KEYWORD SEARCH WITH SECURE CLOUD STORAGE BY USING DUAL SERVER PUBLIC KEY ENCRYPTION

R.Saraswathi1, P.Karthika2
[1]Student Member, Department of Computer Science and Engineering,
[2] Staff Member, Department of Computer Science and Engineering,
VPMM Engineering College, Krishnankovil, TamilNadu, India.

ABSTRACT- Searchable encryption is of increasing interest for protecting the data privacy in secure searchable cloud storage. In this work, we investigate the security of a well-known cryptographic primitive, namely Public Key Encryption with Keyword Search (PEKS) which is very useful in many applications of cloud storage. Unfortunately, it has been shown that the traditional PEKS framework suffers from an inherent insecurity called inside Keyword Guessing Attack (KGA) launched by the malicious server. To address this security vulnerability, we propose a new PEKS framework named Dual-Server Public Key Encryption with Keyword Search (DS-PEKS). As another main contribution, we define a new variant of the Smooth Projective Hash Functions (SPHFs) referred to as linear homomorphic SPHF (LH-SPHF). We then show a generic construction of secure DS-PEKS from LH-SPHF. To illustrate the feasibility of our new framework, we provide an efficient instantiation of the general framework from a Decision Diffie-Hellman DDH-based LH-SPHF and show that it can achieve the strong security against inside KGA.

Keywords

Keyword search, secure cloud storage, encryption, inside keyword guessing attack, smooth projective hash function, Diffie-Hellman language.

1. INTRODUCTION

Cloud storage outsourcing has become a popular application for enterprises and organizations to reduce the burden of maintaining big data in recent years. However, in reality, end users may not entirely trust the cloud storage servers and may prefer to encrypt their data before uploading them to the cloud server in order to protect the data privacy. This usually makes the data utilization more difficult than the traditional storage where data is kept in the absence of encryption. One of the typical solutions is the searchable encryption which allows the user to retrieve the encrypted documents that contain the user-specified keywords, where given the keyword trapdoor, the server can find the data required by the user without decryption. Searchable encryption can be realized in either symmetric or asymmetric encryption setting. In proposed keyword search on ciphertext, known as Searchable Symmetric Encryption (SSE) and afterwards several SSE schemes were designed for improvements. Although SSE schemes enjoy high efficiency, they suffer from complicated secret key distribution. Precisely, users have to securely share secret keys which are used for data encryption.

Otherwise they are not able to share the encrypted data outsourced to the cloud. To resolve this problem, Bonehet al. introduced a more flexible primitive, namely Public Key Encryption with Keyword Search (PEKS) that enables a user to search encrypted data in the asymmetric encryption setting. In a PEKS system, using the receiver's public key, the sender attaches some encrypted keywords (referred to as PEKS ciphertexts) with the encrypted data. The receiver then sends the trapdoor of a to-be-searched keyword to the server for data searching. Given the trapdoor and the PEKS ciphertext, the server can test whether the keyword underlying the PEKS ciphertext is equal to the one selected by the receiver. If so, the server sends the matching encrypted data to the receiver. Despite of being free from secret key distribution, PEKS schemes suffer from an inherent insecurity regarding the trapdoor keyword privacy, namely inside Keyword Guessing Attack (KGA). The reason leading to such a security vulnerability is that anyone who knows receiver's public key can generate the PEKS ciphertext of arbitrary keyword himself. Specifically, given a trapdoor, the adversarial server can choose a guessing keyword from the keyword space and then use the keyword to generate a PEKS ciphertext. The server then can test whether the guessing keyword is the one underlying the trapdoor. This guessing-then-testing procedure can be repeated until the correct keyword is found. Such a guessing attack has also been considered in many password-based systems.

However, the attack can be launched more efficiently against PEKS schemes since the keyword space is roughly the same as a normal dictionary (e.g., all the meaningful English words), which has a much smaller size than a password dictionary (e.g., all the words containing 6 alphanumeric characters). It is worth noting that in SSE schemes, only secret key holders can generate the keyword ciphertext and hence the adversarial server is not able to launch the inside KGA. As the keyword always indicates the privacy of the user data, it is therefore of practical importance to overcome this security threat for secure searchable encrypted data outsourcing.

The contributions of this paper are four-fold.

- We formalize a new PEKS framework named Dual-Server Public Key Encryption with Keyword Search (DSPEKS) to address the security vulnerability of PEKS.
- A new variant of Smooth Projective Hash Function (SPHF), referred to as linear homomorphic SPHF, is introduced for a generic construction of DS-PEKS.
- We show a generic construction of DS-PEKS using the proposed Lin-Hom SPHF.
- To illustrate the feasibility of our new framework, an efficient instantiation of our SPHF based on the Diffie-Hellman language is presented in this paper

In this paper propose a new framework, namely DSPEKS, and present its formal definition and security models. We then define a new variant of smooth projective hash function (SPHF). A generic construction of DS-PEKS from LH-SPHF with formal correctness analysis and security proofs. Finally, we present an efficient instantiation of DS-PEKS from SPHF based on a language defined by the Diffie-Hellman problem. They also analyze the performance of our scheme through comparisons with existing works and experimental evaluation.

1.1 Related Work

In this subsection, we describe a classification of PEKS schemes based on their security.

Traditional PEKS. Following Boneh et al.'s seminal work[5], Abdalla et al. [8] formalized anonymous IBE (AIBE) and presented a generic construction of searchable encryption from AIBE. They also showed how to transfer a hierarchical IBE (HIBE) scheme into a public key encryption with temporary keyword search (PETKS) where the trapdoor is only valid in a specific time interval. Waters [7] showed that the PEKS schemes based on bilinear map could be applied to build encrypted and searchable auditing logs. In order to construct a PEKS secure in the standard model, Khader [9] proposed a scheme based on the k-resilient IBE and also gave a construction supporting multiple-keyword search. The first PEKS scheme without pairings was introduced by Di Crescenzo and Saraswat [11]. The construction is derived from Cock's IBE scheme [12] which is not very practical.

Secure Channel Free PEKS. The original PEKS scheme [5] requires a secure channel to transmit the trapdoors. To overcome this limitation, Baek et al. [13] proposed a new PEKS scheme without requiring a secure channel, which is referred to as a secure channel-free PEKS (SCF-PEKS). The idea is to add the server's public/private key pair into a PEKS system. The keyword ciphertext and trapdoor aregenerated using the server's public key and hence only the server (designated tester) is able to perform the search. Rhee et al. [14] later enhanced Baek et al.'s security model [13] for SCF-PEKS where the attacker is allowed to obtain the relationship between the non-challenge ciphertexts and the trapdoor. They also presented an SCF-PEKS scheme secure under the enhanced security model in the random oracle model. Another extension on SCF-PEKS is by Emura et al. [15]. They enhanced the security model by introducing the adaptively secure SCF-PEKS, wherein an adversary is allowed to issue test queries adaptively.

Against Outside KGA. Byun et al. [16] introduced the offline keyword guessing attack against PEKS as keywords are chosen from a much smaller space than passwords and users usually use well-known keywords for searching documents. They also pointed out that the scheme proposed in Boneh et al. [5] was susceptible to keyword guessing attack. Inspired by the work of Byun et al. [16], Yau et al. [17] demonstrated that outside adversaries that capture the trapdoors sent in a public channel can reveal the encrypted keywords through off-line keyword guessing attacks and they also showed off-line keyword guessing attacks against the (SCF-)PEKS schemes in [13], [18]. The first PEKS scheme secure against outside keyword guessing attacks was proposed by Rhee et al. [19]. In [20], the notion of trapdoor indistinguishability was proposed and the authors showed that trapdoor indistinguishability is a sufficient condition for preventing outside keyword-guessing attacks. Fang et al. [21] proposed a concrete SCF-PEKS scheme with (outside) KGA resilience. Similar to the work in [15], they also considered the adaptive test oracle in their proposed security definition.

Against Inside KGA. Nevertheless, all the schemes mentioned above are found to be vulnerable to keyword guessing attacks from a malicious server (i.e., inside KGA). Jeong et al. [22] showed a negative result that the consistency/ correctness of PEKS implies insecurity to inside KGA in PEKS. Their result indicates that constructing secure and consistent PEKS schemes against inside KGA is impossible under the original framework. A potential solution is to propose a new framework of PEKS. In [10], Peng et al. proposed the notion of Public-key Encryption with Fuzzy Keyword Search (PEFKS) where each keyword corresponds to an exact trapdoor and a fuzzy trapdoor. The server is only provided with the fuzzy trapdoor and thus can no longer learn the exact keyword since two or more keywords share the same fuzzy keyword trapdoor. However, their scheme suffers from several limitations regarding the security and efficiency. On one hand, although the server cannot exactly guess the keyword, it is still able to know which small set the underlying keyword belongs to and thus the keyword privacy is not well preserved from the server. On the other hand, their scheme is impractical as the receiver has to locally find the matching ciphertext by using the exact trapdoor to filter out the non-matching ones from the set returned from the server.

Differences Between This Work and Its Preliminary Version

[1]. Portions of the work presented in this paper have previously appeared as an extended abstract [1]. Compared to [1], we have revised and enriched the work substantially in the following aspects. First, in the preliminary work [1] where our generic DS-PEKS construction was presented, we showed neither a concrete construction of the linear and homomorphic SPHF nor a practical instantiation of the DS-PEKS framework. To fill this gap and illustrate the feasibility of the framework, in this paper (Section 6), we first show that a linear and homomorphic language LDH can be derived from the Diffie-Hellman assumption and then construct a concrete linear and homomorphic SPHF, referred to as SPHFDH, from LDH. We provide a formal proof that SPHFDH is correct, smooth and pseudo-random and hence can be used for the instantiation of our generic construction. We then present a concrete DS-PEKS scheme from SPHFDH. To analyze its performance, we first give a comparison between existing schemes and our scheme and then evaluate its performance in experiments. We also revised the preliminary version [1] to enhance the presentation and readability. In the related work part, compared to the preliminary version, we add more literatures and give a clearer classification of the existing schemes based on their security. We present the security models of DS-PESK as experiments to make them more readable. Moreover, to make the concepts of SPHF and our newly defined variant clearer, we add Fig. 4 and Fig. 5 to highlight their key properties.

1.2 Organization

In Section 2, We propose a new framework, namely DSPEKS, and present its formal definition and security models. We then define a new variant of smooth projective hash function (SPHF)in In Section 3. A generic construction of DS-PEKS from LH-SPHF is shown in Section 5 with formal correctness analysis and security proofs. Finally, we present an efficient instantiation of DS-PEKS from SPHF based on a language defined by the Diffie-Hellman problem in Section 6. We also analyze the performance of our scheme through comparisons with existing works and experimental evaluation.

2 A NEW FRAMEWORK FOR PEKS

In this section, we formally define the Dual-Server Public Key Encryption with Keyword Search (DS-PEKS) and its security model.

2.1 Definition of DS-PEKS

A DS-PEKS scheme mainly consists of (KeyGen, DS-PEKS, DS-Trapdoor; FrontTest; BackTest). To be more precise, the KeyGen algorithm generates the public/private key pairs of the front and back servers instead of that of the receiver. Moreover, the trapdoor generation algorithm DS-Trapdoor defined here is public while in the traditional PEKS definition [5], [13], the algorithm Trapdoor takes as input the receiver's private key. Such a difference is due to the different structures used by the two systems. In the traditional PEKS, since there is only one server, if the trapdoor generation algorithm is public, then the server can launch a guessing attack against a keyword ciphertext to recover the encrypted keyword. As a result, it is impossible to achieve the semantic security as defined in [5], [13]. However, as we will show later, under the DS-PEKS framework, we can still achieve semantic security when the trapdoor generation algorithm is public. Another difference between the traditional PEKS and our proposed DS-PEKS is that the test algorithm is divided into two algorithms, FrontTest and BackTest run by two independent servers. This is essential for achieving security against the inside keyword guessing attack. In the DS-PEKS system, upon receiving a query from the receiver, the front server pre-processes the trapdoor and all the PEKS ciphertexts using its private key, and then sends some internal testing-states to the back server with the corresponding trapdoor and PEKS ciphertexts hidden. The back server can then decide which documents are queried by the receiver using its private key and the received internal testing-states from the front server.

Definition 1 (DS-PEKS). A DS-PEKS scheme is defined by the following algorithms.

- Setup(1_). Takes as input the security parameter, generates the system parameters P;
- KeyGen(P): Takes as input the systems parameters P, outputs the public/secret key pairs (pkFS; skFS), and (pkBS; skBS) for the front server, and the back server respectively;
- DS-PEKS(P; pkFS; pkBS; kw1): Takes as input P, the front server's public key pkFS, the back server's public key pkBS and the keyword kw1, outputs the PEKS ciphertext CTkw1 of kw1;
- DS-Trapdoor(P; pkFS; pkBS; kw2): Takes as input P, the front server's public key pkFS, the back server's public key pkBS and the keyword kw2, outputs the trapdoor Tkw2 ;
- FrontTest(P; skFS;CTkw1 ; Tkw2): Takes as input P, the front server's secret key skFS, the PEKS ciphertext CTkw1 and the trapdoor Tkw2 , outputs the internal testing-state CITS;
- BackTest(P; skBS;CITS): Takes as input P, the back server's secret key skBS and the internal testing-state CITS, outputs the testing result 0 or 1;

Correctness. It is required that for any keyword kw1; kw2, and CTkw1 DS-PEKS(P; pkFS; pkBS; kw1), Tkw2 <- DS-Trapdoor(P; pkFS; pkBS; kw2), we have

BackTest(P; skBS;CITS) ={1 kw1 = kw2;0 kw1 6= kw2:

where CITS FrontTest(P; skFS;CTkw1 ; Tkw2):

2.2 Security Models

In this subsection, we formalise the following security models for a DS-PEKS scheme against the adversarial front and back servers, respectively. One should note that both the front server and the back server here are supposed to be "honest but curious" and will not collude with each other. More precisely, both the servers perform the testing strictly following the scheme procedures. but may be curious about the underlying keyword. We should note that the following security models also imply the security guarantees against the outside adversaries which have less capability compared to the servers.

Adversarial Front Server. In this part, we define the security against an adversarial front server.We introduce two games, namely semantic-security against chosen keyword attack and indistinguishability against keyword guessing attack1 to capture the security of PEKS ciphertext and trapdoor, respectively.

I. Semantic-Security against Chosen Keyword Attack: In the following, we define the semantic-security against chosen keyword attack which guarantees that no adversary is able to distinguish a keyword from another one given the corresponding PEKS ciphertext. That is, the PEKS ciphertext does not reveal any information about the underlying keyword to any adversary.

Formally, we introduce an experiment in Fig. 1 for the SS-CKA security definition against the adversarial front server. In the experiment, the adversary A is given the public/private key pair of the front server and the public key of the back server. In the find phase, A can test any pair of PEKS ciphertext and keyword by querying the oracle OT and eventually output two challenging keywords (kw0; kw1) with the hint information "state". With a randobit b 2 f0; 1g as input, the experiment generates and then sends the PEKS ciphertext CT_ kw of keyword kwb to
A. During the guess phase, A can continue the query to OT and finally output its guess b0. The guess b0 is a valid output of the experiment if and only if that A has never queried OT with the challenge keywords. We refer to such an adversarial front server A in the above experiment as an SS-CKA adversary and define its advantage as
Adv SS-CKA FS;A (ג) = Pr[b = b0] □ 1=2:

Ⅱ. Indistinguishability against Keyword Guessing Attack: This security model captures that the trapdoor reveals no information about the underlying keyword to the adversarial front server. We define the security experiment as shown in Fig. **2.** The experiment is similar to that of SS-CKA experiment except that in the challenge phase, the adversary is given the trapdoor instead of the PEKS ciphertext.
We refer to such an adversarial front server A in the above experiment as an IND-KGA adversary and define its advantage as
AdvIND-KGA FS;A (ג) = Pr[b = b0] □ 1=2:
Adversarial Back Server. The security models of SS-CKA and IND-KGA in terms of an adversarial back server are similar to those against an adversarial front server.
Ⅲ. Semantic-Security against Chosen Keyword Attack. Here the SS-CKA experiment against an adversarial back server is the same as the one against an adversarial front server except that the adversary is given the private key of the back server instead of that of the front server. We omit the details here for simplicity. We refer to the adversarial back server A in the SS-CKA experiment as an SS-CKA adversary and define its advantage as
AdvSS-CKA BS;A (ג) = Pr[b = b0] □ 1=2:
Ⅳ. Indistinguishability against Keyword Guessing Attack. Similarly, this security model aims to capture that the trapdoor does not reveal any information to the back server and hence is the same as that against the front server except that the adversary owns the private key of the back server instead of that of the front server. Therefore, we also omit the details here. We refer to the adversarial back server A in the IND-KGA experiment as an IND-KGA adversary and define its advantage as
AdvIND-KGA BS;A (ג) = Pr[b = b0] □ 1=2:
V. Indistinguishability against Keyword Guessing Attack-II. In our defined security notion of IND-KGA-II, as shown in Fig. 3, it is required that a malicious back server cannot learn any information about the underlying two keywords involved in the internal testing-state. First of all, we should note that both keywords involved in the internal-testing state plays the same role regardless of their initial source (i.e., from the PEKS ciphertext or the trapdoor). Therefore, the task of the adversary is to guess the two underlying keywords in the internal testing state as a whole, instead of each one in the initial PEKS ciphertext and the initial trapdoor. Therefore, it is insufficient for the adversary to submit only two challenge keywords and hence we require the adversary to submit three different keywords in the challenge stage and guess which two keywords are chosen given the challenge internal-testing state.
Formally, in the experiment, the adversary A is given the public key of the front server and the public/private key pair of the back server. In the challenge phase, the adversary outputs three challenging keywords (kw0; kw1; kw2).

3. SMOOTH PROJECTIVE HASH FUNCTIONS

A central element of our construction for dual-server public key encryption with keyword search is smooth projective hash function (SPHF), a notion introduced by Cramer and Shoup [23]. We start with the original definition of an SPHF.

3.1 Original Definition of SPHFs

As illustrated in Fig. 3.1 an SPHF is defined based on a domain X and an NP language L, where L contains a subset of the elements of the domain X, i.e., L _ X. Formally, an SPHF system over a language L _ X, onto a set Y, is defined by the following five algorithms (SPHFSetup; HashKG, ProjKG; Hash; ProjHash):

- SPHFSetup(1_): generates the global parameters param and the description of an NP language instance L;

- HashKG(L; param): generates a hashing key hk for L;
- ProjKG(hk; (L; param)): derives the projection key hp from the hashing key hk;
- Hash(hk; (L; param);W): outputs the hash value hv 2 Y for the word W from the hashing key hk;
- ProjHash(hp; (L; param);W;w): outputs the hash value hv0 2 Y for the word W from the projection key hp and the witness w for the fact that W 2 L.

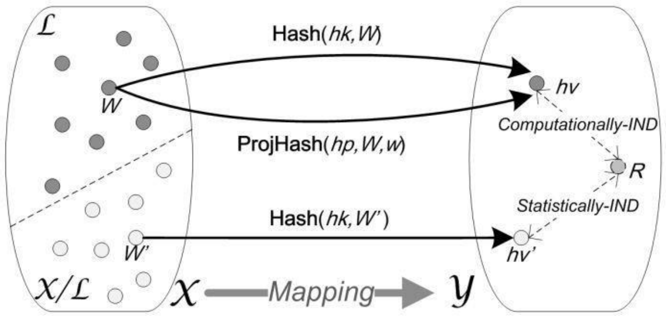

Fig. 3.1 Smooth Projective Hash Function

3.2 A New Variant–Linear and Homomorphic SPHFs

In this paper, we introduce a new variant of smooth projective hash function. In addition to the original properties, we consider two new properties – linear and homomorphic, which are defined below.

4 GENERIC CONSTRUCTION OF DS-PEKS

4.1 Generic Construction

Let SPHF = (SPHFSetup; HashKG; ProjKG; Hash; ProjHash) be a LH-SPHF over the language L onto the set Y. Let W be the witness space of the language L and KW be the keyword space. Our generic construction DS-PEKS works

4.2 Security of DS-PEKS

In this subsection, we analyse the security of the above generic construction DS-PEKS. Due to the space limitation, we omit the proof details here and refer readers to [1] for a full security proof.

Theorem 1. The generic construction DS-PEKS is semantically secure under chosen keyword attacks.

CONCLUSIONw

The proposed a new framework, named Dual-Server Public Key Encryption with Keyword Search (DSPEKS), that can prevent the inside keyword guessing attack which is an inherent vulnerability of the traditional PEKS framework. We also introduced a new Smooth Projective Hash Function (SPHF) and used it to construct a generic DSPEKS scheme. An efficient instantiation of the new SPHF based on the Diffie-Hellman problem is also presented in the paper, which gives an efficient DS-PEKS scheme without pairings.

REFERENCES

[1] Rongmao Chen, Yi Mu, Guomin Yang, Fuchun Guo and Xiaofen Wang "Dual-Server Public-Key Encryption with Keyword Search for Secure Cloud Storage" IEEE transactions on Informatin Forensics and Security, Vol. 11, pp.789-798, 2015

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in EUROCRYPT, 2004, pp. 506–522

[3] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in Computational Science and Its Applications - ICCSA, 2008, pp. 1249–1259.

[4] D. Khader, "Public key encryption with keyword search based on k-resilient IBE," in Computational Science and Its Applications - ICCSA, 2006, pp. 298–308

[5] G. D. Crescenzo and V. Saraswat, "Public key encryption with searchable keywords based on jacobi symbols," in INDOCRYPT, 2007, pp. 282–296.

[6] P. Xu, H. Jin, Q. Wu, and W. Wang, "Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack," IEEE Trans. Computers, vol. 62, no. 11, pp. 2266– 2277, 2013.

[7] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable

encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in CRYPTO, 2005, pp. 205–222.

[8] K. Emura, A. Miyaji, M. S. Rahman, and K. Omote, "Generic constructions of secure-channel free searchable encryption with adaptive security," Security and Communication Networks, vol. 8, no. 8, pp. 1547–1560, 2015.

[9] J. W. Byun, H. S. Rhee, H. Park, and D. H. Lee, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," in Secure Data Management, Third VLDBWorkshop, SDM, 2006, pp. 75–83.

[10] R. Gennaro and Y. Lindell**,** "A framework for password-based authenticated key exchange," in EUROCRYPT, 2003, pp. 524–543.