# An ElGamal Encryption with Fuzzy Keyword Search on Cloud Environment

Yilei Wang[1], Wenyi Bao[1], Yang Zhao[1], Hu Xiong[1,2], and Zhiguang Qin[1]
*(Corresponding author: Hu Xiong)*
School of Information and Software Engineering, University of Electronic Science and Technology of China[1]
No. 4, North Jianshe Road, Chenghua District, chengdu, Sichuan 610054, China
State Key Laboratory of Information Security, Institute of Software & Chinese Academy of Sciences[2]
No. 19, Yuquan Road, Shijingshan District, Beijing 100190, China
(Email: xionghu.uestc@gmail.com)
*(Received June 17, 2015; revised and accepted Aug. 12 & Aug. 24, 2015)*

## Abstract

With the continuous development of cloud computing, more and more sensitive data needs to be centrally stored in the cloud storage. For protecting the privacy of data, sensitive data must be encrypted before being outsourced to the server. The traditional PEKS (Public Encryption Keyword Search) enables users to search data by using keywords in the condition of encryption, however, it not only needs the security channel but also tolerates the huge pairing-computation. Although the pairing-free public key encryption with keyword search has been proposed, it can not support fuzzy keyword search and this drawback greatly reduces the usability of the system. In this paper, the proposed scheme features three good aspects: First, the keywords and data have been encrypted under the server's public key and thus the secure channel of the keywords transmission has been eliminated in the sense that the outside attackers cannot obtain any information related to the keywords without the knowledge of the server's private key. Second, our scheme not only supports accurate keyword search encryption but also supports the search when the keywords input have any spelling mistakes or format inconsistencies, which significantly improved the availability of the system. Finally, the proposed scheme is constructed on the El Gamal encryption instead of the bilinear-pairing encryption, which greatly improve the computational efficiency.

*Keywords: Cloud environment, free secure-channel, fuzzy keywords*

## 1 Introduction

With the development of cloud computing, more and more confidential documents will be stored in the cloud environment. But Clients also worry about the trust rank of the server, so the data stored in the cloud server will be encrypted. Clients can download the all data and decrypt it, then they will search what they want by keywords, however, the process will expend a lot of time and cost. So, it's more and more important to propose an effective searchable encryption scheme.

Boneh et al. propose the keyword search scheme under the condition of public key encryption for the first time in [2]. But a secure channel is established between the server and receiver to transmit data. As all we know, the computing overhead of bilinear pairings is very huge. Thokozani et al. propose a pairing-free PEKS scheme in [14] and it greatly reduces the computational cost. However, this scheme only allows exact keyword search, that is to say, the searching keywords can not tolerate any incorrect spellings and formats. The obvious drawbacks seriously reduces the availability of it.

In this paper, we propose a pairing-free public key encryption with fuzzy keyword search scheme, it does not need to establish a secure channel between the server and receiver. For creating a PEKS ciphertext, the sender will use the server's public key and his own public key. Meanwhile the scheme also supports fuzzy keyword search and don't need to encrypt keywords by bilinear pairings needing much computing cost. When the keyword input by users exactly match the defined keyword, the server will return the related files containing it directly. When the exact match fails, according to the similar semantics of keywords, the server will return the most likely similar matching files. More accurately, this paper will use the similar semantics in [8] and specifically use edit distance to quantify the similarity of the keywords. We will use the wildcard technology to solve the problem of the creation of fuzzy keyword sets. There is no need to list all keywords, and the number of fuzzy keyword sets greatly decrease by utilizing wildcard technology. Compared with the previous schemes, ours meets the three requirements:

1) Secure channel-free;

2) Pairing-free;

3) Fuzzy keyword search.

### 1.1 Related Work

The earliest PEKS scheme was proposed in [2], in which the user can send a secret key to the server, the server can identify all of the data files which contain keywords searched but he can't get any information about the data files. An secure channel-free PEKS scheme was proposed by Baek in [1], the basic idea of it is that the server has its own public key and private key, the data owner creates a PEKS ciphertext by using the server's public key and their own, then the receiver can send a trapdoor to retrieve files through a public channel because the outside attacker that has not obtained the server's private key cannot make any decisions about the PEKS ciphertexts even though the attacker gets all the trapdoors for the keywords that it holds. Fuzzy keyword search over encrypted was first proposed in [8], the scheme use edit distance to quantify the similarity of the keyword and use wildcard technology to create fuzzy keyword sets, the server can return the IDs of files by matching the index of similar keywords, but, its trapdoor is unsafe and vulnerable to keyword guessing attack. Most PEKS proposed so far are based on the bilinear pairings. Thokozani et al. proposed a pairing-free PEKS in [14] and it improved computational efficiency greatly, but this scheme can't support fuzzy keyword search.

Rhee et al. point out that the SCF-PEKS scheme suffers from keyword guessing attack and proposed a scheme which satisfies the property of trapdoor indistinguishability without using an additional secure channel in [11]. For achieving a more effective search, a similar "index" technology was proposed in [9], in which a single index of encrypted hash table is established for the whole file storage. In the index table, each item is made up of the trapdoor of keywords and the encrypted collection of identify numbers of the files which contain relevant keywords. Both of the two schemes only support exact keyword search. Min-Shiang Hwang et al. propose a Study of PEKS in [4] which is a summary of PEKSs and show an overview of six existing security models of PEKS/SCF-PEKS scheme and conclude five security requirements that must satisfy to construct a secure PEKS/SCF-PEKS scheme.

A scheme supports secure keyword ranking was proposed in [15] and returned the ranking of searching files through an effective technology, which enhances the usability of searching system. In [10], an efficient PEKS scheme was proposed, which allows the server to participate in the decipherment, and to return only files containing certain keywords specified by the users, so as to reduce both the computational and communication overhead in decryption for users, on the condition of preserving user data privacy and user querying privacy. Shih-Ting Hsu et al. propose a study of CKSS in [5] and examine six security models by concluding the secret-key setting and public-key setting, and sum up six security requirements that must satisfy to construct a secure conjunctive keyword searchable scheme.

### 1.2 The Advantages of Our Scheme

According to [1, 2, 8], most PEKSs schemes cannot support the public key encryption with fuzzy keyword search. For example, after sending the encrypted messages along with the corresponding keywords into the cloud server, the data owner cannot perform the keyword searching in case the exact passwords has been forgotten, which usually happens when many files has been outsourced in the remote cloud server.

In the addition, most PEKSs need to encrypt keyword ciphertext by bilinear pairings needing much computing cost which will reduce the efficiency of schemes. Although Thokozani et al. proposed a pairing-free PEKS in [14] and it improved computational efficiency greatly. However, there are some obvious drawbacks in the scheme. First the scheme can't decrypt the ciphertext to get the correct messages because its decryption algorithm has some mistakes. Second, it uses the server's public key to encrypt the trapdoor instead of the keyword ciphertext, so, it needs a secure channel to transport the trapdoor.

Different from previous schemes, our proposed scheme provides a promising solution to this problem by supporting public key encryption with keywords search. In this way, only part of keywords or the keywords with some spelling errors can be used to perform the keyword search. Furthermore, our scheme is constructed on the ElGamal instead of bilinear pairings to encrypt keywords, which significantly improve the efficiency.

### 1.3 Organization

The organization of this paper is as follows. Some preliminaries are given in Section 2. The proposed ElGamal encryption with fuzzy keyword search on cloud environment are given in Section 3. The comparison of efficiency is given in Section 4. Its security analyse is given in Section 5. The conclusions will be made in Section 6.

## 2 Preliminaries

In this section, we will review the traditional PEKS, the creating method of fuzzy keyword sets and the definition of some relevant knowledge.

### 2.1 The Sets of Fuzzy Keyword

For proposing an effective and practical fuzzy keyword search scheme, the concept of edit distance is introduced into the solution. If the editing operation is in the same position of a keyword, all relevant keywords will be listed. Using wildcards represents the same position of editing operation in [8].

**Edit Distance.** There are some methods to quantize the similarity of strings. A known edit distance is proposed in [6], $ed(w_1, w_2)$(denoting the edit distance of $w_1$ and $w_2$) means the number of operations which one keyword becoming another similar keyword needs.

The editing operation is made up of three parts:

1) Substitution: changing one character to another in a word;

2) Deletion: deleting one character from a word;

3) Insertion: inserting a single character into a word. Given a keyword $w$.

The wildcard-based fuzzy keyword set of $w_i$ with edit distance d is denoted as $S_{w_i,d} = \{S'_{w_i,0}, S'_{w_i,1}, ..., S'_{w_i,d}\}$, where $S'_{w_i,\tau}$ denotes the set of words $w'_i$ with $\tau$ wildcards. For example,assuming that edit distance $d = 1$ for the keyword "while", its wildcard-based fuzzy keyword set can denote $S_{while,1} = \{while, *while, *hile, w*ile, \cdots, whil*, while*\}$.

**Fuzzy Keyword Search.** Given a set of n encrypted data files $C = (F_1, F_2, ..., F_N)$ stored in the cloud server, a set of different keywords $W = w_1, w_2, ..., w_p$, given the edit distance d, a search input $(w, k)$ ($w$ denotes a keyword, $k(k \leqslant d)$ denotes the input of the edit distance). The server will return $IDs$ of files after the execution of fuzzy keyword search and $IDs$ whose corresponding the data files may contain the keyword $w$, denoting $FID_{w_i}$: if $w = w_i \in W$, return $FID_{w_i}$ immediately; Or if $w \notin W$, return the IDs' set $FID_{w_i}$ for $ed(w, w_i \leqslant k)$.

### 2.2 The First PEKS Scheme

As being described in [2], this search system is made up of a data owner, a data receiver, a server. The Data owner creates some data and then send the encrypted data and keywords to the server. When the server receives them, he can execute the search operating by obtaining the trapdoor from the data receiver. The Data receiver creates the trapdoor and send it to the server to search what he wants.

We review 4 steps of the public-key encryption search (PEKS) algorithm:

1) $KeyGen(s)$: Taking secure parameter $s$, then the algorithm generates the common public key and private key $(pk, sk)$ of data owner and data receiver.

2) $PEKS(pk, W)$: Taking $pk$ and a keyword $W$, the algorithm generates a PEKS ciphertext $S = PEKS(pk, W)$.

3) $Trapdoor(sk, W')$: Taking $sk$ and a keyword $W'$ input, the algorithm generates a search trapdoor $T_{w'}$.

4) $Test(pk, S, T_{w'})$: Taking the public key $pk$, the PEKS ciphertext $S$ and the search trapdoor $T_{w'} = trapdoor(sk, W')$, the algorithm matches if $W = W'$, output "YES", otherwise output "NO".

Data owner executes $KeyGen$ algorithm to generate public-private key pairs. Then data receiver uses the algorithm $Trapdoor$ to generate the trapdoor $T_{W'}$ for a keyword $W'$ input by him. After the server receives the trapdoor, he will execute $Test$ algorithm to determine whether these data files contain the keyword $W'$.

### 2.3 ElGamal Encryption

ElGamal encryption algorithm is a relatively common encryption algorithm and it is based on public key cryptosystem and elliptic curve encryption system which are proposed in 1984. Its security depends on elliptic curve discrete logarithm problem over the finite fields. The algorithm description is in [13] as follows:

*KeyGen*: First select a prime $p$ and obtain primitive root $g$, a random element $x \in F_q$ in which both $x$ and $g$ are less than $p$, then computes $y = g^x$. The user's public key is $y$ and private key is $x$.

*Encryption*: For encrypting a file $F$, pick a random element $r \in F_q$ and computes $c_1 = g^r, c_2 = F(g^x)^r$, then obtain the ciphertext $c_F = (c_1, c_2)$.

*Decryption*: Given $c_F = (c_1, c_2)$, recover the file by computing $c_2/c_1^x = F$.

### 2.4 Discrete Logarithm Problem

Discrete Logarithm Problem: Given a prime number $p$ and a primitive element $a \in \mathbb{Z}_p(\mathbb{Z}_p$ is a finite field),for a integer $b \in \mathbb{Z}_\text{ı}$, finding the unique integer $c$ make $a^c \equiv b(modp)$ is a difficult problem if selecting $p$ carefully.

At present, there is not a polynomial time algorithm of computing discrete logarithm problem.

## 3 Construction

In the section, we will propose our scheme which not only support fuzzy keyword search but also encrypt keywords and data files by ElGamal instead of bilinear pairings, and needn't an additional secure channel to exchange a trapdoor.

Some public system parameters will be generated by first algorithm such as server's public key and private key, the sender's public key and private key.

1) $KeyGen(\gamma_1, \gamma_2)$: Taking the security parameters $\gamma_1$ and $\gamma_2$, this algorithm will generate public key $y = g^x$, private key $x$ for the sender and public key $S = g^z$, private key $z$ for the server.

   We suppose the value of edit distance is $d$, and suppose the keyword set $\{w_1, w_2, w_3, ..., w_i\}$ of every encrypted file in the cloud server. For setting up an index for each keyword $w_i$, first the sender will create a fuzzy keyword set of index $S_{w_i,d}$ ($S_{w_i,d}$ has been explained in Section 2). each element of $S_{w_i,d}$

is the keyword which uses wildcard technology. Every wildcard denotes an editing operation. Then the sender will encrypt each $w_i' \in S_{w_i,d}$. The description of PEKS algorithm is as follows.

2) $PEKS(y, S, w_i')$: To encrypt the keyword $w_i'$, this algorithm first computes $c_i = h_1(w_i')$ and $C_{w_i'} = S \cdot y^{c_i}$ by using one-way hash function. Then the sender sends $C_{w_i'}$ to the server for storage.

   The data user needs to input $(w, k)$ ($(w, k)$ has been explained in Section 2) for searching files, and he will computes all trapdoors $\{T_{w'}\}_{w' \in S_{w,k}}$. the algorithm's description of computing every fuzzy keyword trapdoor based on wildcard-keyword is as follows.

3) $Trapdoor(x, w')$: To retrieve the data files which the user wants, the user chooses a dynamic random element $a$ and computes two trapdoors $T'_{w'} = (g^{h_1(w')})^x \cdot g^a$ and $T''_{w'} = g^a$.

   The algorithm generates the trapdoor $T_{w'} = (T'_{w'}, T''_{w'})$, and when the user computes all fuzzy keyword trapdoors, he sends the trapdoor set $\{T_{w'}\}_{w' \in S_{w,k}}$ to the server for returning what he wants. the description of matching algorithm is as follows.

4) $Test(T'_w, z, C_{w_i'})$: First the server uses its private key to compute $C_w = (C_{w_i'}) \cdot g^{-z} = y^{c_i}$, then test whether $T'_{w'} = T''_{w'} \cdot C_w$. If $w_i' = w'$, the algorithm outputs "YES", then return the corresponding data files or else if $w_i' \neq w'$, the algorithm outputs "NO".

Our proposed scheme needn't a secure channel to transport the trapdoor because of the server's public key $S = g^z$ to encrypt the PEKS $C_{w_i'} = S \cdot y^{c_i}$, and can resist keyword guessing attack. By using El Gamal encryption instead of bilinear pairings encryption and supporting fuzzy keyword search, it improves the efficiency and usability significantly.

# 4 Comparison

In the section, the efficiency and usability analyse of our scheme and the others are compared as follows:

We compare our approach with Li et al. in [10], Baek et al. in [1], Rhee et al. in [11], Boneh et al. in [2], Thokozani et al. in [14] in term of the computation cost. To achieve the similar level of security for our pairing-free approach, the Koblitz elliptic curve $y^2 = x^3 + ax^2 + b$ can be used. The running time of the cryptographic operation listed in Table 1 can be derived using the standard cryptographic library. MIRACAL [12], and the hardware and OS for the experiment is PIV 3 GHZ processor with 512 M bytes storage capacity, and the Windows XP operating system respectively [3]. Bilinear pairings is the most expensive computation operation while scalar multiplication is the next and modular exponentiation is the third, hash function is the least.

Table 1: Cryptographic operation time in milliseconds

| Operations | Time |
|---|---|
| ECC-based scalar multiplication | 0.83 |
| Exponential in $\mathbb{F}_{p^2}$ | 11.20 |
| Pairing-based scalar multiplication | 6.38 |
| Pairing | 20.01 |

Table 2: Frequency of each operation

| Scheme | PA | SM | EX | HF |
|---|---|---|---|---|
| Li et al. in [10] | 3 | - | 5 | 3 |
| Baek et al. in [1] | 2 | 2 | 1 | 1 |
| Rhee et al. in [11] | 3 | 3 | 3 | 3 |
| Boneh et al. in [2] | 1 | 3 | 1 | 3 |
| Thokozani et al. in [14] | - | 3 | 6 | 1 |
| Our scheme | - | 4 | 5 | 1 |

The number of keywords affects the space efficiency of the existing approaches and ours at the most extent. For example, we implement the experiment that there are 5000 keywords needed to be stored , a keyword takes up two bytes in average and each keyword has 50 corresponding fuzzy keywords. So all of keywords will take $5000 \times 2 \times 50 = 500000B \approx 488KB$. We can know that our scheme needs a little space to save keywords and rarely reduces the space efficiency.

Next,we can analyse the computation cost of our scheme that there are 4 scalar multiplications($S \cdot y^{c_i}$, $(g^{h_1(w')})^x \cdot g^a$, $(C_{w_i'}) \cdot g^{-z}$, $T''_{w'} \cdot C_w$), 5 exponentiations($y^{c_i}$, $g^{h_1(w')}$, $(g^{h_1(w')})^x$, $g^a$, $g^{-z}$), 1 hash function($h_1(w_i')$). Also, we can analyse the computational overhead of the other five approaches in the Table 2.

In the Table 2, Let PA, SM, EX and HF be the abbreviate for the Pairing, Scalar Multiplication, Exponentiation and Hash Function, respectively. The comparison focuses on the operation implemented by the sender or the receiver with the server to have sufficient communication capability.

Thus, the computation efficiency is evaluated based on the method proposed in [7]. For example, in the algorithm of Baek et al.'s in [4], 2 PAs , 2 SMs , 1 EX, 1 HF are needed, and the computation time is $2 \times 6.38 + 2 \times 20.01 + 1 \times 11.20 = 63.98$ms(the computation cost of hash function can be ignored.). In the same way, we can calculate the other four existing schemes and ours($59.32ms$). Observing the comparison results listed in Chart 1, although, the computation cost of scheme in [2] is less than ours, it needs a extra secure channel to transport its trapdoor which reduce its efficiency. From the above, our scheme is more efficient than the existing schemes for implementing each exact keyword search.

When a user implements fuzzy keyword search, we denote the size of a fuzzy keyword set by n, n is a constant and he/she can compute n by some searching software quickly and cost $m = 59.32n$ ms for searching, so $m$ is

Table 3: Comparison of usability assumption

| Scheme | CT Ind | Trap Ind | SC | FKS |
|---|---|---|---|---|
| Li et al. in [10] | Satisfied | Not satisfied | Required | unsupported |
| Baek et al. in [1] | Satisfied | Not satisfied | Not Required | unsupported |
| Rhee et al. in [11] | Satisfied | Satisfied | Not Required | unsupported |
| Boneh et al. in [2] | Satisfied | Not satisfied | Required | unsupported |
| Thokozani et al. in [14] | Satisfied | Satisfied | Not Required | unsupported |
| Our scheme | Satisfied | Satisfied | Not Required | supported |

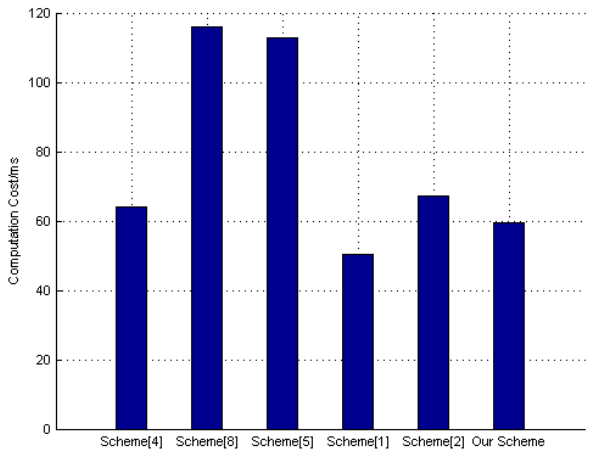


Figure 1: Comparison of the Existing Schemes on Computation Cost

also a constant and can be accepted.

In Table 3, we use CT Ind, Trap Ind, SC, FKS as abbreviations for the meaning of PEKS Ciphertext Indistinguishability, Trapdoor Indistinguishability, Secure Channel between a sender and server, and Fuzzy Keyword Search,respectively.

Our proposed scheme uses the lighter operation of exponentiation because the security bases of it are on the intractability of elliptic curve discrete logarithm problem and ElGamal encryption.

# 5 Security Analyse

The security of our proposed scheme will be discussed in the section.

As all we know, the first PEKS scheme in [2] is a non-interactive searchable encryption scheme semantically secure against a chosen keyword attack in the random oracle model. But in [2], a certain keyword corresponds to a constant trapdoor. Therefore, an outside attacker who intercepts and captures the communications can statistics the frequency of occurrences of these trapdoors, and then he can choose the highest frequency trapdoor to attack. Once the attack is successful, an attacker may know users privacy interests. So it is not against a keyword guessing attack.

**Theorem 1.** *Our pairing-free scheme satisfies the property of trapdoor indistinguishability which can be against a keyword guessing attack in the random oracle.*

*Proof.* As a malicious server or an outside attacker, he can not distinguish whether two trapdoors are from the same keyword. the trapdoor is changed each time because of the difference of the random element $a$ we chooses. If the $a$ is changed, $T'_{w'} = (g^{h_1(w')})^x \cdot g^a$ and $T''_{w'} = g^a$ will be also changed. So the trapdoor can not be distinguished. □

**Theorem 2.** *Our pairing-free scheme satisfies the property of keyword security in the random oracle.*

*Proof.* Even though an outsider attacker or a malicious server knows that two trapdoors are generated by the same keyword and intercept them, they can't do anything about the trapdoors. Suppose an outsider attacker captures the sender's private key $x$ , the trapdoor $T_{w'}$ and $g^a$., then he will compute $g^{h_1(w')}$ through $T'_{w'}$ captured. Due to elliptic curve discrete logarithm problem, he can't compute $g^{h_1(w')}$. So he can't obtain any information about the trapdoor. □

# 6 Conclusion

In this paper, we propose an ElGamal encryption with fuzzy keyword search scheme on cloud environment. It not only supports accurate keyword search encryption but also supports the search when the keywords which are input have any spelling mistakes or format inconsistent situations. For bilinear pairings free, our scheme is more efficient than most others, and it can satisfies the property of trapdoor indistinguishability and keyword security in the random oracle based on the intractability of elliptic curve discrete logarithm problem and ElGamal encryption. Finally it is also a SCF-PEKS. How to reduce the size of a fuzzy keyword set effectively is our future master expectation.

# Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61003230,

Grant 61370026, and Grant 61202445, in part by the Fundamental Research Funds for the Central Universities under Grant ZYGX2013J073, in part by the Applied Basic Research Program of Sichuan Province under Grant 2014JY0041, in part by the National High Technology Research and Development Program of China (863) under Grant 2015AA016007 and in part by the science and technology foundation of Sichuan Province under Grant 2014GZ0109.

## References

[1] J. Baek, R. Safavi-Naini, W. Susilo, "Public key encryption with keyword search revisited," in *The 8th International Conference of Computational Science and Its Applications (ICCSA'08)*, pp. 1249–1259, Perugia, Italy, 2008.

[2] D. Boneh, G. Di Crescenzo, R. Ostrovskyl, "Public key encryption with keyword search," in *The 22th International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'04)*, pp. 506–522, Heidelberg, Berlin, 2004.

[3] X. Cao, W. Kou, X. Du, "A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges," *Information Sciences*, vol. 180, no. 15, pp. 2895–2903, 2010.

[4] S. T. Hsu, C. C. Yang, M. S. Hwang, "A study of public key encryption with keyword search," *International Journal of Network Security*, vol. 2, no. 15, pp. 71–79, 2013.

[5] C. C. Lee, S. T. Hsu, M. S. Hwang, "A study of conjunctive keyword searchable schemes," *International Journal of Network Security*, vol. 5, no. 15, pp. 321–330, 2013.

[6] V. Levenstein, "Binary codes capable of correcting spurious insertions and deletions of ones," *Problems of Information Transmission*, vol. 1, no. 1, pp. 8–17, 1965.

[7] J. Li, H. Du, Y. Zhang, T. Li, "Provably secure certificate-based key-insulated signature scheme," *Concurrency and Computation Practice and Experience*, vol. 26, no. 8, pp. 1546–1560, 2014.

[8] J. Li, Q. Wang, and C. Wang, "Fuzzy keyword search over encrypted data in cloud computing," in *The 29th Conference on Computer Communications (INFOCOM'10)*, pp. 1–5, San Diego, USA, Mar. 2010.

[9] M. Li, S. Yu, K. Ren, and W. Lou, "Searchable symmetric encryption: improved definitions and efficient constructions," in *Proceedings of the 13th ACM conference on Computer and communications security (ACM'06)*, pp. 79–88, New York, NY, USA, 2006.

[10] Q. Liu, G. Wang, J. Wu, "Secure and privacy preserving keyword searching for cloud storage services," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 927–933, 2012.

[11] H. S. Rhee, J. H. Park, W. Susilo, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," *Journal of Systems and Software*, vol. 83, no. 5, pp. 763–771, 2010.

[12] M. Scott, *Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL)*, CertiVox Ltd, Aug. 2015. (`https://www.certivox.com/miracl`)

[13] Y. Tsiounis, M. Yung, "On the security of elgamal based encryption," in *The First International Workshop on Practice and Theory in Public Key Cryptography (PKC'98)*, pp. 117–134, Pacifico Yokohama, 1998.

[14] T. F. Vallent and H. Kim, "A pairing-free public key encryption with keyword searching for cloud storage services," in *The 5th International Conference of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (AFRICOMM'13)*, pp. 70–78, Kampala, Uganda, 2014.

[15] C. Wang, N. Cao, J. Li, "Secure ranked keyword search over encrypted cloud data," in *The IEEE 30th International Conference on Distributed Computing Systems (ICDCS'10)*, pp. 253–262, Genoa, Italy, 2010.

**Yilei Wang** is pursuing his Ph.D. degree in the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC). He received his B.S. degree from School of Communication and Information, UESTC, in 2008. He received his M.S. from Faculty of Engineering, Lund University of Sweden,in 2011. His research interests include security and application of mobile network data.

**Wenyi Bao** received his B.S. degree in the China University of Mining and Technology (CUMT) in 2009. He is currently pursing his M.S. degree in the School of Information and Software Engineering, UESTC. His research interests include: public key encryption with (fuzzy) keyword search.

**Yang Zhao** is an associate professor at the School of Computer Science and Engineering, UESTC. His research interests are in the area of networking security and e-commerce protocol.

**Hu Xiong** is an associate professor at the School of Information and Software Engineering, UESTC. He received his Ph.D degree from UESTC in 2009. His research interests include: cryptography and network security.

**Zhiguang Qin** is the dean and professor in the School of Information and Software Engineering, UESTC. He received his PH.D. degree from UESTC in 1996. His research interests include: information security and computer network.