Inverted Index Based Ranked Keyword Search in Multi-user Searchable Encryption

Manju S. Nair[1](✉) and M. S. Rajasree[2]

[1] College of Engineering, Trivandrum, India
manjusnair8@gmail.com
[2] Government Engineering College, Barton Hill, Trivandrum, India
rajasree40@gmail.com

Abstract. Cloud storage is a promising technology for large scale data storage. Searchable encryption protocols are developed to securely store and efficiently retrieve encrypted data stored in cloud without revealing any sensitive information. As data grows faster, even search schemes requiring linear time are not feasible to handle big data. We propose a sub-linear search scheme using inverted index supporting fine grained search control to multiple users. The documents are ranked to return the most relevant results and to reduce the communication cost. The scheme hides the document-ids and access patten from the cloud provider and proves the security.

Keywords: Searchable encryption · Inverted index · Blind storage

1 Introduction

Big data analytic offers significant advantages in several application areas such as medical research, climate change predictions, scientific research and many more. Cloud storage together with encryption of data is the most viable solution to tackle the growing need of storage requirement, and at the same time preserving the security and confidentiality of the sensitive data. However, encrypted data looses all its functionality. Searchable encryption schemes [2,4,11] support search on encrypted data without revealing any information about the search query or the encrypted data. Inverted index based search are very efficient especially in big data environment as it directly give the search results rather than having the search complexity linearly varying with the number of documents. However, the downside is that the length of each list, the access pattern and search pattern can reveal additional information to the curious cloud provider. Using deterministic trapdoor also leaks search pattern. Oblivious RAM [7] based solution can hide the search and access pattern. However, the high computational cost and storage requirement make it less suitable for practical applications. Homomorphic encryption schemes [6,14] that support computations to performed on encrypted

https://doi.org/10.1007/978-981-13-5826-5_43

data and producing encrypted results is considered as the most viable solution. However, due to the high storage overhead and computational cost these schemes are not proven to be practical as yet. The proposed system hides document-ids and length of each list in the inverted index. In addition, the access and search pattern are also hidden using the blind storage primitive [10]. The scheme also support legitimate users to selectively retrieve data without using trusted third parties or interacting with the data owner.

2 Problem Statement

Fast and secure scheme to upload encrypted data to the cloud by the data owner and provide fine-grained search control to legitimate users without revealing any information such as the number of files, file names, access pattern or search pattern to the cloud provider. We discuss the system model and the security goals of the proposed system.

2.1 System Model

The four types of entities in the system are

1. *Owner*: Uploads the encrypted documents and index to the cloud. An access control list is created to enable legitimate users access the documents.
2. *Cloud Provider*: The cloud provider is assumed to honest but curious. The cloud provider follows the protocol correctly in retrieving documents but may try to extract as much information as possible.
3. *Users*: Each user has access to different set of documents. A legitimate users generate trapdoors for the keyword to be searched using his secret key.
4. *Documents*: Documents are stored in the blind storage. Each file is split into a number of blocks and the encrypted blocks are stored in random positions following the blind storage primitive.

2.2 Security Goals

1. *Query Privacy*: Refers to the amount of information leakage to the cloud provider during user queries. Access pattern and search pattern are also hidden from the cloud provider.
2. *Query Unforgeability*: Ensures that only legitimate user can generate a valid query and no other user including the cloud provider can generate a valid query on behalf of a legitimate user.
3. *Controlled Information Disclosure to the Legitimate Users*: Ensures revealing only those permitted documents containing the search keyword to a legitimate user.

3 Related Work

Curtmola et al. [5] proposed an inverted index based symmetric searchable encryption scheme with sub-linear search complexity. They put forward the notion of adaptive security where a curious cloud server choose their queries as function of previously obtained trapdoors and search outcome and try to extract more information as possible. They extend the scheme to support multiple users to access the data using broadcast encryption. Wang et al. [12] proposed an efficient inverted index based searchable encryption scheme. The scheme supports conjunctive queries and is secure against trapdoor linkability. Paillier additive homomorphic scheme together with the use of blind storage protects the trapdoor and access pattern. However, the users need to interact with the owner for generating the trapdoor and for decrypting the results obtained.

Dynamic searchable encryption scheme proposed by Kamara et al. [8] uses an encrypted inverted index and has sub linear search complexity. The system is adaptively secure against chosen keyword attack. The scheme reveals search pattern to the cloud provider. Wang et.al's [13] scheme support ranked keyword search using order preserving symmetric encryption. Term frequency and the length of the file is used to rank the documents. The inverted index is encrypted using order preserving encryption so that cloud provider can return the most relevant documents to the user. However the search pattern is revealed to the cloud provider.

Even though inverted index based schemes provides sub-linear search time the tradeoff is between efficiency and security. In most of the schemes the deterministic trapdoor generated and the search pattern revealed to cloud provider can lead to statistical attacks on the data stored. We propose an inverted index based search hiding the search and access pattern.

4 Preliminaries

4.1 Bilinear Maps

Definition 1 (Bilinear Pairing). A bilinear pairing [3] is a map $e : G_1 \times G_2 \mapsto G_T$ with G_1, G_2 being cyclic groups of prime order P. g_1 and g_2 are the generators of G_1, G_2 respectively and the following conditions hold.

1. Bilinearity: $e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab} = e({g_1}^b, {g_2}^a) \forall a, b \in Z_p^*$.
2. Non-degeneracy: $e(g_1, g_2) \neq 1_{G_T}$ i.e, $e(g_1, g_2)$ generates G_T.
3. Computability: $e(P, Q)$ is efficiently computable.leng

4.2 Blind Storage

Blind storage proposed by Naveed et al. [10] enable a client to store files in the cloud storage without revealing the names of files, their sizes and the number of files.

1. Keygen: A key K_ϕ for the pseudo random generator ϕ and a key K_{ID} for the full-domain pseudo random function ψ are generated.
2. Build: List of files F are stored in an array D with n_D blocks of m_D bits each as follows.
   Each file $f = (id_f, data_f)$ in F requiring $size_f$ blocks of storage is stored as
   (a) Generate a seed $\sigma_f = \psi_{K_{ID}}(id_f)$ for the pseudo random generator Γ
   (b) A pseudo random sequence $\wedge = \Gamma(\sigma_f)$ is generated.
   (c) Write the $size_f$ blocks of $data_f$ and $H(id_f)$ onto the first $size_f$ free blocks in D indexed by integers from $\wedge$.
   (d) Encrypt each block of D using the pseudo random generator ϕ and the key K_ϕ
3. Access: Using id_f generate the seed $\sigma_f = \psi_{K_{ID}}(id_f)$ and the pseudo random sequence $\wedge = \Gamma(\sigma_f)$ to access the blocks of file id_f.

5 Our Contribution

In this paper we provide a novel, secure and efficient protocol for providing fine-grained search control to a set of users. The major contributions are

– Fast and secure method to selectively retrieve data.
– Ranking of documents is incorporated to ensure that the most relevant documents are returned.
– No interaction with owner of the documents or trusted third parties are required for trapdoor generation.

6 The Proposed Method

The proposed system hides the document-ids in inverted list. Corresponding to each document-id, p duplicate-ids are created by applying the collision resistant hash function H_3 on document-id concatenated with an integer number. Instead of storing the document-id directly on the inverted list we use any one of the encrypted hash values. Hence for the same document, the value stored will be different in different lists and are indistinguishable. In addition a random number of dummy document-ids are created and are hashed and inserted in the list, thus hiding the actual length of each list. An array is created for each user to provide access control. An entry in this array indexed by the hash function H_3 on duplicate document-id and is set as either with the document-id or null value based on whether he can access that document. All entries corresponding to the dummy names are set to null. This array is encrypted using the public key of the user. Search returns the encrypted document-ids and access control array to the user. User identifies the document-id from access control array and generates the seed to retrive files from the blind storage.

7 Concrete Construction

The concrete construction of the frame work discussed above is as follows. We also assumed that every user in the group has a public/private key pair.

1. Setup (1^n): We use ElGamal Elliptic Curve cryptography for group to generate public/private key pairs. An asymmetric Type 2 bilinear map groups is used in encrypting the keywords. The cloud server generate the following public parameters.
   - Public parameters for ElGamal Elliptic Curve cryptography.
   - Owner generates his private/public key pair o and g^o.
   - Each user usr_i in the set generates the private/public key pair u_i and g^{u_i}.
   - Initialize a bilinear map group:$(p, G_1, G_2, G_t, e, g_1, g_2, g_t)$
2. Build (D, W): The data owner creates an inverted index structure. For each word w_i in the dictionary a list is created to contain the document-ids of all documents containing the keyword w_i on the order of relevance. We use the most prominently used ranking technique $TF \times IDF$ (1) to compute the relevance score of query to a document.

$$RScore(d, w) = \frac{1 + ln(f_{d,w})}{|F_d|} \tag{1}$$

   The inherent problem with an inverted index is that the length of each list and the unique hash value generated for each document-id in the inverted index can covey some information to the curious cloud provider leading to a statistical attack. In order to counteract the statistical attack we use the following techniques.
   (a) t random document-ids are created and added to the list to hide the length of list.
   (b) Corresponding to each document in the set, p dummy names are used produce different hash values. We use the hash of document-id in the inverted index, for the same document different hash values can be produced and this can prevent the statistical attack to a greater extend.
   (c) Each keyword in the dictionary is encrypted as

$$H_2(e(H_1(w_j), g_2)^r)$$

   the Algorithm: 1 shows the detailed construction.
3. Encrypt (D): Algorithm executed by the cloud provider to store data in the blind storage. Using the document-id the the data owner stores the data in the blind storage as discussed in Subsect. 4.2.
4. Userkeygen (1^n): The algorithm is executed by the user. The user generates the trapdoor generation key h_i using his private key u_i and owner's public key o as follows.

$$p = g^{ou_i} \tag{2}$$

$$h_i = H_1(x||y) \tag{3}$$

   p is a point on elliptic curve the x coordinate and y coordinate are concatenated and hashed to a point in G_1

5. Ckeygen (U): Algorithm executed by the owner to generate complementary key for each user. The owner using his private key o and user's public key g^{u_i} compute h_i using (2) and (3). The complementary key for user usr_i is created as
$$h'_i = {g_2}^{r/h_i}$$
6. $S_i \leftarrow$ Delegate(U): To provide controlled information disclosure, an array S_i is created to store the access control list(ACL) for each user usr_i. For each document d_i and the corresponding p dummy names, an entry in the array is created to contain either the document-id or a null value based on whether the user can access that document or not. For all dummy document-id's dmy_i the corresponding entries the access control list are set to null. ACL is encrypted using the public key of that user and is uploaded to the cloud. So this array contains $p \times |D| + t$ entries. The algorithm: 2 shows the detailed construction.
7. $tr \leftarrow Trapdoor(w_i)$: User generates the trapdoor for keyword w_i using his secret key h_i.
$$tr = H1(w_i)^{h_i}$$
8. $L, S_i \leftarrow Search(tr, usr_i)$: The cloud provider use the complementary key h'_i to convert user specific trapdoor.
$$\begin{aligned} x &= H2(e(tr, h'_i)) \\ &= H2(e(H1(w_i)^{h_i}, {g_2}^{r/h_i})) \\ &= H2(e(H1(w_i), g_2)^r) \end{aligned}$$
Return the list pointed by $T[x]$ and the S_i to the user.
9. Retrieve (L, S_i): User decrypts the entries in S_i using his private key. L contains the hashed document-ids in the sorted order of relevance. Using the hash value as index the user either retrieves the document-id or null value from S_i based on the permission granted. These document-ids can be used to fetch files from the blind storage. Using the document-id user generates the seed and the pseudo random sequence for accessing data from the blind storage. The blocks can be accessed randomly, and after decryption the target files can be identified from the document-id stored in each block.

8 Security Analysis

We analyze the security of our system in this section

1. Query Privacy: The security notion of query privacy aims in reducing the information leakage that can occur during query processing. In the proposed system the query is encrypted using user's secret key only a legitimate user and owner can generate this key. An owner delegate the access rights to search a document by providing the complementary key. Even though the trapdoor generated for the same keyword by multiple users are different, the index specific trapdoor generated by the cloud provider using the complementary key is deterministic. However, the scheme hides search and access pattern.

Algorithm 1. Algorithm for building Inverted Index

```
1: Derive all distinct keywords $W = \{w_1, w_2 .........w_n\}$ from the document set $D$.
2: For all $w_i \in W$generate Doc($w_i$), the list of documents containing keyword $w_i$
   sorted on the order of relevance of $w_i$.
3: For each document $d_i \in D$ create AuthUser($d_i$), the set of authorized users
   who can access document $d_i$
4: For each document $d_i \in D$ create $p$ dummy names as $d_i||1, d_i||2, ........d_i||p$
5: Create $t$ distinct dummy document-ids as $dmy_i, i \in 1, 2....t$.
   Create an array $T$ of dimension $|W|$.
   for each $w_j \in W$, Create a linked list $L_j$ do
      Compute $x = H_2(e(H_1(w_j), g_2)^r)$
      for each $(d_k \in Doc(w_j)$ do
         Choose a random number $r_1 \in \{1, 2...p\}$
         Add $H_3(d_k||r_1)$ to $L_j$;
      end for
      insert $r_2$ dummy document-ids at random positions to the list $L$
      $T[x]$ =address of list $L_j$
   end for
```

Algorithm 2. Algorithm for building Access Control List

```
for each $(u_i \in U)$ do
   Create an array $S_i$
   for each $(d_j \in D)$ do
      for all $r_1 \in \{1, 2...p\}$ do
         $ind = H_3(d_j||r_1)$
         if $u_i \in$ AuthUser$(d_j)$ then
            $S_i[ind] = d_j$
         else
            $S_i[ind] = Null$
         end if
      end for
   end for
   for each (dummy document-id $dmy_i$) do
      $ind = H_3(dmy_i)$
      $S_i[ind] = Null$
   end for
end for
encrypt each $S_i[ind]$ using the public key of $u_i$.
```

2. Query Unforgeability: A legitimate user generates the secret key g^{ou_i} using owner's public key g^o and his private key u_i. We implemented the system using ElGamal elliptic curve cryptography. The x and y coordinates of g^{ou_i} is concatenated to form a string and hashed to value in Z_p as t=$H(g^{ou_i})$. The user use this value t to create trapdoor. By Computational Diffie-Hellman assumption no other user can compute g^{ou_i} from g^o and g^{u_i} to forge the trapdoor.

3. Controlled Information Disclosure: In the access control array created by the owner, only those document-ids that are accessible to the user are stored and all other entries are kept null. A legitimate user use this document-id for retrieving a document from the cloud storage. Hence the user can access only the permitted documents.

9 Performance Analysis

Pycharm community IDE, Charm Crypto 0.43 [1] framework and the Pairing-based cryptographic library [9] are used for implementation on an Intel core i7-3770 CPU @ 3.4 GHz and 8 GB RAM. For providing search control, we used Elliptic curve group prime192v2 to generate the public parameters for secret key generation. The time taken for the trapdoor generation is nearly 0.00092 s and the pairing operation takes nearly 0.0044s. As the trapdoor directly gives the starting address of the list containing the document-ids, the search complexity is $\mathcal{O}(1)$.

Compared with Wang et al.'s [12] scheme our scheme provide ranked results and no interaction with the owner is required for generating the trapdoor and decrypting the results. Even though Wang et.al's [13] scheme support ranking of documents the scheme reveals search and access pattern to the cloud provider.

10 Conclusion

The proposed system support multiple users to selectively retrieve data without using trusted third parties. Fast and efficient retrieval of data is ensured using inverted index and by ranking the documents. The inherent disadvantage of revealing search and access pattern in inverted index based search are overcome using blind storage. The system ensures fine grained search control to multiple users with out using shared keys. A legitimate user will get only the permitted set document-ids even when the search query is present in several other documents.

References

1. Akinyele, J.A., et al.: Charm: a framework for rapidly prototyping cryptosystems. J. Cryptogr. Eng. **3**(2), 111–128 (2013). https://doi.org/10.1007/s13389-013-0057-3
2. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_30
3. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. SIAM J. Comput. **32**(3), 586–615 (2003)
4. Chang, Y.-C., Mitzenmacher, M.: Privacy preserving keyword searches on remote encrypted data. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 442–455. Springer, Heidelberg (2005). https://doi.org/10.1007/11496137_30

5. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. J. Comput. Secur. **19**(5), 895–934 (2011)
6. Gentry, C., Halevi, S.: Implementing gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_9
7. Goldreich, O., Ostrovsky, R.: Software protection and simulation on oblivious rams. J. ACM (JACM) **43**(3), 431–473 (1996)
8. Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, pp. 965–976. ACM (2012)
9. Lynn, B.: PBC Library (2006). http://crypto.stanford.edu/pbc
10. Naveed, M., Prabhakaran, M., Gunter, C., et al.: Dynamic searchable encryption via blind storage. In: 2014 IEEE Symposium on Security and Privacy (SP), pp. 639–654. IEEE (2014)
11. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: Proceedings of the 2000 IEEE Symposium on Security and Privacy, S&P 2000, pp. 44–55. IEEE (2000)
12. Wang, B., Song, W., Lou, W., Hou, Y.T.: Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee. In: 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 2092–2100. IEEE (2015)
13. Wang, C., Cao, N., Li, J., Ren, K., Lou, W.: Secure ranked keyword search over encrypted cloud data. In: 2010 IEEE 30th International Conference on Distributed Computing Systems (ICDCS), pp. 253–262. IEEE (2010)
14. Wu, D.J.: Fully homomorphic encryption: cryptography's holy grail. XRDS Crossroads ACM Mag. Students **21**(3), 24–29 (2015)