# Public Key Encryption with Keyword Search from Lattices

Chunxiang Gu, Yan Guang, Yuefei Zhu, Yonghui Zheng
Zhengzhou Information Science and Technology Institute
P.O. Box 1001-770, Zhengzhou, 450002, P.R.China

**Abstract.** Lattice-based cryptography had recently acquired much attention due to many potential advantages: their resistance so far to cryptanalysis by quantum algorithms, their asymptotic efficiency and conceptual simplicity, and the guarantee that their random instances are as hard as the hardness of lattice problems in worst case. In this paper, we propose a public key encryption with keyword search (PEKS) using lattices. PEKS is a mechanism for searching on encrypted data. It enables user Alice to send a secret value $T_w$ to a server that will enable the server to locate all encrypted messages containing the keyword $w$, but learn nothing else. The scheme can be proven secure with the hardness of the standard Learning With Errors (LWE) problem in the random oracle model.

**Keywords**: lattice-based cryptography, public key encryption with keyword search, provable secure.

## 1. Introduction

Recently, lattices have emerged as a powerful mathematical platform on which to build a rich variety of cryptographic primitives. Starting from the work of Ajtai[1], many new constructions with lattices have been proposed, such as one-way functions and hash functions[1,2], trapdoor functions[3], public-key encryption schemes[4], ID-based encryption schemes[5-7], fully homomorphic encryption schemes[8,9], and so on. Lattice-based schemes are attractive due to many potential advantages: their asymptotic efficiency and conceptual simplicity; their resistance so far to cryptanalysis by quantum algorithms (as opposed to schemes based on factoring or discrete log); and the guarantee that their random instances are as hard as the hardness of lattice problems in worst case. X. Boyen [10] said that "*all those factors conspire to make lattices a prime choice, if not the primary one yet, for mathematical crypto design looking out into the future*."

Keyword-based search technique allows users to selectively retrieve files of interest and has been widely applied in plaintext search scenarios, such as Google search. However, for privacy, data encryption may also demands the protection of keyword since keywords usually contain important information related to the data files. This leads to the traditional plaintext search techniques useless in this scenario. To securely search over encrypted data, some searchable encryption techniques have been developed [11-18].

*Public-key encryption with keyword search* (PEKS) was first proposed by Boneh et.al [12] in 2004. The mechanism enables one to search encrypted keywords without compromising the security of the original data. Suppose Bob wants to send Alice a message $m$ with keywords $w_1,...w_n$. Let $pk_A$ be Alice's public key. Bob encrypts

$m$ using a standard public key encryption $E(.)$ and appends to the resulting ciphertext a list of PEKS ciphertext of each keyword. That is $E(M, pk_A) \| PEKS(w_1, pk_A) \| ... \| PEKS(w_n, pk_A)$. This kind of encrypted messages may be stored in a server. Alice can give the server a certain trapdoor $T_w$ through a secure channel that enables the server to test whether one of the keywords associated with the message is equal to the word $w$ of Alice's choice. Given $PEKS(w', pk_A)$ and $T_w$, the server can test whether $w = w'$. If $w \neq w'$ the server learns nothing more about $w'$. PEKS mechanism can be widely used in many practical applications, such as encrypted emails extraction[12], encrypted and searchable audit logs[13], and encrypted files extraction in Cloud[16].

We should note that the proposed PEKS schemes are mainly using pairings[12,17] and some traditional cryptography tools (such as RSA) [11]. In this paper, we propose a public-key encryption with keyword search scheme using lattices. The basic construction only gets probabilistic consistency, so we further propose a method to make up the shortfall. The scheme can be proven secure with the hardness of the standard Learning With Errors (LWE) problem in the random oracle model.

The rest of this paper is organized as follows: In Section 2, we recall some preliminary works. In Section 3 and 4, we describe the details of our new schemes and their security proofs. Finally, we conclude in Section 5.

## 2. Preliminaries

### 2.1 Public-key Encryption with Keyword Search

Recall that a PEKS scheme consists of four polynomial-time algorithms[12]: *KeyGen, PEKS, Trapdoor*, and *Test*. The *KeyGen* algorithm generates a public/private key pair $(pk, sk)$. The *PEKS* algorithm produces a searchable encryption of keyword $w$ with receiver's public key. The *Trapdoor* algorithm generates a trapdoor $T_w$ for keyword $w$ with receiver's private key, and the *Test* algorithm verifies whether a cipher-text matches a trapdoor.

The general notion of security of PEKS scheme is indistinguishability against chosen keyword attack [12]. A PEKS scheme is semantic security if no polynomial time adversary A has a non-negligible advantage against a challenger C in the following game.

**Security Game:**

**KeyGen.** C runs *KeyGen* algorithm to generate a key pair $(pk, sk)$, and gives $pk$ to A .

**Phase 1.** A can adaptively ask the challenger for the trapdoor $T_w$ for any keyword $w \in \{0,1\}^*$ of his choice.

**Challenge.** At some point, A sends the challenger two words $w_0, w_1$ on which it wishes to be challenged. The only restriction is that A did not previously ask for the

trapdoors $T_{w_0}$ or $T_{w_1}$. The challenger picks a random $b \in \{0,1\}$ and gives the attacker $C = PEKS(pk, w_b)$ as the challenge PEKS cipher-text.

**Phase 2.** A can continue to adaptively ask the challenger for the trapdoor $T_w$ for any keyword $w$ of his choice as long as $w \neq w_0, w_1$.

**Guess.** Eventually, the attacker A outputs $b' \in \{0,1\}$ and wins the game if $b = b'$.

The advantage of A in this game is defined as $|\Pr[b' = b] - \frac{1}{2}|$.

### 2.2 Lattices and hardness assumption

**Definition 1**. Given $n$ linearly independent vectors $b_1, b_2, ..., b_n \in R^m$, the lattice $\Lambda$ generated by them is denoted $L(b_1, b_2, ..., b_n)$ and define as:

$$L(b_1, b_2, ..., b_n) = \{\sum_{i=1}^{m} x_i b_i \mid x_i \in Z\}$$

The vectors $b_1, b_2, ..., b_n$ are called the basis of the lattice. Let $B = \{b_1, b_2, ..., b_n\}$, we let $\tilde{B}$ denote its Gram-Scahmidt orthogonalization of the vectors $b_1, b_2, ..., b_n$ taken in that order, and we let $\| S \|$ denote the length of the longest vector in $B$ for the Euclidean norm.

**Definition 2**. For $q$ prime and $A \in Z_q^{n \times m}$ and $u \in Z_q^n$, define:

$$\Lambda_q^{\perp}(A) = \{e \in Z^m \mid A \cdot e = 0 \bmod q\}$$

$$\Lambda_q^{u}(A) = \{e \in Z^m \mid A \cdot e = u \bmod q\}$$

Ajtai [1] and later Alwen and Peikert[19] showed how to sample an essentially uniform matrix $A \in Z_q^{n \times m}$ with an associated basis $T_A$ of $\Lambda_q^{\perp}(A)$ with low Gram-Scahmidt norm.

**Proposition 1**[20]. For any prime $q \geq 2$ and $m \geq 5n \log q$, there exists a probabilistic polynomial-time algorithm ***TrapGen*** that outputs a pair $A \in Z_q^{n \times m}$ and $T_A$ such that $A$ is statistically close to uniform and $T_A$ is a basis for $\Lambda_q^{\perp}(A)$ with length $L = \| \tilde{T}_A \| \leq m\omega(\sqrt{\log m})$ with all but $n^{-\omega(1)}$ probability.

We further review Gaussian functions used in lattice based cryptographic constructions.

**Definition 3**. Let $m$ be a positive integer and $\Lambda \in R^m$ be a $m$ dimensional lattice. For any vector $c \in R^m$ and any positive parameter $\sigma \in R_{>0}$, we define:

$$\rho_{\sigma,c}(x) = \exp(-\pi \| x - c \|^2 / \sigma^2) \quad \text{and} \quad \rho_{\sigma,c}(\Lambda) = \sum_{x \in \Lambda} \rho_{\sigma,c}(x)$$

The discrete Gaussian distribution over $\Lambda$ with center $c$ and parameter $\sigma$ is

$$\forall y \in \Lambda \quad D_{\Lambda,\sigma,c}(y) = \frac{\rho_{\sigma,c}(y)}{\rho_{\sigma,c}(\Lambda)}$$

For notational convenience, $\rho_{\Lambda,\sigma,0}$ and $D_{\Lambda,\sigma,0}$ are abbreviated as $\rho_\sigma$ and $D_{\Lambda,\sigma}$.

Gentry *et.al.* [3] construct the following algorithm for sampling from the discrete Gaussian $D_{\Lambda,\sigma,c}$, given a basis $T_A$ for the *m*-dimensional lattice $\Lambda$ with $\sigma \geq \| \tilde{T}_A \| \omega(\sqrt{\log m})$. Specialized to the case of random lattices, they show an algorithm:

***SamplePre***$(A, T_A, u, \sigma)$: On input a matrix $A \in Z_q^{n \times m}$ with 'short' trapdoor basis $T_A$ for $\Lambda_q^{\perp}(A)$, a target image $u \in Z_q^n$ and a Gaussian parameter $\sigma \geq \| \tilde{T}_A \| \omega(\sqrt{\log m})$, outputs a sample $e \in Z_q^m$ from a distribution that is within negligible statistical distance of $D_{\Lambda_q^u(A),\sigma}$.

Security of all our construction reduces to the LWE problem, a classic hard problem on lattices defined by Regev[4].

**Definition 4**. Consider a prime $q$, a positive integer $n$, and a distribution $\chi$ over $Z_q$, all public. An $(Z_q, n, \chi)$-LWE problem instance consists of access to an unspecified challenge oracle $\Psi$, being either, a noisy pseudo-random sampler $\Psi_s$ carrying some constant random secret key $s \in Z_q^n$, or, a truly random sampler $\Psi_\phi$, whose behaviors are respectively as follows:

-- $\Psi_s$ : output noisy pseudo-random sample of the form $(u_i, v_i) = (u_i, u_i^T \cdot s + x_i) \in Z_q^n \times Z_q$, where $s \in Z_q^n$ is a uniformly distributed persistent value invariant across invocations, $x_i \in Z_q$ is a fresh sample from $\chi$, and $u_i$ is uniform in $Z_q^n$.

-- $\Psi_\phi$: output truly uniform random sample from $Z_q^n \times Z_q$.

The $(Z_q, n, \chi)$-LWE problem allows repeated queries to the challenge oracle $\Psi$. We say that an algorithm A decides the $(Z_q, n, \chi)$ -LWE problem if $|\Pr[A^{\Psi_s} = 1] - \Pr[A^{\Psi_\phi} = 1]|$ is non-negligible for a random $s \in Z_q^n$.

Regev has showed that for certain noise distributions $\chi$, the LWE problem is as hard as the worst-case SIVP and GapsSVP under a quantum reduction.

**Proposition 2**[4]. For an $\alpha \in (0,1)$ and a prime $q > 2\sqrt{n}/\alpha$, let $\overline{\Psi}_\alpha$ denote the distribution over $Z_q$ of the random variable $\lfloor qX + \frac{1}{2} \rfloor \bmod q$ where $X$ is a normal random variable with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$. Then, if there exists an efficient, possibly quantum, algorithm for deciding the $(Z_q, n, \chi)$-LWE problem, there exists a quantum poly-time algorithm for approximating the SIVP and GapSVP problems, to within $\tilde{O}(n/\alpha)$ factors in the $l_2$ norm, in the worst case.

## 3. A PEKS Scheme from Lattice

### 3.1 The Basic Construction

Let $n, m$ be positive integers, $q$ be prime, with $q \geq 2$ and $m \geq 5n \log q$, $H: \{0,1\}^* \to Z_q^n$ be a hash function. The scheme is described as follows:

- $KeyGen(1^n)$**:** On input a security parameter $n$, invoke $TrapGen(q,n)$ to generate a uniformly random matrix $A \in Z_q^{n \times m}$ along with a short basis $T_A$ for $\Lambda_q^\perp(A)$, and output the public key $pk = A$ and secret key $sk = T_A$.
- $PEKS(A, w)$**:** Compute $u = H(w)$, select $s \leftarrow Z_q^n$ uniformly and compute $p = A^T \cdot s + x \in Z_q^m$ and $c = u^T \cdot s + y \in Z_q$, where $x \leftarrow \chi^m$ and $y \leftarrow \chi$ are noise vectors. Output the cipher-text $(p,c)$.
- $Trapdoor(A, T_A, w)$: Compute $u = H(w)$ and choose a $e \in Z_q^m$ satisfying $A \cdot e = u$ by $e \leftarrow Sample\Pr e(A, T_A, u, \sigma)$, where $\sigma$ is a Gaussian parameter. Output $e$ as the trapdoor.
- $Test((p,c), e)$: Compute $b = c - e^T \cdot p$. If $|b| < q/4$, output 1 ("yes"); otherwise, output 0 ("no").

Suppose $(p,c) = PEKS(A, w)$ and $e = Trapdoor(A, T_A, w')$, then

$$b = c - e^T p = u^T \cdot s + y - e^T (A^T \cdot s + x) = (u^T - (A \cdot e)^T) \cdot s + y - e^T \cdot x$$

It is easy to see that if $w = w'$, then $|b| = |y - e^T \cdot x|$, just as discussion in [3], with parameters $q \geq 5rm$, $\alpha \leq 1/(r\sqrt{m} \cdot \omega(\sqrt{\log n}))$, and $\chi = \overline{\Psi}_\alpha$, with overwhelming probability we have $|b| < q/4$, where $r$ is a Gaussian parameter.

However, if $w \neq w'$, with the randomness of $u$ and $e$, $u^T - (A \cdot e)^T$ can be seen as a random vector in $Z_q^n$. That is $|b|$ is a random in $[0, q]$ (over the random choices of $u$ and $e$). Hence, the probability of $|b| < q/4$ is $\frac{1}{4}$. This means that there is probability $\frac{1}{4}$ that the *Test* algorithm output 1 when $w \neq w'$, and so the consistency of the scheme can not be ensured. We will deal with this problem in the following subsection.

**3.2 Improvement for consistency**

In order to decrease the possibility that the *Test* algorithm output 1 when $w \neq w'$, we can modified the basic construction as follows.

Let $k = poly(n)$, for $i = 1,...,k$, $H_i : \{0,1\}^* \to Z_q^n$ be $k$ different hash functions. The *KeyGen* algorithm is the same as before.

- $PEKS(A, w)$**:** Select $s \leftarrow Z_q^n$ uniformly and compute $p = A^T \cdot s + x \in Z_q^m$. For $i = 1,...,k$ ,compute $u_i = H_i(w)$ , and $c_i = u_i^T \cdot s + y_i \in Z_q$ , where $x \leftarrow \chi^m$ and $y_i \leftarrow \chi$ are noise vectors. Output the cipher-text $(p, \{c_i\}_{i=1,...,k})$.
- $Trapdoor(A, T_A, w)$ : For $i = 1,...,k$ , compute $u_i = H_i(w)$ and choose a $e_i \in Z_q^m$ satisfying $A \cdot e_i = u_i$ by $e_i \leftarrow Sample\Pr e(A, T_A, u_i, \sigma)$, where $\sigma$ is a Gaussian parameter. Output $(\{e_i\}_{i=1,...,k})$ as the trapdoor.
- $Test((p, \{c_i\}_{i=1,...,k}), \{e_i\}_{i=1,...,k})$: For $i = 1,...,k$, compute $b_i = c_i - e_i^T \cdot p$. If for all $i = 1,...,k$, $|b_i| < q/4$, output 1 (“yes”); otherwise, output 0 (“no”).

Just as above discussion, if $w = w'$, then the *Test* algorithm output 1 with overwhelming probability. And if $w \neq w'$ , the *Test* algorithm output 0 with probability 1- $2^{-2k}$. When $k$ is large enough, we can ensure the consistency of the scheme.

### 3.3 Efficiency analysis

To guarantee the security from LWE problem, we select the parameters of our scheme following [21]: with the security parameter *k*, the public key is the matrix $A \in Z_q^{n \times m}$, where $n = k^3$, $q = 2^{\sqrt{n}} = 2^{k^{1.5}}$ and $m = 2n \log q \approx 2n\sqrt{n} = O(k^{4.5})$. We now analyze the computation of the basic construction. The computation of *p* and *c* in the *PKES* algorithm are $m \cdot n(\log q)^2 + m \log q$ and $n(\log q)^2 + \log q$ respectively. Hence the total computation is $(m+1) \cdot n(\log q)^2 + (m+1) \log q$. The computation of the *Test* algorithm is $m(\log q)^2 + \log q$.

We also implemented the algorithms of our scheme with Maple.14, on an Intel 2.9 GHz Core i7 system. With $k = 5, m = 2796, n = 125, q = 2321$, the size of public key is about 0.5MB, and the time of a *PEKS* algorithm is about 0.3 seconds.

Since its first introduction by Boneh et.al [12], the proposed PEKS schemes are mainly using pairings[12,17] and some traditional cryptography tools (such as RSA) [11]. Recently, due to the potential advantages, many lattices-based public key encryption schemes have been proposed[4-7]. But to the best of our knowledge, this is the first lattices-based *public-key encryption with keyword search* scheme.

### 3.4 Security Proof

We present the security proof for the basic construction. The security proof of improvement version is obviously almost the same.

**Proposition 3**. In the random oracle mode, suppose there is an polynomial-time adversary $\mathcal{F}$ that have non-negligible advantage $\varepsilon$ in attacking the scheme, then we can construct a polynomial-time algorithm $\mathcal{S}$ that can solve the LWE problem with probability $\frac{2\varepsilon(Q_H - 1)}{{Q_H}^2}$, where $Q_H$ denotes the number of queries that $\mathcal{F}$ can query to the random oracle $H(.)$.

**Proof.** With the adversary $\mathcal{F}$, we construct the algorithm $\mathcal{S}$ as follows:

$\mathcal{S}$ requests from $\Psi$ and receives, for each $i = 0,...,m$, a fresh pair $(u_i, v_i) \in Z_q^n \times Z_q$, and selects a random integer $Q^* \in [1, Q_H]$.

**KeyGen**. $\mathcal{S}$ assemble the random matrix $A \in Z_q^{n \times m}$ from $m$ of given LWE samples by letting the $i$_th column of $A$ be the vector $u_i$ for all $i = 1,...,m$, and gives the public key $A$ to $\mathcal{F}$.

**Phase 1**. $\mathcal{S}$ answers queries of $\mathcal{F}$ as follows:

--$H(.)$ query: For the $Q$_th query input $w \in \{0,1\}^*$, if this is query number $Q^*$ (i.e. $Q = Q^*$, $\mathcal{S}$ defines $H(w) = u_0$ and return $H(w)$. Otherwise, $\mathcal{S}$ chooses $e \leftarrow D_{Z^m,r}$, and computes $u = A \cdot e$, saves the tuple $(w,u,e)$ in $H\_list$ for future use, and returns $H(w) = u$.

--*Trapdoor*(.) query: For input $w \in \{0,1\}^*$, if $H(w) = u_0$, $\mathcal{S}$ aborts and fails. Otherwise, $\mathcal{S}$ retrieves the saved tuple $(w,u,e)$ from the $H\_list$ (w.l.o.g., we can assume that an Trapdoor query on $w$ is preceded by a $H(.)$ query with $w$). By construction, $e$ is the trapdoor for keyword $w$. Return $e$ as the trapdoor to the adversary.

**Challenge.** Once $\mathcal{F}$ decides that Phase 1 is over it outputs two keywords $w_0, w_1$ on which it wishes to be challenged. $\mathcal{S}$ runs the above algorithm for responding to $H(.)$ queries twice with input $w_0, w_1$. If both $H(w_0) \neq u_0$ and $H(w_1) \neq u_0$, then $\mathcal{S}$ aborts. Otherwise, suppose $H(w_b) = u_0$ $\mathcal{S}$ retrieves $v_0,...,v_m \in Z_q$ from the LWE instance, and sets $c_0 = [v_1,...,v_m]^T \in Z_q^m$, $c_1 = v_0 \in Z_q$. $\mathcal{S}$ responds with the challenge cipher-text $(c_0, c_1)$. (Note: When $\Psi$ is a pseudo-random LWE oracle then $c_0 = A^T \cdot s + x$ and $c_1 = u_0{}^T \cdot s + y$ for some random $s \in Z_q^n$ and noise values $x \in \chi^m$ and $y \in \chi$. In this case $(c_0, c_1)$ is a valid encryption for $w_b$. When $O$ is a random oracle then $(c_0, c_1)$ is uniform in $Z_q^m \times Z_q$.

**Phase 2.** $\mathcal{S}$ answers queries of $\mathcal{F}$ the same way it does in Phase 1 with the only restriction that $w \neq w_0, w_1$ for Trapdoor queries.

**Guess.** Eventually $\mathcal{F}$ outputs $b' \in \{0,1\}$.

Finally, if $b = b'$ $\mathcal{S}$ outputs 1, otherwise $\mathcal{S}$ outputs 0. The distribution of the public parameters is identical to its distribution in the real system as are responses to private key queries. If $\mathcal{S}$ does not abort then the challenge cipher-text is distributed either as in the real system or is independently random in $Z_q^m \times Z_q$. Hence, if $\mathcal{S}$ does not abort then its advantage in solving LWE is the same as $\mathcal{F}$'s advantage in attacking

the system.

Because $Q^*$ is chosen randomly in 1 and $Q_{H_1}$, $\mathcal{S}$ does not abort in the simulation of *Trapdoor*(.) with probability $1-1/Q_H$. The probability that $H(w_0)=u_0$ or $H(w_1)=u_0$ is $2/Q_H$. So we conclude that the advantage of $\mathcal{S}$ in solving LWE is $\frac{2\varepsilon(Q_H-1)}{{Q_H}^2}$.

## 4. Conclusion

Lattice-based cryptosystems are becoming increasingly popular in the research community. In this paper, we propose a PEKS scheme using lattices. Although fruitful lattices-based cryptographic schemes have been proposed, but to the best of our knowledge, this is the first lattices-based *public-key encryption with keyword search* scheme. The scheme can be proven secure with the hardness of the standard LWE problem in the random oracle model. We should note that it may be not a difficult task to modify this scheme to obtain a version secure in the standard model.

## References

[1]. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pages 99-108, New York, NY, USA, 1996. ACM.

[2]. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In FOCS, pages 356-365, 2002.

[3]. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, STOC, pages 197-206. ACM, 2008.

[4]. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing, pages 84-93, New York, NY, USA, 2005. ACM.

[5]. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, STOC, pages 197-206. ACM, 2008.

[6]. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In Advances in EUROCRYPT 2010, volume 6110 of LNCS, pages 553-572. Springer, 2010.

[7]. S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Advances in CRYPTO 2010, volume 6223 of LNCS, pages 98-115. Springer, 2010.

[8]. C. Gentry. Fully homomorphic encryption using ideal lattices. In STOC, pages 169-178, 2009.

[9]. C. Gentry. Toward basing fully homomorphic encryption on worst-case hardness.

In CRYPTO 2010, pages 116-137, 2010.

[10]. X. Boyen. Expressive Encryption Systems from Lattices. In: Cryptology and Network Security 2011. LNCS 9092, pp 1-12. Springer, 2011.

[11]. M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in Proceedings of Crypto 2007, volume 4622 of LNCS. Springer-Verlag, 2007.

[12]. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in Proc. of EUROCRYP'04, 2004.

[13]. B. Waters, D. Balfanz, G. Durfee, and D. Smetters, "Building an encrypted and searchable audit log," in Proc. of 11th Annual Network and Distributed System, 2004.

[14]. Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in Proc. of ACNS'05, 2005.

[15]. D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in Proc. of TCC'07, 2007, pp. 535–554.

[16]. J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy Keyword Search over Encrypted Data in Cloud Computing. IEEE INFOCOM 2010.

[17]. C. Gu, Y. Zhu. New Efficient Searchable Encryption Schemes from Bilinear Pairings. International Journal of Network Security, Vol.10, No.1, pp.25–31, Jan. 2010.

[18]. F. Bao, R. Deng, X. Ding, and Y. Yang, "Private query on encrypted data in multi-user settings," in Proc. of ISPEC'08, 2008.

[19]. X. Boyen. Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More. In: P.Q.Nguyen(eds.) PKC 2010. LNCS, vol 6056,pp.499-517,Springer, Heidelberg (2010)

[20]. J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In STACS, pages 75-86, 2009.

[21]. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE, IEEE 52nd Annual Symposium on Foundations of Computer Science(FOCS2011) Proceedings, Palm Springs, CA, USA, October 22-25, 2011, 97-106.

**Chunxiang Gu**, received his Ph.D. degree from Information Engineering University in 2006. He is an associate professor in Zhengzhou Information Science and Technology Institute. His research interests include network security and cryptography.
Email:gcxiang5209@yahoo.com.cn.