International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 3, March 2016

Ensuring Semantic Security and anonymity using SPCHS Scheme

[1]Saravanan.S, [2]Vaiibhav.K.S, [3]Vigneshwar.R, [4]Vignesh Adithya K.L
[1]Assistant Professor, [2],[3],[4] Final Year B.E
[1][2][3][4] R.M.K. College of Engineering and Technology, Chennai.
[1]saravanan.s@rmkcet.ac.in,[2]vaiibhavsridhar@gmail.com,[3]vigneshwarravichandrababu@gmail.com,
[4]vickykutty129@gmail.com

Abstract- **The Existing semantically secure public-key searchable encryption schemes take search time linear with the total number of the cipher texts which makes retrieval from large-scale databases prohibitive. In order to overcome this problem, this paper proposes searchable public-key cipher texts with the hidden structures (SPCHS) for keyword search as fast as possible without compromising with semantic security of the encrypted keywords. In SPCHS, all keyword-searchable cipher texts are structured by the hidden relations, and also with the search trapdoor corresponding to a keyword, the minimum information is disclosed to a search algorithm as the guidance to find all matching cipher texts in an efficient manner. We have constructed an SPCHS scheme in which the cipher texts have a hidden star-like structure. We prove our proposed scheme to be semantically secure in the random oracle (RO) model. The search complexity of our scheme is fully dependent on the actual number of the cipher texts containing the queried keyword, rather than the number of all cipher texts. Hence we present a generic SPCHS construction from anonymous identity-based encryption and collision-free identity-based key encapsulation mechanism (IBKEM) with anonymity. We are interested in providing highly efficient search performance without compromising with semantic security in PEKS. We illustrate two collision-free IBKEM instances, which are semantically secure and anonymous, respectively, in the RO and standard models. The latter instance enables us to construct an SPCHS scheme with semantic security in the standard model.**

Index Terms— **Public-key searchable encryption, semantic security, identity-based key encapsulation mechanism, identity based encryption.**

I. INTRODUCTION

The advantage of Public key Encryption with Keyword Search is that anyone who knows the receiver's public key can upload keyword-searchable cipher texts to a server. The receiver can delegate the keyword search to the server. More specifically, each sender separately encrypts a file and its extracted keywords and sends the resulting cipher texts to a server; when the receiver wants to retrieve the files containing a specific keyword, he delegates a keyword search trapdoor to the server; the server finds the encrypted files containing the queried keyword without knowing the original files or the keyword itself, and returns the corresponding encrypted files to the receiver; finally, the receiver decrypts these encrypted files.[1] The authors of PEKS [1] also presented. semantic security against chosen keyword attacks (SS- CKA) in the sense that the server cannot distinguish the cipher texts of the keywords of its choice before observing the corresponding keyword search trapdoors. It seems an appropriate security notion, especially if the keyword space has no high min-entropy. Existing semantically secure PEKS schemes take search time linear with the total number of all cipher texts. This makes retrieval from large-scale databases prohibitive. Therefore, more efficient search performance is crucial for practically deploying PEKS schemes.

One of the prominent works to accelerate the search over encrypted keywords in the public-key setting is deterministic encryption introduced by Bellare et al. in [2]. An encryption scheme is deterministic if the encryption algorithm is deterministic. Bellare et al. [2] focus on enabling search over encrypted keywords to be as efficient as the search for unencrypted keywords, such that a cipher text containing a given keyword can be retrieved in time complexity logarithmic in the total number of all cipher texts. This is reasonable because the encrypted keywords to their binary values. However, deterministic encryption has two inherent limitations. First, keyword privacy can be guaranteed only for keywords that are a priori hard-to-guess by the adversary (i.e., keywords with high min-entropy to the adversary); second, certain information of a message

leaks inevitably via the cipher rtext of the keywords since the encryption is deterministic. Hence, deterministic encryption is only applicable in special scenarios.

A. Our Motivation and Basic Ideas

We are interested in providing highly efficient search performance without sacrificing semantic security in PEKS. Observe that a keyword space is usually of no high min-entropy in many scenarios. Semantic security is crucial to guarantee keyword privacy in such applications. Thus the linear search complexity of existing schemes is the major obstacle to their adoption. Unfortunately, the linear complexity seems to be inevitable because the server has to scan and test each cipher text, due to the fact that these cipher texts are indistinguishable to the server.

A closer look shows that there is still space to improve search performance in PEKS without sacrificing semantic security if one can organize the cipher texts with elegantly designed but hidden relations. Intuitively, if the keyword-searchable cipher texts have a hidden star-like structure, as shown in Figure 1, then search over cipher texts containing a specific keywords may be accelerated. Specifically, suppose all cipher texts of the same keyword form a chain by the correlated hidden relations, and also a hidden relation exists from a public Head to the first cipher text of each chain. With a keyword search trapdoor and the Head, the server seeks out the first matching cipher text via the corresponding relation from the Head. Then another relation can be disclosed via the found cipher text and guides the searcher to seek out the next matching cipher text. By carrying on in this way, all matching cipher texts can be found. Clearly, the search time depends on the actual number of the cipher texts containing the queried keyword, rather than on the total number of all cipher texts.

To guarantee appropriate security, the hidden star-like structure should preserve the semantic security of keywords, which indicates that partial relations are disclosed only when the corresponding keyword search trapdoor is known. Each sender should be able to generate the keyword-searchable cipher texts with the hidden star-like structure by the receiver's public-key; the server having a keyword search trapdoor should be able to disclose partial relations, which is related to all matching cipher texts.

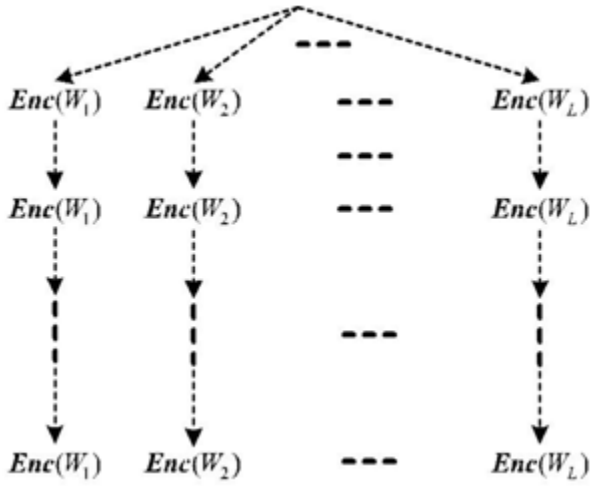

Fig. 1. Hidden star-like structure formed by keyword searchable cipher texts

B. Our Work

We start by formally defining the concept of Searchable Public-key Cipher texts with Hidden Structures (SPCHS) and its semantic security. In this new concept, keyword-searchable cipher texts with their hidden structures can be generated in the public key setting; with a keyword search trapdoor, partial relations can be disclosed to guide the discovery of all matching cipher texts. Semantic security is defined for both the keywords and the hidden structures. It is worth noting that this new concept and its semantic security are suitable for keyword-searchable cipher texts with any kind of hidden structures. In contrast, the concept of traditional PEKS does not contain any hidden structure among the PEKS cipher texts; correspondingly, its semantic security is only defined for the keywords.

Following the SPCHS definition, we construct a simple SPCHS from scratch in the random oracle (RO) model. The scheme generates keyword-searchable cipher texts with a hidden star-like structure. The search performance mainly depends on the actual number of the cipher texts containing the queried keyword. For security, the scheme is proven semantically secure based on the Decisional Bilinear Diffie-Hellman (DBDH) assumption [3] in the RO model.

We are also interested in providing a generic SPCHS construction to generate keyword-searchable cipher texts with a hidden star-like structure. Our generic SPCHS is inspired by several interesting observations on Identity-Based Key Encapsulation Mechanism (IBKEM).

We build a generic SPCHS construction with Identity-Based Encryption (IBE) and collision-free full-identity

malleable IBKEM.

C. Related Work

Search on encrypted data has been extensively investigated in recent years. From a cryptographic perspective, the existing works fall into two categories, i.e., symmetric searchable encryption [14] and public-key searchable encryption.

Symmetric searchable encryption is occasionally referred to as symmetric-key encryption with keyword search (SEKS). This primitive was introduced by Song et al. in [15]. Their instantiated scheme takes search time linear with the size of the database. A number of efforts [16]–[20] follow this research line and refine Song et al.'s original work. The SEKS scheme due to Curtmola et al. [14] has been proven to be semantically secure against an adaptive adversary. It allows the search to be processed in logarithmic time, although the keyword search trapdoor has length linear with the size of the database. In addition to the above efforts devoted to either provable security or better search performance, attention has recently been paid to achieving versatile SEKS schemes as follows. The works in [14] and [21] extend SEKS to a multi-sender scenario. The work in [22] realizes fuzzy keyword search in the SEKS setting. The work in [23] shows practical applications of SEKS and employs it to realize secure and searchable audit logs. Chase et al. [24] proposed to encrypt structured data and a secure method to search these data. To support the dynamic update of the encrypted data, Kamara et al. proposed the dynamic searchable symmetric encryption in [25] and further enhanced its security in [26] at the cost of large index. The very recent work [27] due to Cash et al. simultaneously achieves strong security and high efficiency.

II. PROPOSED SPCHS SCHEME

We first explain intuitions behind SPCHS. We describe a hidden structure formed by cipher texts as (C, **Pri**, **Pub**), where C denotes the set of all cipher texts, **Pri** denotes the hidden relations among C, and **Pub** denotes the public parts. In case there is more than one hidden structure formed by cipher-texts, the description of multiple hidden structures formed by cipher texts can be (C, ($\mathbf{Pri}_1$, $\mathbf{Pub}_1$), ..., ($\mathbf{Pri}_N$, $\mathbf{Pub}_N$)), where N ∈ N. Moreover, given (C, $\mathbf{Pub}_1$, ..., $\mathbf{Pub}_N$) and ($\mathbf{Pri}_1$, ..., $\mathbf{Pri}_N$) except ($\mathbf{Pri}_i$, $\mathbf{Pri}_j$) (where i _= j), one can neither learn anything about ($\mathbf{Pri}_i$, $\mathbf{Pri}_j$) nor decide whether a cipher text is associated with $\mathbf{Pub}_i$ or $\mathbf{Pub}_j$.

In SPCHS, the encryption algorithm has two functionalities. One is to encrypt a keyword, and the other is to generate a hidden relation, which can associate the generated cipher text to the hidden structure. Let (**Pri**, **Pub**) be the hidden structure. The encryption algorithm must take **Pri** as input, otherwise the hidden relation cannot be generated since **Pub** does not contain anything about the hidden relations. At the end of the encryption procedure, the **Pri** should be updated since a hidden relation is newly generated (but the specific method to update **Pri** relies on the specific instance of SPCHS). In addition, SPCHS needs an algorithm to initialize (**Pri**, **Pub**) by taking the master public key as input, and this algorithm will be run before the first time to generate a cipher text. With a keyword search trapdoor, the search algorithm of SPCHS can disclose partial relations to guide the discovery of the cipher texts containing the queried keyword with the hidden structure.

Definition 1 (SPCHS): SPCHS consists of five algorithms:

- **SystemSetup**(1^k , W): Take as input a security parameter 1^k and a keyword space W, and probabilistically output a pair of master public-and-secret keys (**PK**, **SK**), where

  PK includes the keyword space W and the cipher text space C.

- **StructureInitialization**(**PK**): Take as input **PK**, and probabilistically initialize a hidden structure by outputting its private and public parts (**Pri**, **Pub**).

- **StructuredEncryption**(**PK**, W, **Pri**): Take as inputs **PK**, a keyword W ∈ W and a hidden structure's private part **Pri**, and probabilistically output a keyword-searchable

  cipher text C of keyword W with the hidden structure, and update **Pri**.

- **Trapdoor**(**SK**, W): Take as inputs **SK** and a keyword W ∈ W, and output a keyword search trapdoor T_W of W .
- **Structured Search**(**PK**, **Pub**, C, T_W): Take as inputs **PK**, a hidden structure's public part **Pub**, all keyword-

  searchable cipher texts C and a keyword search trapdoor T_W of keyword W , disclose partial relations to guide finding out the cipher texts containing keyword W with the hidden structure.

An SPCHS scheme must be consistent in the sense that given any keyword search trapdoor T_W and any hidden structure's public part **Pub**, algorithm

Structured Search(**PK**, **Pub**, C, T_W) finds out all cipher texts of keyword W with the hidden structure **Pub**.

In the application of SPCHS, a receiver runs algorithm **System Setup** to set up SPCHS. Each sender uploads the public part of his hidden structure and keyword-searchable cipher texts to a server, respectively by algorithms

Structure Initialization and **Structured Encryption**. Algorithm **Trapdoor** allows the receiver to delegate a keyword search trapdoor to the server. Then the server runs algorithm **Structured Search** for all senders' structures to find out the cipher texts of the queried keyword.

The above SPCHS definition requires each sender to maintain the private part of his hidden structure for algorithm **Structured Encryption**. A similar requirement appears in symmetric-key encryption with keyword search (SEKS) in which each sender is required to maintain a secret key shared with the receiver. This implies interactions via authenticated confidential channels before a sender encrypts the keywords to the receiver in SEKS. In contrast, each sender in SPCHS just generates and maintains his/her private values locally, i.e., without requirement of extra secure interactions before encrypting keywords.

In the general case of SPCHS, each sender keeps his/her private values **Pri**. We could let each sender be stateless by storing his/her **Pri** in encrypted form at a server and having each sender download and re-encrypt his/her **Pri** for each update of **Pri**. A similar method also has been suggested by [27].

The semantic security of SPCHS is to resist adaptively chosen keyword and structure attacks (SS-CKSA). In this security notion, a probabilistic polynomial-time (PPT) adversary is allowed to know all structures' public parts, query the trapdoors for adaptively chosen keywords, query the private parts of adaptively chosen structures, and query the cipher texts of adaptively chosen keywords and structures (including the keywords and structures which the adversary would like to be challenged). The adversary will choose two challenge keyword-structure pairs. The SS-CKSA security means that for a cipher text of one of two challenge keyword-structure pairs, the adversary cannot determine which chal-lenge keyword or which challenge structure the challenge cipher text corresponds to, provided that the adversary does not know the two challenge keywords' search trapdoors and the two challenge structures' private parts.

Definition 2 (SS-CKSA Security): Suppose there are at most $N \in \mathbb{N}$ hidden structures. An SPCHS scheme is SS-CKSA secure, if any PPT adversary A has only a negligible advantage $Adv^{SS\text{-}CKSA,A}$ to win in the following SS-CKSA game:

S PC H S

- ***Setup Phase***: A challenger sets up the SPCHS scheme by running algorithm **SystemSetup** to generate a pair of master public-and-secret keys (**PK**, **SK**), and initializes N hidden structures by running algorithm

  Structure Initialization N times (let **PSet** be the set of all public parts of these N hidden structures.); finally the challenger sends **PK** and **PSet** to A.

- ***Query Phase 1***: A adaptively issues the following queries multiple times.
  – Trapdoor Query $Q_{T\ rap}$ (W): Taking as input a keyword $W \in \mathbb{W}$, the challenger outputs the keyword search trapdoor of keyword W ;

  – Privacy Query Q_{Pri} (**Pub**): Taking as input a hidden structure's public part **Pub** $\in$ **PSet**, the challenger outputs the corresponding private part of this structure;

  – Encryption Query Q_{Enc} (W, **Pub**): Taking as inputs a keyword $W \in \mathbb{W}$ and a hidden structure's public part **Pub**, the challenger outputs an SPCHS cipher text of keyword W with the hidden structure **Pub**.

- **Challenge Phase**: A sends two challenge keyword-and-structure pairs (W_0^* , $\mathbf{Pub}^*_0$) $\in \mathbb{W} \times$ **PSet** and (W_1^* , $\mathbf{Pub}^*_1$) $\in \mathbb{W} \times$ **PSet** to the challenger; The challenger randomly chooses $d \in \{0, 1\}$, and sends a challenge cipher text C_d^* of keyword W_d^* with the hidden structure $\mathbf{Pub}^*_d$ to A.

- **Query Phase 2**: This phase is the same as **Query Phase 1**. Note that in **Query Phase 1** and **Query Phase 2**,

  A cannot query the corresponding private parts both of $\mathbf{Pub}^*_0$ and $\mathbf{Pub}^*_1$ and the keyword search trapdoors both of W_0^* and W_1^* .
- **Guess Phase**: A sends a guess d to the challenger. We say that A wins if d = d . And let $Adv^{SS\text{-}CKSA,A}$ =

S PC H S

Pr [d = d] − $\frac{1}{2}$ be the advantage of A to win in the above game. A weaker security definition of SPCHS is the selective-keyword security. We refer to this weaker security notion as SS-sK-CKSA security, and the corresponding

attack game as SS-sK-CKSA game. In this attack game, the adversary A chooses two challenge keywords before the SPCHS scheme is set up., but the adversary still adaptively chooses two challenges.

III. A SIMPLE SPCHS SCHEME FROM SCRATCH

e : $G \times G \rightarrow G_1$ [44], [45] is an efficiently computable bilinearity and non-degenerate function with the property $e(g^a, g^b)$ $e(g, g)^{ab}$, where (a, b) $\$$ Z^* and $e(g, g)$ is hidden structures at **Challenge Phase**. Let $Adv^{SS\text{-}sK\text{-}CKSA}$ denote the advantage of adversary A to win in this game.

III. A SIMPLE SPCHS SCHEME FROM SCRATCH

Let $\gamma \xleftarrow{\$}$ denote an element γ randomly sampled from . Let G and G_1 denote two multiplicative groups of

Figure 2 shows a hidden star-like structure, which is generated by the SPCHS instance. The algorithm will find out all cipher texts of keyword W_i with the hidden star-like structure, and stop the search if no matching cipher text is found.

Structured Search repetitively discloses the value of Pt^- and matches the value with all cipher texts' first parts to find out the matching cipher texts. Since all disclosed values of Pt^- are either collision-free (due to the hash function H) and random (according to algorithm **Structured Encryption**), no more than one cipher text matches in each matching process. The found cipher texts should contain the queried keyword, since given a keyword search trapdoor, algorithm **Structured Search** only can disclose the values of Pt^- , which are corresponding to the queried keyword. Formally, we have Theorem 3 on consistency whose proof can be found in Supplemental Materials B.

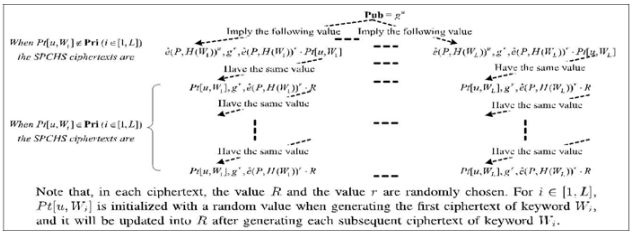

Fig. 2. Hidden star-like structure generated by the above SPCHS instance.

IV. RESULTS AND DISCUSSIONS

The admin and user login is created for generating searchable public key cipher text with hidden structures for fast keyword search. The keyword details along with the keyword search and keyword search system graph is shown in the following figures.

Figure 3: Admin and User Login

Figure 4: User Details

Figure 5: Keyword Details

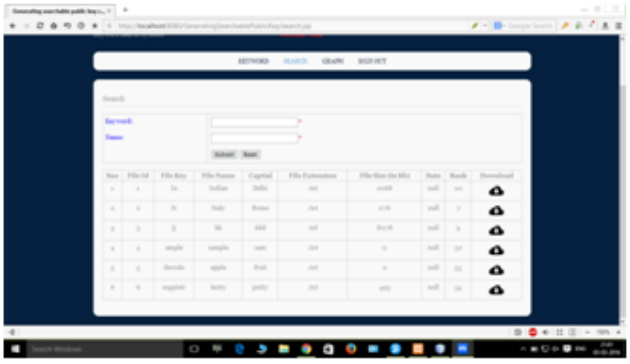

Figure 6: Searching keyword

Figure 7: Keyword Search System Graph

V. CONCLUSION AND FUTURE WORK

Our proposed work deals with the investigation of as-fast-as-possible search in PEKS with semantic security. We proposed the concept of SPCHS as a variant of PEKS. The newly defined concept allows keyword-searchable cipher texts to be generated with a hidden structure. Given a keyword search trapdoor, the proposed search algorithm of SPCHS can disclose part of this hidden structure for guidance on finding out the cipher texts of the queried keyword. Semantic security of SPCHS captures the privacy of the keywords and the invisibility of the hidden structures. We proposed an SPCHS scheme from scratch with semantic security and anonymity in the RO model. The resulting SPCHS scheme can generate keyword-searchable cipher texts with a hidden star-like structure. It has search complexity mainly linear with the exact number of the cipher texts containing the queried keyword. It outperforms existing PEKS schemes with semantic security, whose search complexity is linear with the number of all cipher texts. Moreover, if both the underlying IBKEM and IBE have semantic security and anonymity, the resulting SPCHS is semantically secure. We have identified several interesting properties, i.e., collision-freeness and full-identity malleability in some IBKEM instances, and formalized these properties to build a generic SPCHS construction.

REFERENCES

[1] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in Advances in Cryptology— EUROCRYPT (Lecture Notes in Computer Science), vol. 3027,

C. Cachin and J. L. Camenisch, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 506–522.

[2] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in Advances in Cryptology—CRYPTO (Lecture Notes in Computer Science), vol. 4622, A. Menezes, Ed. Berlin, Germany: Springer-Verlag, 2007, pp. 535–552.

[3] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in Advances in Cryptology— EUROCRYPT (Lecture Notes in Computer Science), vol. 3027, C. Cachin and J. Camenisch, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 223–238.

[4] X. Boyen and B. Waters, "Anonymous hierarchical identity based encryption (without random oracles)," in Advances in Cryptology—CRYPTO (Lecture Notes in Computer Science), vol. 4117, C. Dwork, Ed. Berlin, Germany: Springer-Verlag, 2006, pp. 290–307.

[5] C. Gentry, "Practical identity-based encryption without random oracles," in Advances in Cryptology— EUROCRYPT (Lecture Notes in Computer Science), vol. 4004, S. Vaudenay, Ed. Berlin, Germany: Springer-Verlag, 2006, pp. 445–464.

[6] G. Ateniese and P. Gasti, "Universally anonymous IBE based on the quadratic residuosity assumption," in Topics in Cryptology—CT-RSA (Lecture Notes in Computer Science), vol. 5473, M. Fischlin, Ed. Berlin, Germany: Springer-Verlag, 2009, pp. 32–47.

[7] L. Ducas, "Anonymity from asymmetry: New constructions for anonymous HIBE," in Topics in Cryptology—CT-RSA (Lecture Notes in Computer Science), vol. 5985, J. Pieprzyk, Ed. Berlin, Germany: Springer-Verlag, 2010, pp. 148–164.

[8] M. Abdalla, D. Catalano, and D. Fiore, "Verifiable random functions: Relations to identity-based key encapsulation and new constructions," J. Cryptol., vol. 27, no. 3, pp. 544–593, Jul. 2013.

[9] E. S. V. Freire, D. Hofheinz, K. G. Paterson, and C. Striecks, "Programmable Hash functions in the multilinear setting," in Advances in Cryptology—CRYPTO (Lecture Notes in Computer Science), vol. 8042.

[10] R. Canetti and J. A. Garay, Eds. Berlin, Germany: Springer-Verlag, 2013, pp. 513–530. [10] S. Garg, C. Gentry, and S. Halevi, "Candidate multilinear maps from ideal lattices," in Advances in Cryptology—EUROCRYPT (Lecture Notes in Computer Science), vol. 7881, T. Johansson and P. Q. Nguyen, Eds. Berlin, Germany: Springer-Verlag, 2013, pp. 1–17.

[11] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in Applied Cryptography and Network Security (Lecture Notes in Computer Science), vol. 3531, J. Ioannidis, A. Keromytis, and M. Yung, Eds. Berlin, Germany: Springer-Verlag, 2005, pp. 442–455.

[12] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order preserving symmetric encryption," in Advances in Cryptology— EUROCRYPT (Lecture Notes in Computer Science), vol. 5479, A. Joux, Ed. Berlin, Germany: Springer-Verlag, 2009, pp. 224–241.

[13] F. Bao, R. H. Deng, X. Ding, and Y. Yang, "Private query on encrypted data in multi-user settings," in Information Security Practice and Experience (Lecture Notes in Computer Science), vol. 4991, L. Chen, Y. Mu, and W. Susilo, Eds. Berlin, Germany: Springer-Verlag, 2008, pp. 71–85.

[14] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in Proc. IEEE INFOCOM, Mar. 2010, pp. 1–5.

[15] B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, "Building an encrypted and searchable audit log," in Proc. NDSS, 2004, pp. 5–6.