Speech Encryption Scheme Based on BFV Homomorphic Encryption

Qiuyu Zhang, Yujiao Ba, Yingjie Hu, Zhenyu Zhao, and Yugui Jia
(Corresponding author: Qiuyu Zhang)

School of Computer and communication, Lanzhou University of Technology
No. 287, Lan-Gong-Ping Road, Lanzhou 730050, China
Email: zhangqylz@163.com, bayujiaolut@163.com
(Received Dec. 30, 2021; Revised and Accepted June 21, 2022; First Online July 3, 2022)

Abstract

To solve the problems of traditional speech encryption schemes in the cloud storage, such as data leakage, low efficiency of speech homomorphic encryption, and significant expansion of ciphertext data, a speech encryption scheme based on the Brakerski / Fan-Vercauteren (BFV) homomorphic encryption was proposed. Firstly, in order to reduce the computational complexity of homomorphic encryption, the original speech data was segmented by fixed-length segmentation technology to generate a series of two-dimensional arrays. Secondly, BFV homomorphic encryption algorithm was used to batch coding and encrypt the segmented speech arrays to improve the efficiency of speech homomorphic encryption. Finally, improved the ciphertext calculation algorithm and combined with the modulus switching and linearization technology to perform ciphertext operation on the encrypted speech data, which can reduce the length of the resulting ciphertext and ensure the correctness of the decryption. Experimental results show that the proposed scheme can resist the selective plaintext attack with low computational complexity and ciphertext expansion. The proposed scheme has higher security and encryption-decryption efficiency compared with the existing probabilistic statistics and additional homomorphic cryptosystem. It can be used to store speech data in cloud computing securely.

Keywords: BFV; Homomorphic Encryption; SEAL Library; Speech Encryption

1 Introduction

With the continuous development of cloud storage and multimedia technologies, cloud computing solves the pressure caused by massive speeches on local storage, but brings about the security problem of speech data leakage. Front-end encryption of speech data is one of the effective methods to solve the problem of cloud data leakage. However, data privacy depends on the access control policy of the cloud in traditional speech encryption schemes, resulting in the exposure of private data to cloud operators [21, 24]. Speech homomorphic encryption in cloud computing does not need to expose the key in Cloud Server and supports the data calculation function in the ciphertext domain, and provides feasible support for feature extraction in the ciphertext domain for the next step of speech recognition and speech retrieval [7].

The speech homomorphic encryption scheme uses the most advanced homomorphic encryption technology to ensure that while Cloud Server provides encrypted storage and computing functions, the speech data stored by the user will not be exposed in unencrypted form. Homomorphic encryption allows direct operations in the ciphertext domain, and after decryption, the same results can be obtained as those in the plaintext domain. Data privacy depends on the most advanced cryptography [25]. However, there are two limitations in homomorphic ciphertext operation: the limitation of operation types and large computational overhead [2]. Almost all fully homomorphic encryption only supports data of integer type, and requires a fixed multiplication depth, and cannot be added and multiplied infinitely [15, 18]. Homomorphic ciphertext operation is more time-consuming than plaintext operation, so speech homomorphic encryption schemes need to pay more attention to performance. Therefore, how to convert speech into integer data suitable for homomorphic encryption and improve the efficiency of homomorphic encryption has important research value.

Most of the existing speech encryption schemes are based on chaotic mapping, double chaotic mapping, combination of multiple chaotic mapping, hyperchaotic systems and other chaotic encryption methods [1]. Basically, they do not support data calculation in ciphertext domain, which means that users must share the decryption key with Cloud Server in order to carry out feature extraction, classification, matching, retrieval and other operations. With the continuous development of homomorphic encryption technology, it has attracted much attention in the field of speech encryption with its unique ciphertext

calculation algorithm. For example, Shi *et al.* [23] proposed a less data expansion probability statistics addition homomorphic speech encryption scheme, which effectively reduces the data expansion of ciphertext speech. Imran *et al.* [17] used the El-Gamal algorithm for the encryption and decryption of speech signals. The security of the system is based on calculating the discrete logarithmic modulus of large prime numbers. It will take thousands of years for an attacker to crack the password system. Badii *et al.* [4] proposed a speech accelerated encryption framework in cloud environment. The framework deployed partial homomorphic encryption and symmetric encryption, which has good security, but did not give the experimental performance of the algorithm. The above speech homomorphic encryption schemes are all based on PHE (Partially Homomorphic Encryption). In the cloud, only homomorphic addition or multiplication can be performed on ciphertext, which is not conducive to applying to the feature extraction of ciphertext speech in the field of speech recognition or speech retrieval. In terms of actual performance, the two most promising schemes of FHE (Fully Homomorphic Encryption) are BGV (Brakerski Gentry Vaikuntanathan) and BFV [16], and their security depends on RLWE (Ring Learning With Errors). Moreover, the addition of ciphertext data in BFV scheme can almost be considered as not consuming noise budget.

In summary, most of the existing speech encryption methods are based on chaotic mapping [1], and there is relatively little research on speech homomorphic encryption [4, 17, 23], and most of the existing speech homomorphic encryption schemes are based on PHE or SWHE (Somewhat Homomorphic Encryption), which is weak compared with FHE in ciphertext calculation. To solve the above problems, this paper uses the speech data in the TIMIT database [28] as the research object, and proposes a speech encryption scheme based on BFV homomorphic encryption. The scheme encrypts the speech data with matrix structure, which can encrypt more data at one time, making the operation efficiency of addition and multiplication higher, the complexity lower and the data expansion less. The main contributions of this work are as follows:

1) In order to find the best parameters and balance the performance of encryption, segmentation technology is used to reduce the computational complexity. The original speech data processed by positive integer is divided into multiple matrices according to the fixed length for evaluation, so that the computational complexity is reduced from the double logarithm relationship of the original speech data to the double logarithm relationship of the number of matrices.

2) Batching technology is used to reduce the computational complexity and improve the efficiency of encryption scheme. Pack multiple numbers into a plaintext polynomial, and use SIMD (Single Instruction Multiple Data) technology to operate multiple numbers in the plaintext slot of the polynomial in parallel.

3) Improve the ciphertext calculation algorithm of BFV, and combined modulus switching and relinearization technology to reduce the ciphertext expansion and noise of homomorphic encryption. Modulus switching and relinearization technology can reduce the length and noise of the resulting ciphertext speech, which can still decrypt the ciphertext correctly, which has a positive impact on noise consumption and performance.

The rest of the paper is arranged as follows. Section 2 gives the system model. Section 3 describes the BFV speech homomorphic encryption scheme and its processing process in detail. Section 4 analyzes the encryption performance of the proposed scheme. In Section 5, the proposed scheme is verified by experiments and compared with the existing methods. Section 6 summarizes the work of this paper.

2 System Model

Figure 1 shows a privacy threats model in cloud computing. This model has external attack threats and internal attack threats on cloud server. In order to improve the security of the speech encryption system in cloud computing.

Figure 2 shows the speech homomorphic encryption model of the proposed scheme. The system model consists of three entities: data owner (DO), cloud server (CS), and data user (DU). The work completed by each part is as follows:

Data Owner (DO): DO owns local speech data $m = \{m_1, m_2, \ldots, m_i\}$. To ensure the privacy and security of speech data, the speech data is preprocessed and homomorphic encrypted to obtain the speech ciphertext $c = \{c_1, c_2, \ldots, c_i\}$, where i represents the number of speech data. Finally, the generated speech ciphertext c is outsourced to CS for storage.

Cloud Server (CS): CS stores speech ciphertext c uploaded by DO and performs ciphertext calculations on c to obtain the new ciphertext c^*. When receiving DU's search request, CS returns the query result c^* to DU.

Data User (DU): After receiving the query result from CS, Du decrypts the key sent by DO to obtain speech plaintext.

3 The Proposed Scheme

The specific processing steps of the proposed speech homomorphic encryption scheme are as follows:

Step 1. Speech preprocessing. The digitized original speech data is segmented according to the fixed length, and the plaintext is packaged and encoded using batching technology to realize the parallel operation of encryption.

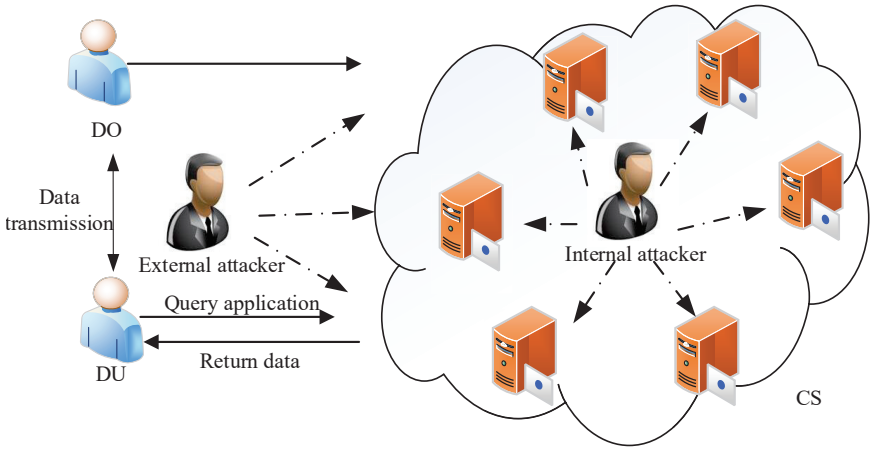


Figure 1: Privacy threat model in cloud computing.

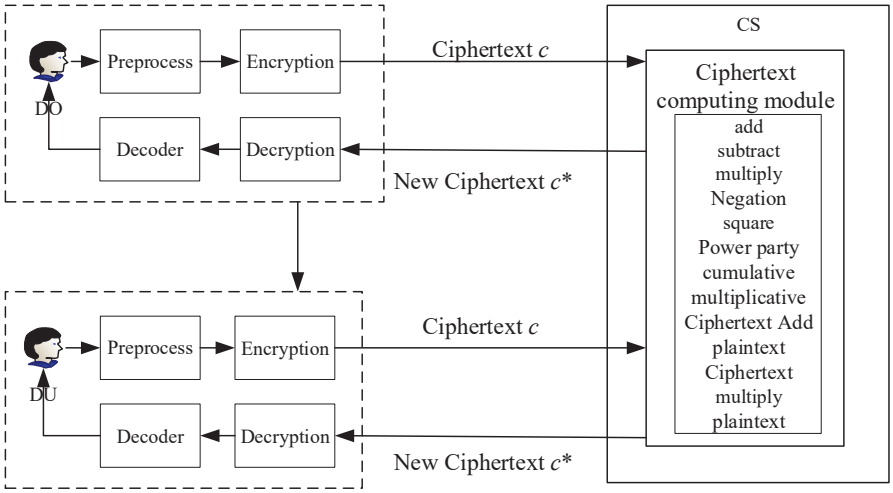


Figure 2: Speech homomorphic encryption model.

Step 2. Speech homomorphic encryption. BFV homomorphic encryption algorithm combined with batching technology is used to encrypt and decrypt speech data.

Step 3. Ciphertext calculation. Improve the ciphertext calculation algorithm of BFV homomorphic encryption for data operation in the ciphertext domain and use modulus switching and relinearization technology to reduce ciphertext expansion and noise.

Step 4. Speech reconstruction. Decrypt and decode the result of the ciphertext calculation and perform the inverse operation of the ciphertext calculation to obtain the decrypted speech data and restore it to the decrypted speech.

3.1 Speech Preprocessing

Figure 3 shows the speech preprocessing process, which is mainly composed of five parts: A/D, positive integer processing, segmentation, batching and cyclic coding.

The processing process of the speech preprocessing module is as follows:

Step 1. A/D. Analog to digital conversion, convert analog speech signals into digital speech data.

Step 2. Positive integer processing. In order to make the data suitable for homomorphic encryption processing and obtain the highest efficiency, data expansion and data migration are used to process all speech data into positive integers. Data expansion multiplies all data by 10^{φ} (φ is determined according to the data used, that is, φ significant bits are reserved, the parameter $\varphi = 6$ used in this paper), the floating-point number is processed as an integer. Data migration firstly finds the minimum value in the speech data and subtracts the value from all data, processes all the data as positive integers and stores them as a .txt file.

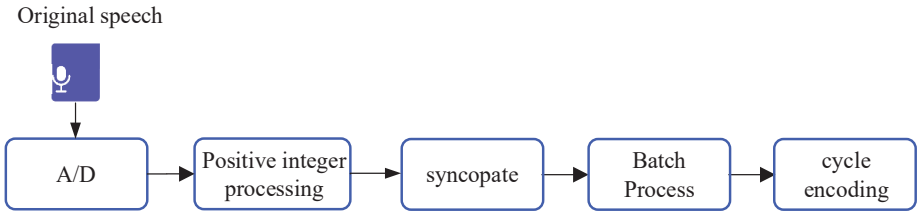


Figure 3: Speech preprocessing flowchart.

Step 3. Segmentation. The read speech data is segmented to a fixed length according to the polynomial modulus, and the last speech segment is filled with random numbers. Segmentation technology increases the amount of communication, but it also reduces the plaintext space and computational complexity.

Table 1 shows the encryption parameters of the proposed scheme. The polynomial modulus selected by the BFV scheme in this paper is 8192. Therefore, each matrix obtained by segmenting the speech data is 2×4096, which realizes the full utilization of space.

Table 1: Encryption parameters

Name	Parameter
Encryption	BFV
Polynomial module	8192
Ciphertext module	218(43+43+44+44+44) bits
Plaintext module	786433

Step 4. Batching. CRT and SIMD are used to package the segmented data into a series of matrices to realize parallel operation.

This paper uses batching technology (Batching) [10], that is, using the Chinese Remainder Theorem (CRT) [27] and SIMD [8] to pack n numbers into a plaintext polynomial. And the operation of this polynomial is equivalent to the same operation on the n numbers in the plaintext slot (i.e. parallel computing [22]).

To ensure that the multiplicative group of the plaintext module t contains a subgroup with 2^n elements, when $t = 1 (\mathrm{mod} 2^n)$ is a prime number, that is, batching can be used correctly when $t > 2^n$. That is, when $\xi \in Z_t$, $\exists \xi^m \neq 1 (\mathrm{mod} t), 0 < m < 2^n$, make $\xi^{2^n} = 1 (\mathrm{mod} t)$, ξ^{2^n} is the 2^n primitive unit root of the plaintext module t. The decomposition of the polynomial under the plaintext module t is as Equation (1):

$$x^{2^n} + 1 = (x - \xi)(x - \xi^3) \cdots (x - \xi^{(2^n - 1)}) \bmod t \quad (1)$$

That is, the polynomial of mod $x^{2^n} + 1$ is isomorphic with a set of n polynomials of mod $(x - \xi)$ to mod $(x - \xi^{(2^n - 1)})$ respectively. Let $d = 2^n$, according to CRT, R can be decomposed as Equation (2):

$$\begin{aligned} R = \frac{Z_t[x]}{(x^d+1)} &= \frac{Z_t[x]}{\prod_{i=0}^{d-1}(x-\xi^{i+1})} \\ &\cong \prod_{i=0}^{d-1} \frac{Z_t[x]}{(x-\xi^{i+1})} \\ &\cong \prod_{i=0}^{d-1} Z_t[\xi^{i+1}] \\ &\cong \prod_{i=0}^{d-1} Z_t \end{aligned} \quad (2)$$

$\prod_{i=0}^{d-1} Z_t$ can be expressed as $Z_t \times Z_t \times \ldots \times Z_t$, or as a d-dimensional vector. Therefore, the addition of the two elements on the right side needs to perform the addition of d corresponding components, which corresponds to the one-time addition of two polynomials on the ring R_t on the left side, that is, it has additive homomorphism. Similarly, multiplication has multiplication homomorphism.

SIMD enables multiple execution components to access memory at the same time, and the feature of obtaining all operands once for calculation makes SIMD suitable for intensive calculations of multimedia data. The coefficients of plain text polynomials (plain text slots) are obtained by using CRT in polynomial rings. SIMD can be used to insert plain text arrays into slots When performing a homomorphic addition operation on two ciphertexts, the array vectors performed a homomorphic addition operation by components to improve the calculation efficiency and reduce the ciphertext conversion rate. The total number of batch "slots" equals the system modulus t of the polynomial, which are organized into $2-(t/2)$ matrices that can be encrypted and calculated, each slot contains an integer that modulates the plaintext modulus.

Step 5. Cyclic coding. The Encode function of the SEAL library is used to encode all batch packaged matrices in plaintext polynomials for subsequent batch encryption operations.

3.2 Speech Homomorphic Encryption

The BFV homomorphic encryption scheme contains four algorithms: key generation algorithm (**GenKey**), encryption algorithm (**Enc**), decryption algorithm (**Dec**) and ciphertext calculation algorithm (**Eval**), where ciphertext calculation refers to the mathematical operations performed on the ciphertext domain. The above

algorithms are all probabilistic polynomial time (PPT) algorithms [11].

3.2.1 System Parameters

Table 2 shows the definitions of symbols in this paper.

Table 2: The definitions of symbols

Symbol	Definition
R	Polynomial ring
R_t	Plaintext space with plaintext mod $t > 1$
Z_q	Set of integers $(-q/2, q/2]$ with ciphertext mod $q > 1$
R_q	The set of polyno in R with coefficient Z_q
$f(x)$	Unreducible polynomial of degree d
$Z[x]$	Set of unreducible polynomials of degree d
α_i	Coefficient of element $\mathbf{a}$ of ring R
$\|\|\mathbf{a}\|\|$	Infinite norm of element $\mathbf{a}$ of ring R
δ_R	R expansion factor
$[a]_q$	The only integer in Z_q
$[\mathbf{a}]_q$	The set of integers obtained by applying $[\cdot]_q$ to all the coefficients of the element $\mathbf{a}$ of the ring R
D	Given probability distribution
χ	Discrete Gaussian distribution on integers
B	B-bounded bounds

The expansion factor of the polynomial ring $R = \frac{Z[x]}{f(x)}$ is: $\delta_R = max\{||\mathbf{a} \cdot \mathbf{b}||/(||\mathbf{a}|| \cdot ||\mathbf{b}||), \mathbf{a}, \mathbf{b} \in R\}$, where $f(x) = x^d+1$, $d = 2^n$. The infinite norm $\mathbf{a} = \prod_{i=0}^{d-1} a_i \cdot x^i$ of the element $||\mathbf{a}|| = max_i|a_i|$ of the ring R.

For $a \in Z$, the modular operation can be described as $[a]_q = a \bmod q$, where $[\mathbf{a}]_q$ represents the set of all integers obtained by the modular operation of all coefficients of element $\mathbf{a}$. q is the ciphertext modulus, and $l = \log_\omega q$ means that the integer q is decomposed into l parts according to the base ω. For a given probability distribution D, $x \leftarrow D$ means that x is randomly and uniformly sampled from the probability distribution D. When B is large enough, if it obeys the discrete Gaussian distribution χ on integers on $[B, B]$, it is called B-bounded.

3.2.2 Encryption Algorithm Description

The BFV homomorphic encryption scheme can be equivalent to a circuit model $C(c_1, c_2, \cdots, c_\lambda) \in C_\lambda$, let C_λ be the circuit set, and its algorithm is described as follows:

1) Key generation algorithm $\mathbf{GenKey}(1^\lambda)$. Input λ, output sk, pk and evk. Take the element $\mathbf{s} \leftarrow \chi$, and output the private key $sk = \mathbf{s}$. Take element $\mathbf{a} \leftarrow R_q$, error $\mathbf{e} \leftarrow \chi$, let $s = sk$, and output public key $pk = \left([-(\mathbf{a}s + \mathbf{e})]_q, \mathbf{a}\right)$. This obtains a pair of public and private keys (pk, sk). Let $i \in \{0, \cdots, l\}$, $\alpha_i \leftarrow R_q$, the error $\mathbf{e}_i \leftarrow \chi$, and the output ciphertext calculation key $evk = ([-(\alpha_i s + \mathbf{e}_i) + \omega^i s^2]_q, \alpha_i)$.

2) Encryption algorithm $\mathbf{Enc}(m, pk)$. Input the public key $p = pk$ and the plaintext $m \in R_t$, take element $\mathbf{u} \leftarrow \chi$, error $\mathbf{e}_1, \mathbf{e}_2 \leftarrow \chi$, output ciphertext $c = ([\Delta \cdot m + p[0]\mathbf{u} + \mathbf{e}_1]_q, [p[1]\mathbf{u} + \mathbf{e}_2]_q)$.

3) Decryption algorithm $\mathbf{Dec}(c', sk)$. Input private key $s = sk$, ciphertext c, output decrypted plaintext $m' = \left[\frac{t}{q}[c[0] + c[1] \cdot s]_q\right]_t$.

4) Ciphertext calculation algorithm $\mathbf{Eval}(evk, C, c)$. Input the ciphertext calculation key evk, circuit $C(c_1, c_2, \cdots, c_\lambda) \in C_\lambda$and ciphertext $(c_1, c_2, \cdots, c_\lambda)$, and output the ciphertext calculation result c'.

Based on the above BFV homomorphic encryption algorithm, the proposed speech homomorphic encryption algorithm is shown in **Algorithm 1**.

In Algorithm 1, given the speech array $swp[b][4096]$, $(b = 16)$, initialize the speech matrix pod, plain text polynomial sequence $plain$, ciphertext sequence $encrypted$, decrypt polynomial $replain$, and decode speech matrix $repod$;

Algorithm 1 Speech homomorphic encryption algorithm based on BFV

```
Loop
for x = 1 to b/2 do
    Pack the swp data into b/2 speech matrix pod;
end for
Loop
for y = 1 to b/2 do
    Encode pod into a plaintext polynomial plain;
    Encrypt plain into a ciphertext sequence encrypted;
end for
Decrypt and verify encrypted respectively;
Decode and verify replain respectively.
```

3.3 Ciphertext Calculation

Eval is an important module of the homomorphic encryption scheme and a random algorithm. It takes the system parameter $Params$, the ciphertext calculation key evk, two ciphertexts c_1 and c_2 as input and outputs the ciphertext c_3. The correctness of **Eval** is that if c_1 is the encryption of the plaintext element m_1 and c_2 is the encryption of the plaintext element m_2, then c_3 should be the encryption of the operation of m_1 and m_2.

3.3.1 Homomorphism

1) Addition homomorphism: Add(c_0,c_1)

Input two ciphertexts c_0 and c_1, and the sum of these two ciphertexts is obtained by calculating Equation (3):

$$c_{add} = \left[[c_0[0] + c_1[0]]_q, [c_0[1] + c_1[1]]_q\right] \quad (3)$$

2) Multiply homomorphism: Multiply(c_0, c_1)

Input two ciphertexts c_0 and c_1, and the final result (c'_0, c'_1) obtained by the calculation of Equation(5) and Equation(6) is the product of these two ciphertexts $c_0 \times c_1$.

$$\begin{cases} c_0 = \left[\frac{t}{q}\left(c_0[0] + c_1[0]\right)\right]_q \\ c_1 = \left[\frac{t}{q}\left(c_0[0]c_1[1] + c_0[1]c_1[0]\right)\right]_q \end{cases} \tag{4}$$

$$c_2 = \left[\frac{t}{q}\left(c_0[1] + c_1[1]\right)\right]_q = \sum_{i=0}^{l} c_2^{(i)}\omega^i \tag{5}$$

$$\begin{cases} c'_0 = c_0 + \sum_{i=0}^{l} evk[i][0]c_2^i \\ c'_1 = c_1 + \sum_{i=0}^{l} evk[i][1]c_2^i \end{cases} \tag{6}$$

3.3.2 Improved Ciphertext Calculation Algorithm

Microsoft's Simple Encrypted Arithmetic Library (SEAL) [26] is an open source software library based on the BFV homomorphic encryption scheme. The library supports the RNS variant of the BFV scheme [13], which enable ciphertext operations on integers.

Eval enables encrypted speech data to perform feature extraction, classification and other operations in speech recognition, speech retrieval and other systems in the encrypted domain. To reduce the cipher noise of homomorphic encryption, an improved BFV's **Eval** based on speech data as shown in **Algorithm 2**.

Algorithm 2 Improved BFV ciphertext calculation algorithm

```
Loop
for i = 1 to b/2 do
    Perform interleaved AddPlain on the ciphertext sequence;
end for
Loop
for i = 1 to b/2 do
    AddMany b/2 ciphertext sequences;
    Perform MulPlain;
    Perform Square;
end for
Relinearize the calculated ciphertext.
```

The given ciphertext sequence is encrypted and the improved **Eval** in this paper can be described as:

AddPlain. Add the ciphertext to the plaintext. The ciphertext + plaintext calculation is a random algorithm that takes $Params$, evk, c_1, and m_2 as input, and outputs c_3. The correctness of AddPlain is: if c_1 is the encryption of m_1, then c_3 should be the encryption of $m_1 + m_2$.

AddMany. Accumulate the ciphertext calculated.

MulPlain. Multiply the ciphertext with the plaintext. Same as AddPlain, input $Params$, evk, c_1, and m_2, output c_3. The correctness of MulPlain is: if c_1 is the encryption of m_1, then c_3 should be the encryption of $m_1 \times m_2$.

Square. Perform a square operation on the result of the above ciphertext operation.

Relinearize. Relinearize the result of the above ciphertext operation.

Then, ciphertext calculation on encrypted data requires 1 MulPlain operation, 1 Square operation, and 15 AddPlain operations.

3.3.3 Modulus Switching and Relinearisation Technology

Modulus switching [20] is a technology that converts encryption parameters along the modulus chain to improve the efficiency of time and space. Noise is formed during the homomorphic operation, and the original ciphertext is expanded so that can be swapped down the modulus chain [9]. Table 3 shows the modulus switching process.

Table 3: The process of modulus switching

Ciphertext module	Modulus chain
$q_0q_1q_2q_3$	3(highest)
$q_0q_1q_2$	2
q_0q_1	1
q_0	0(lowest)

If c is the original ciphertext after one or more homomorphic operations, the ciphertext modulus $q = q_0q_1q_2q_3$ is the four different prime numbers of c. The first modulus switching redistributes the information of module q_3 to the q_0, q_1, and q_2 modules of c, which is usually done in the same way in the Residue Number System (RNS), and c is generated from the modulus $q_0q_1q_2$. Similarly, the c obtained from the second modulus switching is essentially the ciphertext generated based on the modulo q_0 and q_1. The c obtained from the third modulus switching is essentially the encrypted ciphertext of the module q_0. The BFV scheme indexes the prime factors of q in the order of decreasing size, so each modulus switching deletes the current smallest factor [19].

Since FHE's ciphertext has a linear size in $\log(q)$, the rounding division operation can reduce the ciphertext length from $\log(q)$ to $\log(q')$, while the proportion of noise remains roughly unchanged, that is, the ciphertext is compressed. To still decrypt the ciphertext correctly, simply make the noise ratio higher than the threshold value. With the encryption parameter settings selected in this paper, the communication volume can be reduced by about 5 to 10 times.

The BFV scheme in this paper uses a relinearization operation to reduce the size of the original ciphertext, and the plaintext corresponding to the ciphertext remains unchanged. The size of the BFV ciphertext is the number of polynomials it contains. Initially, the ciphertext consists of 2 polynomials with a size is 2. After each multiplication operation, the size of the resulting ciphertext becomes $M + N - 1$, where M and N are the sizes of the ciphertext participating in the multiplication operation. Although the ciphertext can still be decrypted normally when size becomes larger, it has two negative effects: First, the computational overhead of multiplication and addition will increase. Each ciphertext multiplication requires $O(M \times N)$ degree polynomial multiplication, while ciphertext addition requires $O(M + N)$. Second, each multiplication operation consumes more noise. Relinearization returns the size of the ciphertext back to the minimum of 2, thus providing greater performance benefits.

The relinearization operation requires the creation of corresponding Relinearization Key (Rlk), and each operation from size M to 2 requires $M - 2$ Rlk. An error will occur if the size of the ciphertext requiring repainting does not match the number of keys for the repainting operation. However, the relinearization operation itself has computational overhead and noise consumption and is related to the decomposition bit count parameter (denoted as w). w is an integer between 1 and 60. If w is smaller, the calculation speed is slower, but the noise consumption per operation is smaller. Conversely, if w is larger, the calculation speed is faster, but the noise consumption is higher for each operation. When the value of w is large, it corresponds to a noise budget threshold under each set of encryption parameter. When the noise budget of the ciphertext is above the threshold, each relinearization operation will consume more noise budget. Once the noise budget of the ciphertext is reduced below the threshold, the noise budget consumption for relinearization will drop to a small amount.

3.4 Speech Reconstruction

The processing process of the speech reconstruction is as follows:

Step 1. Decoding. Decode the decryption polynomial *replain* into a decoded speech matrix *recode*.

Step 2. Matrix processing. Process the decoded speech matrix recode, discard the last k random numbers of the last unit of the decode matrix, and store the matrix elements as a one-dimensional array.

Step 3. Inverse operation. Perform the inverse operation of the ciphertext operation on the processed one-dimensional array, and restore it to the positive integer data of the decrypted speech data.

Step 4. Restore data. Through the inverse process of data movement and data expansion, the positive integer data is restored to speech data using the parameters (minimum value and φ) used for preprocessing.

Step 5. Restore speech. Set the sampling rate, number of channels, sampling depth, etc. Then synthesize the speech data into a speech file to obtain the decrypted speech.

4 Encryption Performance Analysis

The speech used in the experiment is the speech in the TIMIT speech library [28]. The frequency of the speech data is 16kHz, and the sampling accuracy is a 4-second speech segment in 16bit single-channel wav format. Select the speech in the speech database for experiments, and use PyCharm for speech preprocessing to obtain speech integer data as the database. If the data is more complex speech data (such as: each speech has a different length and a large difference, the speech data is multi-channel data, etc.), the multi-channel speech data is converted into single-channel speech data, and the speech data of different duration is divided into different numbers of matrix data for subsequent algorithms. The homomorphic encryption of speech data is realized through Visual Studio 2019 and part of the custom code of Microsoft's SEAL homomorphic encryption library.

Experimental hardware environment: Intel(R) Core (TM) i5-8250U CPU, 1.80GHz, RAM 12GB.

Software Environment: Windows 10, PyCharm, Matlab, Visual Studio 2019.

4.1 Correctness Analysis

This section uses the symbols and definitions in Section 3.2.1. In order to prove that the proposed speech homomorphic encryption scheme has the correctness of decryption, according to the definition of modular operation [12]:

$$\begin{aligned} &c_0 + c_1 \cdot \mathbf{s} = p[0] \cdot \mathbf{u} + \mathbf{e_1} + \Delta \cdot m + p[1] \cdot \mathbf{u} \cdot \mathbf{s} + \\ &\mathbf{e_2} \cdot \mathbf{s} \bmod q = \Delta \cdot m + \mathbf{e} \cdot \mathbf{u} + \mathbf{e_1} + \mathbf{e_2} \cdot \mathbf{s} \bmod q \end{aligned} \tag{7}$$

The following Lemma 1 can be proved [13].

Lemma 1. Suppose $\Delta = [q/t]$, $r_t(q) = q \bmod t$, then $q = \Delta \cdot t + r_t(q)$. v is the noise contained in the ciphertext, assuming $||\chi|| < B$, if Equation (8) holds:

$$[c_0 + c_1 \cdot \mathbf{s}]_q = \Delta \cdot m + v \tag{8}$$

where $||v|| \leq 2 \cdot \delta_R \cdot B^2 + B$, that is, when $2 \cdot \delta_R \cdot B^2 + B < \Delta/2$, the proposed scheme has the correctness of decryption.

4.2 Security Analysis

The proposed scheme defines the security of the scheme through chosen plaintext attacks (CPA), and uses the Game-Hopping [3] method to verify the security of the

proposed scheme. Use indistinguishability (IND) [14] to verify semantic security. The verification design is divided into adversary and challenger. The specific description is as follows:

Game 0: Challenger B obtains the key pair (pk, sk) through the key generation algorithm **GenKey** (1^{λ}) of the BFV speech homomorphic encryption scheme, and then sends pk to attacker A. Challenger B announces the ciphertext $c = \mathbf{Enc}_{pk}(m_b), b \in \{0,1\}$, and attacker A needs to crack the plaintext m_b corresponding to c. The probability that the attacker A wins in Game 0 is Equation (9):

$$Adv_{IND-CPA}(A) = \left| \begin{array}{l} \Pr\left[A\left(pk, \mathbf{Enc}_{pk}(m_0)\right) = 1\right] \\ -\Pr\left[A\left(pk, \mathbf{Enc}_{pk}(m_1)\right) = 1\right] \end{array} \right| \tag{9}$$

Game 1: pk of Game 1 is directly selected randomly in R_q. The output samples of uniform distribution on R_q and discrete Gaussian distribution are indistinguishable. Therefore, attacker A cannot distinguish Game 1 from Game 0, that is, the probability of attacker A winning in Game 1 is equivalent to Game 0, which can be described as Equation(10):

$$|Adv_{1IND-CPA}(A) - Adv_{IND-CPA}(A)| = 0 \tag{10}$$

Game 2: c is randomly selected uniformly within the range of $U\{0,1\}$, and is no longer generated by the encryption algorithm. In Game 2 and Game 1, the advantage of attacker A is mainly to solve the RLWE problem. Therefore, the relationship between the probability that the attacker A wins in Game 2 and Game 1 can be described as Equation (11):

$$\begin{array}{l} |Adv_{2IND-CPA}(A) - Adv_{1IND-CPA}(A)| \\ = \mathrm{RLWE}_{d,q,\Omega} Adv(A) \end{array} \tag{11}$$

At this time, c and pk provided by the challenger B are all random, that is, they are not related to m_b, $b \in \{0,1\}$. Therefore, attacker A has no advantage in Game 2, so the probability of attacker A winning in Game 2 is Equation (12):

$$Adv_{2IND-CPA}(A) = 0 \tag{12}$$

Can be obtained from the above:

$$\begin{array}{c} Adv_{IND-CPA}(A) = \mathrm{RLWE}_{d,q,\Omega} Adv(A) \\ +\mathrm{RLWE}_{d,q,\Omega} Adv(A) \end{array} \tag{13}$$

Guessing: The adversary guesses the value of b and outputs the guessed value b'. If $b = b'$, the attacker's attack is successful.

In summary, the advantage of any attacker A in the above game is defined as:

$$\left| \Pr[b' = b] - \frac{1}{2} \right| \tag{14}$$

If the advantage is negligible, it proves that the attacker will not win the game. From Equation (13), $Adv_{IND-CPA}(A)$ can be ignored under the difficult assumption of $\mathrm{RLWE}_{d,q,\Omega}$. That is, the proposed scheme is semantically indistinguishable under the standard model and has semantic security [6]. The proposed scheme conforms to the IND-CPA [5] security model.

5 Experimental Results and Analysis

5.1 Noise Budget

There are four sources of noise: AddPlain, AddMany, Square, MultPlain, Relinearize. The purpose of the noise budget is to verify the parameters and is carried out during the design stage. Table 4 shows the noise budget consumed by the algorithm in this paper.

Table 4: The noise budget consumed

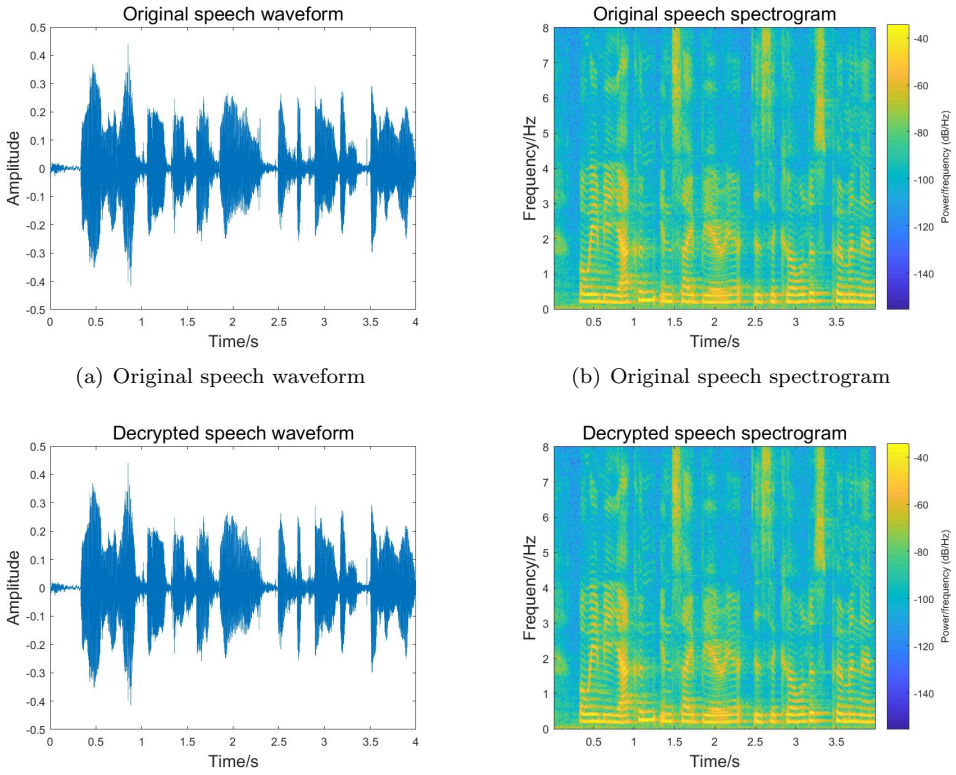
Algorithm	Noise budget(bit)
AddPlain	0
AddMany	1
Square	33
MultPlain	33
Relinearize	0

The noise budget value in Table 4 is measured along the BFV homomorphic encryption circuit. AddPlain and AddMany in this scheme can almost be regarded as no noise budget; Relinearize does not consume the noise budget. Noise will affect the correctness of decryption. Once the noise is lower or higher than the threshold, it will lead to the failure of decryption. Figure 4 shows the waveform and spectrogram of the original speech and the decrypted speech.

It can be seen from Figure 4 that the decrypted speech waveform and spectrogram are visually identical to the original speech, indicating that the proposed scheme has the correctness of decryption. In addition, by analyzing the correlation between the original speech and the decrypted speech signal, the correlation coefficient between the decrypted speech and the original speech is around ± 1, which shows the correctness of the decryption and also shows that the speech restoration and reconstruction performance is strong.

5.2 Ciphertext Calculation Performance

The total number of pre-processed speech data is 64,000, and the direct encryption needs to be performed 64,000 times to obtain 64,000 ciphertexts. To perform ciphertext calculations on these ciphertexts, a simple accumulation algorithm (Addmany) also needs to be performed 63999 times. In this paper, the speech data is batch-processed,



(a) Original speech waveform

(b) Original speech spectrogram

(c) Decrypted speech waveform

(d) Decrypted speech spectrogram

Figure 4: Waveform and spectrogram of the original speech and decrypted speech.

and the data is packaged into 8 plaintext matrices. When encrypting, only 8 batches of matrix encryption are required, and 8 ciphertexts are obtained. Table 5 shows the performance of various calculations performed by the BFV speech homomorphic encryption algorithm.

Table 5: Ciphertext calculation performance test

Algorithm	Computation time(s)
Encryption	8.117
Decryption	1.566
Addition	0.141
Multiplication	16.649
AddPlain	0.021
MultPlain	2.487
Square	12.023
Relinearize	4.671

It can be seen from Table 5 that the ciphertext addition is about 7 times slower than AddPlain. The ciphertext addition is equivalent to AddPlain, but the efficiency of AddPlain is much faster than the ciphertext addition. Therefore, in order to improve the efficiency of ciphertext calculation, this paper replaces ciphertext addition with AddPlain. Similarly, can replace ciphertext multiplication with MultPlain.

5.3 Ciphertext Calculation Performance

The proposed scheme uses batching technology in the encryption domain to realize the parallel operation of large integers, and adopts a matrix structure, which can encrypt more samples at a time. And in the ciphertext calculation module combined with modulo exchange technology and re-linearization to reduce the expansion of ciphertext. Table 6 shows the comparison between the ciphertext data expansion and the probability encryption system generated by the proposed scheme.

Table 6: Ciphertext data expansion comparison

Evaluation index	Proposed	Ref. [28]
Original data (KB)	126	126
Ciphertext data (KB)	3344	-
Ciphertext expansion	26.5397	6.6667×10^6

From the data analysis in Table 6, we can see that the probability encryption system has more data expansion than the proposed method. This paper uses modular exchange technology and re-linearization technology to effectively reduce the ciphertext expansion in the ciphertext calculation process.

In this section, the performance of the proposed scheme is compared with the latest chaotic encryption scheme, improved probabilistic statistical addition homomorphic encryption scheme and El-Gamal encryption scheme pro-

posed in the existing scheme [1,17,23], and the proposed scheme is objectively and accurately evaluated. All experiments use speech data with a 4s TIMIT data set speech file 10 times on a PC, and take the average value of each item for comprehensive evaluation, as shown in Table 7.

As can be seen from Table 7, the overall performance of the proposed scheme is superior to the speech homomorphic encryption schemes proposed in [23] and [17]. The encryption efficiency of the proposed scheme is nearly 2 times faster than that of the probabilistic encryption system proposed in [23], and the decryption efficiency is nearly 6 times faster than that of the probabilistic encryption system. This is because the proposed scheme uses the segmentation and batching techniques to realize the parallel computation of multiple data. Although the encryption and decryption efficiency of the proposed scheme is not as good as the speech chaotic encryption scheme proposed in [1], the proposed scheme has higher security and can support ciphertext domain data calculation to reduce the time consumption of subsequent calculation. Therefore, the proposed scheme can be used to protect the privacy of speech data storage in public cloud.

6 Conclusions

In this paper, a speech encryption scheme based on BFV homomorphic encryption is proposed, which solves the risks of data privacy exposure of traditional speech encryption methods in the cloud and the problems of the low efficiency of encryption and the large expansion of encrypted data in existing speech homomorphic encryption schemes. The ciphertext calculation function supported by homomorphic encryption is used to solve the problem of original data being exposed to the cloud server. By combining segmentation and batching technology, the proposed scheme performs batch polynomial encoding and encrypting of speech data, reducing the plaintext space to improve the efficiency of speech homomorphic encryption. And modulus switching technique and relinearization technique are adopted to decrease the length of encrypted data to reduce the amount of ciphertext expansion. Experimental results show that the proposed scheme can effectively improve the efficiency of homomorphic speech encryption and reduce the expansion of ciphertext data on the premise of ensuring the correctness of decryption. And it also has high security and can resist selected plaintext attacks at the same time.

In future work, we will further research the ciphertext calculation module of speech homomorphic encryption to solve the problem of feature extraction in ciphertext speech data, so as to improve the security and efficiency of speech recognition and speech retrieval systems.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61862041). The authors would like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[1] O. M. Al-Hazaimeh, "A new speech encryption algorithm based on dual shuffling henon chaotic map," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 3, pp. 2203–2210, 2021.

[2] A. Alabdulatif, I. Khalil, X. Yi , "Towards secure big data analytic for cloud-enabled applications with fully homomorphic encryption," *Journal of Parallel and Distributed Computing*, vol. 137, no. 0743-7315, pp. 192–204, 2020.

[3] M. R. Albrecht, M. Chase, H. Chen, "Homomorphic encryption standard," *IACR Cryptol. ePrint Arch.*, vol. 2019, pp. 939, 2019.

[4] A. Badii, R. Faulkner, R. Raval, "Accelerated encryption algorithms for secure storage and processing in the cloud," in *In 2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP). IEEE*, pp. 1–6, Fez, Morocco, May 2017.

[5] M. Barcau, V. Pasol, "On the ind-cpa security of ring homomorphic encryption schemes over f2," *Proceedings of the Romanian Academy Series A-MathEmatics Physics Technical Sciences Information Science*, vol. 21, no. 1, pp. 3–10, 2020.

[6] D. A. Basin, A. Lochbihler, S. R. Sefidgar, "Crypthol: Game-based proofs in higher-order logic," *Journal of Cryptology*, vol. 33, no. 2, 2020.

[7] V. Biksham, D. Vasumathi, "A lightweight fully homomorphic encryption scheme for cloud security," *International Journal of Information and Computer Security*, vol. 13, no. 4, pp. 357–371, 2020.

[8] E. Boyle, L. Kohl, P. Scholl, "Homomorphic secret sharing from lattices without fhe," in *In 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Cham*, pp. 3–33, Darmstadt, Germany, May 2019.

[9] X. Che, T. Zhou, N. Li, "Modified multi-key fully homomorphic encryption based on ntru cryptosystem without key-switching," *Tsinghua Science and Technology*, vol. 25, no. 5, pp. 564–578, 2020.

[10] H. Chen, R. Gilad-Bachrach, K. Han, "Logistic regression over encrypted data from fully homomorphic encryption," *BMC medical genomics*, vol. 11, no. 4, pp. 3–12, 2018.

[11] N. Donselaar, "Probabilistic parameterized polynomial time," in *In International Conference on Current Trends in Theory and Practice of Informatics*, pp. 179–191, Novy Smokovec, Slovakia, January 2019.

[12] A. R. Ebrahimi, A. S. Ebrahimi, K. A. Hassani, "A new ring-based sphf and pake protocol on ideal lattices," *the ISC International Journal of Information Security*, vol. 11, no. 1, pp. 75–86, 2019.

Table 7: Performance comparison

Scheme	Security	Key space	Encryption time (s)	Decryption time (s)	Ciphertext expansion
Proposed	IND-CPA	2^{128}	13.0878	4.1264	126
[23]	IND-CPA	-	25.3075	24.2481	6.6667×10^6
[17]	IND-CPA	-	-	-	-
[1]	-	-	2.3005	2.8775	-

[13] S. Halevi, Y. Polyakov, V. Shoup, "An improved rns variant of the bfv homomorphic encryption scheme," in *In Cryptographers' Track at the RSA Conference(CT-RSA). Springer, Cham*, pp. 83–105, San Francisco, CA, USA, March 2019.

[14] Y. Han, S. Li, Y. Cao, "Voice-indistinguishability: Protecting voiceprint in privacy-preserving speech data release," in *In 2020 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, London, United Kingdom, July 2020.

[15] K. Hariss, H. Noura, A. E. Samhat, "An efficient fully homomorphic symmetric encryption algorithm," *Multimedia Tools and Applications*, vol. 79, no. 17, pp. 12139–12164, 2020.

[16] I. Iliashenko, V Zucca, "Faster homomorphic comparison operations for bgv and bfv," *Proceedings on Privacy Enhancing Technologies*, vol. 2021, no. 3, pp. 246–264, 2021.

[17] O. A. Imran, S. F. Yousif, I. S. Hameed, "Implementation of el-gamal algorithm for speech signals encryption and decryption," *Procedia Computer Science*, vol. 167, no. 3, pp. 1028–1037, 2020.

[18] G. P. Kanna, V. Vasudevan, "A fully homomorphic-elliptic curve cryptography based encryption algorithm for ensuring the privacy preservation of the cloud data," *Cluster Computing*, vol. 22, no. 4, pp. 9561–9569, 2019.

[19] A. Kim, Y. Polyakov, V. Zucca, "Revisiting homomorphic encryption schemes for finite fields," *IACR Cryptol. ePrint Arch.*, vol. 2021, pp. 204, 2021.

[20] H. Q. Le, P. K. Mishra, S. Nakamura, "Impact of the modulus switching technique on some attacks against learning problems," *IET Information Security*, vol. 14, no. 3, pp. 286–303, 2020.

[21] S. Meftah, B. H. M. Tan, C. F. Mun, "Doren: Towards efficient deep convolutional neural networks with fully homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3740–3752, 2021.

[22] L. R. Scott, T. Clark, B. Bagheri, "Scientific parallel computing," *Princeton University Press*, 2021.

[23] C. Shi, H. Wang, Y. Hu, "A speech homomorphic encryption scheme with less data expansion in cloud computing," *KSII Transactions on Internet & Information Systems*, vol. 13, no. 5, pp. 2588–2609, 2019.

[24] K. Suresha, P. V. Karthick, "Enhancing data security in cloud computing using threshold cryptography technique," *Advances in cybernetics, cognition, and machine learning for communication technologies*, vol. 643, pp. 158–166, 2020.

[25] C. N. Umadevi, G. Kumaresan, N. P. Gopalan, "Fully homomorphic symmetric key encryption-based access control over out-sourced cloud data using smith normal form," *Journal of Applied Security Research*, vol. 16, no. 2, pp. 247–257, 2021.

[26] A. Wood, K. Najarian, D. Kahrobaei, "Homomorphic encryption for machine learning in medicine and bioinformatics," *ACM Computing Surveys (CSUR)*, vol. 53, no. 4, pp. 1–35, 2020.

[27] H. Xiao, Y. Huang, Y. Ye, "Robustness in chinese remainder theorem for multiple numbers and remainder coding," *IEEE Transactions on Signal Processing*, vol. 66, no. 16, pp. 4347–4361, 2018.

[28] V. Zue, S. Seneff, J. Glass, "Speech database development at mit: Timit and beyond," *Speech communication*, vol. 9, no. 4, pp. 351–356, 1990.

Biography

Qiuyu Zhang Researcher/Ph.D. supervisor, graduated from Gansu University of Technology in 1986, and then worked at school of computer and communication in Lanzhou University of Technology. He is vice dean of Gansu manufacturing information engineering research center, a CCF senior member, a member of IEEE and ACM. His research interests include network and information security, information hiding and steganalysis, multimedia communication technology.

Yujiao Ba is currently a master student of the School of Computer and Communication, Lanzhou University of Technology, China. She received the BS degrees in computer science and technology from Lanzhou University of Technology, Gansu, China, in 2019. Her research interests include network and information security, audio signal processing and application, multimedia data security.

Yingjie Hu is currently a Ph.D student of the School of Computer and Communication, Lanzhou University of Technology, China. She received the master degree in computer software and theory from Lanzhou University, China, in 2011. Her research interests include information security, privacy preserving technology, and audio signal processing and application.

Zhenyu Zhao is currently a master student of the School of Computer and Communication, Lanzhou University of Technology, China. He received the BS degrees in electronic information science and technology

from Science and Technology College of North China Electric Power University, Hebei, China, in 2018. His research interests include audio signal processing and application, multimedia data security.

Yugui Jia is currently a master student of the School of Computer and Communication, Lanzhou University of Technology, China. She received the BS degree from the University of Tarim in 2019. Her research interests include network and information security and privacy preservation technology.