degree in computer software and theory from the Northwest University in 2004, and received Ph.D. in the Nanjing University of Aeronautics and Astronautics in 2015. From 2013 to 2014, he was an academic visitor at Information Security Group of Royal Holloway, University of London. He is a professor at Zhejiang Wanli University. He is also a visiting researcher at State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences now. Currently his researches focus on fully homomorphic encryption, lattice-based cryptography and blockchain.



Dechao Sun **https://orcid.org/0000-0003-0771-093X**

He received Ph.D. in School of Information Science and Engineering from Ningbo University in 2018. He is an associate professor in Wanli University. His current research interests include artificial intelligence and network security.