**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

***IJCSMC, Vol. 4, Issue. 3, March 2015, pg.27 – 34***

**RESEARCH ARTICLE**

# Seclusion Search over Encrypted Data in Cloud Storage Services

**G. Karthika Priya Dharshini[1], D.Viji[2], K.Saravanan[3]**

PG Students, Adhiparasakthi Engineering College, Melmaruvathur, India

karthikapriyadharshini89@gmail.com [1], dviji2k@gmail.com [2], sksarwan234@gmail.com [3]

***Abstract** - Cloud computing is envisioned as the next generation architecture of IT enterprises, providing convenient remote access to data storage and application services. While this outsourced storage model can potentially bring great economical savings for data owners and users, but due to wide concerns of data owners that their private data may be involuntarily exposed or handled by cloud providers. Although end-to-end encryption techniques have been proposed as promising solutions for secure cloud data storage. In this article, we identify the system requirements and challenges towards achieving privacy assured searchable outsourced cloud data services. This paper present a general methodology for this, using searchable encryption techniques, which allows encrypted data to be searched by users without leaking information about the data itself and users queries. The statistical measure approach, i.e., relevance score, from information retrieval to build a secure searchable index, and develop a one-to-many order preserving mapping technique to properly protect those sensitive score information. The resulting design is able to facilitate efficient server side ranking without losing keyword privacy.***

## I. INTRODUCTION

In this emerging computing platform, the cloud provider, application developers, and end users can all get better benefits. One of the most attractive cloud services nowadays is data storage, where end users outsource large volumes of data to cloud servers to enjoy virtually unlimited hardware/software resources and efficient access, without investing a large amount of capital up front for their own data warehousing and maintenance. The fact that data owners and cloud server are no longer in the same trusted domain may put the outsourced unencrypted data at risk the cloud server may leak data information to unauthorized entities [5] or even be hacked [6]. It follows that sensitive data have to be encrypted prior to outsourcing for data privacy and combating unsolicited accesses. An organization that is risk averse enough to avoid the public cloud should be building a secure cloud possibly the company should be building its dream cloud, which contains all the security controls that it thinks are missing from a public environment.

Cloud computing protects data privacy, sensitive cloud data has to be encrypted before being outsourced to the commercial public cloud, which makes effective data utilization service a very difficult task. Traditional searchable encryption techniques allow users to securely search over encrypted data through keywords, they support only Boolean search, which is not sufficient to meet the effective data utilization need that is inherently demanded by large number of users. In this paper, we solve the problem of secure ranked keyword search over encrypted cloud data. Ranked search greatly enhances system usability by enabling search result relevance ranking instead of sending irrelevant results, and further ensures the file retrieval accuracy. Specially, we explore the statistical measure approach, i.e. relevance score, from information retrieval to build a secure searchable index, and also protect those sensitive score information. The resulting design is able to facilitate efficient server-side ranking without losing keyword privacy. Thorough analysis shows that our proposed solution enjoys "as-strong-as-possible" security guarantee compared to previous searchable encryption schemes.

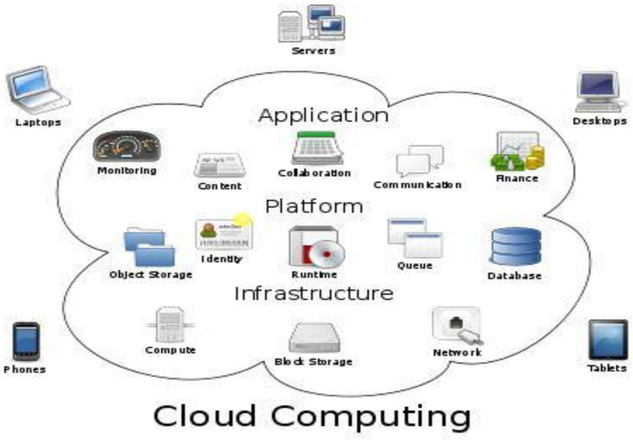

Figure 1. Cloud computing architecture

A privacy approach for owners to take back control of their data is to adopt end-to-end data encryption. Therefore, to build afull-fledged cloud data service, it is highly desirable to enable privacy assured search over encrypted data, which ideally does not leak any sensitive user information to the cloud, such as business secrets or private personal activities. Without being able to effectively utilize the outsourced data, the cloud will merely be a remote storage with limited values.

## II. RELATED WORK

Searchable encryption has also been considered in the public-key setting. Boneh-et-al. presented the first public-key-based searchable encryption. A great disadvantage anyone with the public key can write to the data stored on the server but only authorized users with the private key can search. Li et al proposed fuzzy keyword search on encrypted cloud data.[9] Later, Wang et al very recently explored privacy-assured similarity search mechanism over outsourced cloud data.[11] Following researches on secure ranked search over encrypted data, very recently, Cao et al [10] proposed a privacy-preservingmulti-keyword ranked search scheme, with support of multi-keyword query. They choose the principle of ―coordinate matching, i.e., as many matches as possible, to capture the similarity between a multi-keyword search query and data documents, and later quantitatively formalize the principle by a secure inner product computation mechanism. One disadvantage of the scheme is that cloud server has to linearly traverse the whole index of all the documents for each search request, while ours is as efficient as existing SSE schemes with only constant search cost on cloud server.

We present a general methodology for constructing privacy-assured searchable schemes based on several building blocks, including recently developed efficient symmetric-key encryption primitives (e.g., symmetric searchable encryption [SSE]). For each of the proposed usable search functionalities, we survey recent research advances, and give insights on the advantages and limitations of each approach. Ending with a set of future challenges, this article intends to bring attention to and motivate further research on enabling privacy-assured searchable cloud storage a reality.Secure top-k retrieval from Database Community from database community are the most related work to our proposed RSSE. Theidea of uniformly distributing posting elements using an order-preserving cryptographic function. However, the order-preserving mapping function proposed does not support score dynamics, i.e., any insertion and updates of the scores in the index will result in the posting list completely rebuilt. Zerr-et-al. use a different order-preserving mapping based on pre-sampling and training of the relevance scores to be outsourced, which is not as efficient as our proposed schemes. Besides, when scores following different distributions need to be inserted, their score transformation function still needs to be rebuilt. On the contrary, in our scheme the score dynamics can be gracefully handled, which is an important benefit inherited from the original OPSE.

Encrypted cloud data hosting service involving three different entities as, data owner, data user, and cloud server. Data owner hasa collection of n data files C = {F1, F2, . . . , Fn} that he wants to outsource on the cloud server in encrypted form while still keeping the capability to search through them for effective data utilization reasons. To do so, before outsourcing, data owner will first build a secure searchable index I from a set of m distinct keywords W = {w1, w2, . . . ,wm} extracted from the file collection C, and store both the index I and the encrypted file collection C on the cloud server. Specifically, the following challenging questions are being investigated:[17] 1. Privacy-assured and Effective Cloud Data Utilization: how to enable an encrypted cloud data search service with strong privacy-assurance, while enjoying high service-level performance inherently demanded by the large number of data users and huge amount data files in cloud. Two challenging research tasks are fuzzy and ranked keyword search over encrypted cloud data. 2. Secure Cloud Storage Auditing: how to design efficient data integrity verification mechanisms for strong correctness assurance of cloud data storage service, given the challenge that data is no longer locally possessed by data owners. The design should be publicly auditable, privacy-preserving, and support data dynamics. 3. Scalable and Owner-controlled Cloud Data Sharing: how to enable data owners to reliably and efficiently enforce the dissemination of sensitive cloud data among large number of users in a fine-grained and scalable way, when the data reside in the open untrusted cloud environment. 4. Secure Data Computation Outsourcing: How can a computationally weak end-user securely outsource expensive data?

## III. PRIVACY-ASSURED SEARCHABLE CLOUD STORAGE ARCHITECTURE

### *A. Problem Statement*

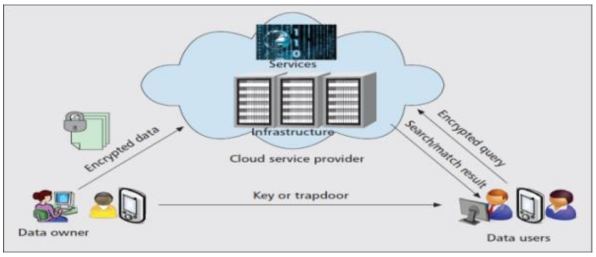

Figure 2. System architecture for searchable cloud data storage services.

We begin by describing a general cloud data storage service architecture involving three (types of) entities (Fig. 1). The *data owner* (or data contributor) is one or multiple entities who generate and encrypt data, and upload them to the cloud server. The owner can be either an organization or an individual. The *cloud server* belonging to a CSP possesses significant storage and computation resources, and provides them to end users in a pay-per-user manner. There are one or more *data users* in the system, who may need to perform queries over the outsourced data in order to extract useful information. In addition, in order to enable public auditing, a third party auditor can be employed, which is discussed in [13] and is outside the scope of this article. The owner's data are encrypted end-to-end using secret keys created by him/herself, and a searchable index is usually created and encrypted along with the outsourced data. To allow data access and search by users, the data owner usually generates and distributes search tokens (or trapdoors), which are encrypted queries to users, either actively or upon users' requests. When a user wants to gain file access or initiate a query, he/she submits a corresponding token to the server, who then returns a matching set of documents in an encrypted format. In some situations, the data user and data owner can be the same physical entity.

Within the scope of this article, we focus on how to enable privacy-assured search for cloud data services. The above system architecture captures a wide range of searchable cloud data storage applications. In some scenarios, the data owner and user can be the same person; for example, Alice uploads her personal albums to Dropbox and wants to search for a particular photo afterward. Or if we consider a corporate data owner, a company may outsource its business records to a cloud server to enjoy the low-cost storage. At the same time, an employee in the auditing department may need to search the business database for records containing sensitive activities. Alternatively, the data owner may be an individual while the user can be a company. For instance, consider a pervasive healthcare application where each patient uploads her health monitoring data periodically to a third party medical service. The latter operates in the cloud, and will provide health reports to the patient by evaluating the patient's data using some flexible diagnostic criteria.

We consider the cloud server to be semi-trusted. This means that most of the time it behaves properly and does not deviate from the protocol, but it will try to find out as much private information in the stored data as possible. This assumption considers data exposure threats from both insiders and outsiders, and is in line with the current technology trend and business model. For insider threats, there may be curious employees inside the CSP who access user data for their own benefit. Thus, the CSP and data owners are not assumed to be in the same trust domain. In addition, some users may also try to access/utilize the data beyond their privileges, either individually or in collusion with each other.

Depending on the information available to the adversary, a basic threat model (known ciphertext, KC) is to assume that the cloud server only possesses the encrypted data and searchable index. In a stronger known background (KB) model, the cloud server may possess some statistical information about the outsourced dataset (e.g., the distributions of term frequency and document frequency). Under this model, the security guarantee is often termed *as-strong-as-possible*.

## IV. SYSTEM REQUIREMENTS

Next, we sketch a set of desirable system design requirements from both the functionality and privacy aspects.

Functional Properties — for data search, perhaps the most important property is *usability*, which is the basis for attracting customers. The current Google search is a great example of what is necessary in plaintext domain search. The following is an (incomplete, but typical) list of them:

- *Multi-keyword search*: The search condition should support Boolean expressions consisting of combinations of multiple keywords, including conjunctive normal form (CNF) and disjunctive normal form (DNF).
- *Result ranking*: The ranked search function greatly enhances the relevance of returned search results and reduces communication overhead, which is highly desirable for building usable cloud data services.
- *Error tolerance*: To accommodate various typos, representation inconsistencies, and so forth, search schemes should have a fuzzy nature. This means a search needs to also return relevant results for keywords within a certain edit distance from the input query.
- *Handle structured data*: A large portion of today's online data is represented using rich structures beyond simple text form, such as social network graphs. Without being able to utilize those structured data, the economic potential of cloud services will not be fully realized.

We note that in the encrypted domain, it is very difficult for the above properties to be simultaneously achieved. We describe how the state-of-the-art schemes achieve some combination of them.

Privacy Assurance —In a searchable cloud storage service, both the owner's outsourced data and users' queries over those data may contain sensitive information and need protection against an adversary. More specifically, the system should meet the following privacy requirements:

- *Data and index confidentiality*: Without the secret key *K*, no one, including the cloud server, should be able to learn sensitive information from the owner's private data. Similarly, they should not be able to deduce sensitive information underlying the data index, because the index is often closely related to the data itself.
- *Query confidentiality*: Users' most important concern is to hide the search criteria on which they are evaluating the data (e.g., their query keywords). These should not be derivable from the search trapdoor and data/index sent to the cloud server, even when the server possesses some additional background information such as keyword distribution.
- A higher-level requirement is *query unlinkabilit*y, that is, the cloud server shall not learn whether two queries have the same criteria. Note that this intrinsically requires the trapdoor to be non-deterministic.

Efficiency — A privacy-assured data search scheme should have low computation, communication, and storage overheads. For such a scheme to be deployed in a large-scale cloud storage system with economic practicality, we argue that the search process should be completed within both constant communication round and computation time (independent of the database size). In general, the privacy guarantee conflicts with efficiency and functionality goals.

## V. TOWARD DESIGNING PRIVACY-ASSURED SEARCH SCHEMES

### *A. Related Techniques*

First, we briefly discuss and compare several existing techniques, and their relevance to the privacy-assuredcloud-based search problem.

- *Secure multiparty computation (SMC)*: In SMC, each party $P_i$ possess some private input $x_i$, and every party computes some (public) function $f(x_1,x_n)$ without revealing $x_i$ to others, except what can be derived from the input and output.
- *Private information retrieval (PIR)*: PIR involves two parties: a client and a server. In asymmetric PIR, the server hosts a public database D, while the client retrieves a record *i*from D without revealing *i*to the server. In symmetric PIR (a.k.a. oblivious transfer), the non-retrieved records should also be withheld from the client, which can be regarded as a special case of SMC.
- *Searchable encryption (SE)*: SE also involves a client and a server, where the latter stores an encrypted database $D^{\tilde{}}$, and the former possesses a private query *Q* that wants to obtain the query result *Q*(D) without revealing both *Q* and plaintext D to the server.

The above also represent different computation models. In the SMC and PIR models, the computation is usually split among the parties, and each party maintains some (private) input. However, in SE, the computation is mainly carried out by the server, and the server does not possess any input. Thus, it can be seen that SE is the technique most relevant to the problem defined in this article.

Symmetric Searchable Encryption — Curtmola*et al.* proposed SSE, which is a deterministic symmetric key encryption scheme with security guarantees under rigorous definitions [6]. The SSE scheme is based on the inverted index and uses pseudorandom functions/per- *Top-down* methodology for designing privacy-assured search schemes.

Methodology —we describe a top-down approach in Fig. 2. Given a search functionality in the plaintext domain, one can decompose it into a certain data index structure and primitive data operations using relevant information retrieval (IR) principles. Then we can try to find a proper encryption scheme to encrypt the data while simultaneously allowing data operations. The second step is nontrivial. Although efficiency could be improved due to the adopted index structure, care needs to be taken to prevent leakage of private information to the cloud server, especially when the server possesses background information on the query statistics. As a result, sometimes the encryption primitive itself may need to be adapted to meet mutations, which makes the search quite efficient. Roughly speaking, the index consists of blinded keywords $f_k(w_i)$ and lists of FIDs containing $w_i$, where $f()$ is a pseudorandom function and $k$ is the secret key. The search trapdoor is also in the same form so that the server can perform matching. However, it only supports single-keyword exact query.

Order-Preserving Symmetric Encryption — In OPSE [2], the numerical ordering of plaintext is preserved after encryption. Boldyreva*et al.* [2] provide the first cryptographic construction of OPSE that is provably secure under the security framework of pseudorandom function or pseudorandom permutation. It can be regarded as a function $g(\Diamond)$ from a domain D = $\{1\ldots M\}$ to a range R = $\{1\ldots N\}$.

1. One-To-ManyOrder-PreservingMapping-OPM

Algorithm

```
procedure OPMK(wi)(Sij(D,R,m, id (F))
while |D|! = 1 do
    {D,R} <--BinarySearch (K,D,R,m);
end while
coin <-$^{R}$--TapeGen(K, (D,R, 1||m, id(F)));
c <--$^{coin}$--R;
return c;
end procedure
procedure BinarySearch(K,D,R,m);
    M <--|D|, N<-- |R|;
    d min(D) - 1, r<-- min(R) – 1;
    y ←r + [N/2];
    coin<-$^{R}$-- TapeGen (K, (D,R, 0||y));
    x<-$^{R}$-- d + HYGEINV(coin,M,N, y - r);
    if m ≤ x then
        D ← {d + 1 . . . x};
        R← {r + 1, . . . , y};
    else
        D ← {x +1 . . .d+M} ;
        R← {y +1 . . . r + N};
    end if
    return {D, R},
end procedure
```

Achieving Secure Ranked Search over Encrypted Data an especially important functionality in plaintext IR is to support ranking mechanisms over search results according to user-specified relevance criteria. Usually, this is achieved by building an inverted index (index structure) and adopting a ranking function to compute the rank of each file relevant to a given search request (primitive data operations are keyword matching and sorting). For example, the following function can be used:

$$Score(Q, F_d) = \sum_{t \in Q} \frac{1}{|F_d|} \cdot (1 + \ln f_{d,t}) \cdot \ln\left(1 + \frac{N}{f_t}\right). \quad (1)$$

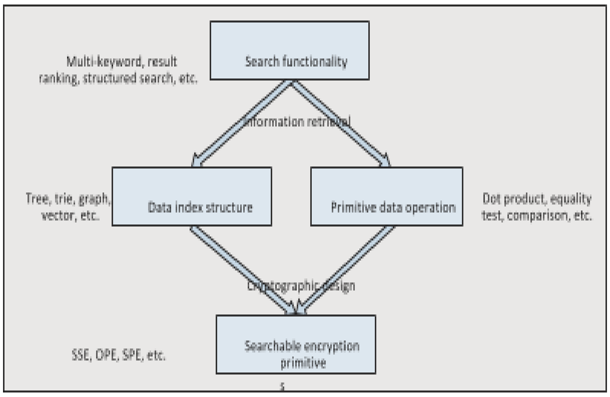
Multi-keyword, result ranking, structured search, etc.
Search functionality
Information retrieval
Tree, trie, graph, vector, etc.
Data index structure
Primitive data operation
Dot product, equality test, comparison, etc.
Cryptographic design
SSE, OPE, SPE, etc.
Searchable encryption primitive
s

Figure 3. Top-down methodology for designing privacy-assured search schemes.

as statistical measurement for evaluating relevance score in the information retrieval community uses the TF IDF rule, where term frequency (TF) is simply the number of times a given term or keyword (we will use them interchangeably hereafter) appears within a file (to measure the importance of the term within the particular file), and inverse document frequency(IDF) is obtained by dividing the number of files in the whole collection by the number of files containing the term (to measure the overall importance of the term within the whole collection). Here, Q denotes the searched keywords; fd;t denotes the TF of term t in file Fd ; ft denotes the number of files that contain term t; N denotes the total number of files in the collection; and $|F_d|$ is the length of file $F_d$, obtained by counting the number of indexed terms, functioning as the normalistion factor.

Toward achieving secure ranked search in the encrypted domain, Zerr*et al.* [16] observed that although the posting list elements (document IDs that contain each keyword in an inverted index) are encrypted, the term frequency distribution for each posting list can still help adversaries re-identify the keyword. Thus, they proposed to transform the relevance scores such that their distribution is uniform for each keyword. They show that this scheme satisfies a definition called *r*- confidentiality in a statistical sense. However, it requires much pre-processing and does not easily handle score dynamics, while the security level is weak.

To improve both the efficiency and privacy, Wang *et al.* proposed a ranked symmetric searchable encryption (RSSE) scheme [12] that enables result ranking for single keyword query. To ensure privacy, a straightforward yet ideally secure RSSE scheme can be derived based on the existing SSE solution in [6], but requires two rounds of interactions between the user and the cloud server, which incurs high communication overhead. Thus, they adopt OPSE [2] to obtain practical performance, where the numerical ordering of the plaintexts is preserved after encryption. Specifically, during the search operation the relevance order (OPSE encrypted relevance scores) of each document is revealed to the server. In this way, efficient relevance score ranking can be done as in the plaintext domain. However, because the original OPSE is a deterministic encryption scheme, this still leaks much information. If the server has some background information on the dataset, such as the distribution of relevance scores for each plaintext keyword, it could reverse engineer the keyword.

To break this determinacy, the authors propose one-tomanyorder-preserving mapping (OPM), which maps the same relevance score to different encrypted values. They incorporate the unique file IDs together with the plaintext as the random seed in the final ciphertext chosen process in OPSE. Thus, the same plaintext will no longer be deterministically assigned to the same ciphertext, but instead to a random value within the randomly assigned bucket in a range R. Furthermore, they use different keys to encrypt the relevance score for different posting lists (documents containing each keyword) to make the OPM more indistinguishable.

Therefore, we have to modify the OPSE to suit our purpose. In order to reduce the amount of information leakage from the deterministic property, an one-to-many OPSE scheme is thus desired, which can flatten or obfuscate the original relevance score distribution, increase its randomness, and still preserve the plaintext order. To do so, we first briefly review the encryption process of original deterministic OPSE, where a plaintext m in domain D is always mapped to the same random-sizednon-overlapping interval bucket in range R, determined by a keyed binary search over the range R and the result of a random HGD sampling function. A ciphertext c is then chosen within the bucket by using m as the seed for some random selection function. Our one-to-manyorder-preserving mapping employs the random plaintext-to-bucket mapping of OPSE, but incorporates the unique file IDs together with the plaintext m as the random seed in the final ciphertext chosen process. Due to the use of unique file ID as part of random selection seed, the same plaintext m will no longer be deterministically assigned to the same ciphertext c, but instead a random value within the randomly assigned bucket in range R. The whole process is shown in Algorithm 1, adapted from [14].TapGen is the random coin generator and HYGEINV is the he efficient function implemented in Mat lab as our instance for the HGDð_Þ sampling function. The correctness of our one-to-manyorder-preserving mapping follows directly from the Algorithm 1. Note that our rational is to use the OPSE block cipher as a tool for different application scenarios and achieve better security, which is suggested by and consistent with [14]. Now, if we denote OPMas our one-to-manyorder-preserving mapping function with parameter: OPM:{0.1}l{0.1}log|D|→{0.1}log|R|,our proposed RSSE scheme can be described as follows: In the setup phase

The RSSE scheme achieves data and index privacy, because the relevance scores in the searchable index are encrypted using OPSE with OPM. The highly flattened one-to-many mapping and the fully randomized score-to-bucket assignment in OPSE makes it difficult for an adversary to predict the original plaintext score distribution by observing the ciphertext. In addition, this scheme hides the search keyword from the adversary. But since the trapdoor is deterministic, it does not provide query unlinkability. For efficiency, the encrypted index generation and search operations can both be finished within seconds for 1000 documents.

The above method cannot directly handle multiple-keyword ranked search, because the order of OPSE's ciphertext will not be preserved for the sum of multiple relevance scores. To support secure multi-keyword ranked search over encrypted data (MRSE), Cao *et al.* [3] proposed to adopt another similarity measure from the IR community, *coordinate matching*, which captures the relevance of documents to a query through the number of query keywords appearing in a document. Each document index and the query are described as a binary vector (index structure), respectively, such that the similarity is measured by the dot product of the two vectors (primitive data operation). In order to protect the index privacy and search privacy, one shall encrypt the index and query vectors, and compute the similarity score over ciphertexts.

To this end, they propose a secure inner-product computation mechanism that adapts the SPE scheme originally used for secure *k*-nearestneighbour (kNN) query in [15]. Basically, the search operation should compute the dot product between a query vector $^{Æ}q$and each data (index) vector $p^{Æ}_{i}$. However, a straightforward application of SPE is not secure as it linearly preserves the dot product, by which the server can statistically analyze similarity scores for two queries differing in one keyword to learn that keyword (called *scale analysis attack*), especially in the KB model. Therefore, to build a secure MRSE that preserves search privacy, they obfuscate the document frequency to diminish the chances for re-identification of keywords. In particular, they propose to add randomness to both the data vector and query vector in order to blind the exact similarity score from the server. The randomness is added on the fly, by extending both vectors with dummy random keywords.

In the MRSE scheme, the data and index privacy are achieved since the encryption algorithm is secure in the KC model. In addition, under the KB model *query confidentiality* is achieved as well as trapdoor unlinkability. It introduces nearly constant search overhead with the increase of keywords; In contrast, in other multiple-keyword search schemes this is linear.

### *B. Trade-offs and Impacts of Design Choices*

In designing privacy-assured search schemes, it is generally perceived that efficiency vs. privacy is an intrinsic trade-off; That is, to achieve higher efficiency, the privacy level will be sacrificed. However, when access pattern privacy is acceptable to leak, higher efficiency may be gained without decreasing privacy, which depends on all the factors including the IR method, index structure, and encryption primitive. For instance, in [14] the use of a trie instead of a list to encode the fuzzy keywords dramatically enhances the efficiency, while the scheme still remains provably secure due to the security of SSE. On the other hand, in [3, 4], the trade-off is mainly between accuracy and privacy; That is, the privacy level decreases if higher result ranking accuracy is desired.

## VI. FUTURE CHALLENGES

There are many interesting research issues worth further investigation. The works mentioned above have a common characteristic: they relax the privacy guarantees (i.e., "as strong as possible") to achieve higher efficiency performance. While there are formal privacy definitions for searchable encryption that reveal the access pattern [6], for as-strong-aspossible schemes, how to formally analyze the privacy level given various known background information remains an interesting and important open problem. Addressing it may require tools from information theory and statistics. Differential privacy [7] is a useful formal privacy measure, but it only applies to statistical queries. It is necessary to develop similar privacy metrics for ranked search.

For multi-keyword ranked search, it is desirable to enable advanced relevance criteria such as the ones commonly used in IR. The problem is how to hide the sum of multiple keywords' relevance scores from the server, which may possess statistical distribution information to re-identify the search keywords. A possible approach would be to use a similar procedure as in MRSE to build randomized document indexes and query vectors.

We investigate future enhancement of our ranked search mechanism including the reversibility of our proposed one-to-many order-preserving mapping technique.

In addition, for wider applicability in different scenarios, can we make public-key-based searchable encryption more practical and secure? We studied privacy enhancements of public-key-based multidimensional queries in [10], while its efficiency is still well behind symmetric-key-based solutions. Finally, it is also interesting to ask if more complex data utilization functions can be efficiently evaluated on encrypted data; For example, running database join/merge queries, or graph algorithms on structured data.

## VII. CONCLUDING REMARKS

In this article, we identify the problem and challenges of enabling privacy-assured searchable cloud data storage services. Recent research advances in this field are surveyed, which suggest that achieving functionally rich, usable, and efficient search on encrypted data is possible without sacrificing privacy guarantee too much. The steady evolution of this field will need to bring expertise from the cryptography, database, and information retrieval communities.

## REFERENCES

[1] S. Zerr et al., “Zerber+r: Top-k Retrieval from a Confidential Index,” Proc. EDBT ’09, 2009.

[2] A. Boldyreva et al., “Order-Preserving Symmetric Encryption,” Proc. Eurocrypt ’09, LNCS, vol. 5479, Springer, 2009.

[3] N. Cao et al., “Privacy-PreservingMulti-Keyword Ranked Search Over Encrypted Cloud Data,” IEEE INFOCOM, 2011, pp. 829–37.

[4] N. Cao et al., “Privacy-Preserving Query over Encrypted Graph-Structured Data in Cloud Computing,” 31st Int’l. Conf. Distributed Computing Systems, 2011, pp. 393–402.

[5] M. Chase and S. Kamara, “Structured Encryption and Controlled Disclosure,” Advances in Cryptology-ASIACRYPT 2010, 2010, pp. 577–94.

[6] R. Curtmola et al., “Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions,” Proc. ACM CCS ’06, 2006.

[7] C. Dwork, “Differential Privacy,” Automata, Languages and Programming, 2006, pp. 1–12.

[8] S. Kamara and K. Lauter, “Cryptographic Cloud Storage,” Proc. 14th Int’l. Conf. Financial Cryptography and Data Security, Berlin, Heidelberg, 2010, pp. 136–49.

[9] J. Li et al., “Fuzzy Keyword Search over Encrypted Data in Cloud Computing,” Proc. IEEE INFOCOM ’10 MiniConf. San Diego, CA, Mar. 2010.

[10] M. Li et al., “Authorized Private Keyword Search over Encrypted Data in Cloud Computing,” 31st Int’l. Conf. Distributed Computing Systems, 2011, pp. 383–92.

[11] D. Song, D. Wagner, and A. Perrig, “Practical Techniques for Searches on Encrypted Data,” Proc. IEEE S & P 2000, 2000.

[12] C. Wang et al., “Secure Ranked Keyword Search Over Encrypted Cloud Data,” Proc. ICDCS ’10, 2010.

[13] C. Wang et al., “Toward Publicly Auditable Secure Cloud Data Storage Services,” IEEE Network, vol. 24, no. 4, July–Aug. 2010, pp. 19–24.

[14] C. Wang et al., “Achieving Usable and Privacy-Assured Similarity Search over Outsourced Cloud Data,” Proc. IEEE INFOCOM ’12, Orlando, FL, Mar. 2012.

[15] W. K. Wong et al., “Secure KNN Computation on Encrypted Databases,” Proc. SIGMOD, 2009.

[16] M. Armbrust et al., “Above the Clouds: A Berkeley View of Cloud Computing,” Feb 2009.

[17] “Challenges toward Achieving Privacy and Secure Searchable Outsource cloud data Storage Services”, IJARCSSE