Privacy Preserving Multikeyword Searching Over Encrypted Data

Vijaya Sayaji Chavan¹, Dr. M. Z. Shaikh², Mithun Mhatre³

¹,²,³Bharati Vidyapeeth College of Engineering, Kharghar, Navi Mumbai, India

Abstract— As so much advantage of cloud computing, most of the data owners centralize their sensitive data into the cloud. With a mass of data files stored in the cloud server, it is important to provide keyword based search service to data user. However, in order to protect the data privacy, sensitive data is usually encrypted before outsourced to the cloud server, which makes the search technologies on plaintext unusable. The system preserves the high search efficiency inherited from the inverted index while lifting the one-time-only search limitation of the previous solutions which simultaneously meets a set of strict privacy requirements. A major challenge exposed from the existing efforts is the difficulty to protect user’s query privacy so this challenge is faced and tried to remove in this scheme.

I. INTRODUCTION

Privacy and security are the most important issues in cloud computing. To achieve high flexibility and to reduce cost, many data owners are outsourcing their data management system to public cloud. However, data utilization, e.g. keyword search, is a challenging problem due to the data encryption. Downloading the entire encrypted data set first then searching over the decrypted data is difficult task. Therefore, the search operation must be done at the cloud side and over the encrypted data. First of all, the keyword privacy is compromised once a keyword is searched. As a result, the index must be rebuilt for the keyword once it has been searched. Obviously, such a solution is counterproductive. Secondly, the existing inverted index based searchable schemes do not support conjunctive multi-keyword search, which is the most common form of queries nowadays. So this solution solves the problem of building a search-able encryption scheme based on the inverted index to over-come the above limitations. This scheme proposes a practical inverted index based public-key searchable encryption scheme. This overcomes the one-time-only search limitation in the existing schemes. This scheme supports conjunctive multi-keyword search using only one trapdoor while the existing invert index based searchable encryption schemes only support single keyword search. A probabilistic trapdoor generation algorithm is used to break the trapdoor link ability. So it preserves the index and trapdoor privacy.

To provide stronger security guarantee, this scheme uses an efficient oblivious transfer protocol to hide the access pattern. Comparing with the existing public-key searchable encryption schemes which use expensive pairing operations, this scheme is more efficient because it only need multiplication and exponentiation.

II. LITERATURE SURVEY

Generalized INverted InDeX (Ginix)[1], presents index structure, which merges consecutive IDs in inverted lists into intervals to save storage space. With this index structure, more efficient algorithms can be devised to perform basic keyword search operations, i.e., the union and the intersection operations, by taking the advantage of intervals. Specifically, these algorithms do not require conversions from interval lists back to ID lists. As a result, keyword search using Ginix can be more efficient than those using traditional inverted indices. The performance of Ginix is also improved by reordering the documents in datasets using two scalable algorithms.

Table 1 sample dataset of 7 paper titles.

<table>
<thead>
<tr>
<th>ID</th>
<th>Dataset content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Keyword querying and ranking in databases</td>
</tr>
<tr>
<td>2</td>
<td>Keyword searching and browsing in databases</td>
</tr>
<tr>
<td>3</td>
<td>Keyword search in relational databases</td>
</tr>
<tr>
<td>4</td>
<td>Efficient fuzzy type-ahead search</td>
</tr>
<tr>
<td>5</td>
<td>Navigation system for product search</td>
</tr>
<tr>
<td>6</td>
<td>Keyword search on spatial databases</td>
</tr>
<tr>
<td>7</td>
<td>Searching for hidden-web databases</td>
</tr>
</tbody>
</table>
Though typical searchable coding schemes permit a user to firmly search quite encrypted knowledge from facet to facet keywords and by selection retrieve files of interest, these techniques support solely actual keyword search. In this paper they tried to solve the problem of realistic fuzzy keyword search over encrypted cloud knowledge where as maintaining keyword privacy. Fuzzy keyword search greatly enhances system usability by returning the matching files once users’ looking inputs precisely match the predefined keywords or the nearest doable matching files supported keyword similarity linguistics, once actual match fails. In this resolution, tendency to exploit edit distance to quantify keywords similarity and extremely developed techniques on constructing fuzzy keyword sets, which understand optimized storage and demonstration overheads. This paper have a propensity to boot propose a novel symbol-based trie-traverse looking scheme, where a multi way tree structure is constructed positive persecution symbols remodel post from the resulted fuzzy keyword sets. Through exact security analysis, we have a leaning to show that our designed resolution is secure and privacy preserving, where as properly realizing the goal of fuzzy keyword search. Exhaustive experimental outcome expose the energy of the planned resolution.

### III. METHODOLOGY

#### System Model:

The system model in this work is shown in Fig.1. There are three entities in the system, a cloud server, a data owner and multiple users. The data owner generates the encrypted index and outsources it along with the encrypted data into the cloud. An authorized user submits a query request to the server in the form of a trapdoor which he gets from the data owner through a secure channel. After receiving the trapdoor, the cloud server matches the encrypted index with the trapdoor. Finally, the cloud server returns the matching documents as the search result. The access control between data owner and the users can be achieved using existing protocols.
Index privacy:

The index privacy is twofold. First, the cloud server should not learn the content of the index since the content of the index directly reflects the content of the documents. Second, the cloud server should deduce no information about the document through analyzing the encrypted index. Such information includes 1) whether a document contains certain keyword(s), and 2) whether different documents contain a common keyword.

Trapdoor privacy:

A trapdoor is generated for each query request to allow the cloud server to search over the encrypted index. Intuitively, the trapdoor contains the query information but in an encrypted form. Given a trapdoor, the cloud server should learn nothing about the user’s query from it. We consider the protection of the following information for trapdoor privacy: the content of the query, the number of the keywords in the query, and the fact that whether the same query has been searched before.

Access pattern:

Access pattern refers to the accessed documents, i.e. the search results. The adversary could further deduce the private information of the index and the trapdoor from the access pattern. To avoid such leakage, the search results of queries must be indistinguishable from each other.

Searchable Encryption Scheme:

A searchable Encryption Scheme uses the following notations through the rest of the topic:

\[ \Sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n) \] is a finite set of document collection, where \( \sigma_i \) is the ID of the ith document.

\[ \Omega = (w_1, w_2, \ldots, w_m) \] is a finite set of keyword collection from \( \Sigma \), which is denoted as dictionary.

\[ I = (I_{w_1}, I_{w_2}, \ldots, I_{w_m}) \] is an inverted index for the document set \( \Sigma \). Each \( I_{wi} \) is a list which contains

\[ \Sigma_i = (\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{i\ell}) \] where \( wi \in \Sigma, i \leq j \leq p \)

\[ I \] is the encrypted searchable index based on \( I \).

\( Q \subseteq \Omega \) is a query request which is a subset of the dictionary.

\( TQ \) is the trapdoor for the query \( Q \).

[\( n \)] means an integer set from 1 to n.

[\( S \)] refers to the cardinality of \( S \) which can be a set, a list or a vector.

Setup(\( k \)):

The data owner first chooses two k-bit prime numbers \( p, q \) such that \( \gcd(pq, (p-1)(q-1)) = 1 \).

Then the data owner follows the key generation process in Fig.1 to generate the key pair for the Paillier algorithm, i.e. \( pk = (n, g), sk = (\lambda, \mu) \). The data owner keeps \( sk \), a pseudorandom permutation (PRP) \( f \) and an invertible matrix \( M \) as the master key \( MK \). The degree of \( M \) is determined by the size of the dictionary \( m \). At last, the data owner publishes the public key to the cloud server.

IndexGen(\( MK, I \)):

IndexGen(\( MK, I \)) takes as input the master key \( MK \) and an index \( I \) for the document set \( \Sigma \). It outputs the encrypted searchable index \( I \).

TrapdoorGen(\( MK, Q \)):

It takes \( e \) as input the master key \( MK \) and query \( Q \). It outputs the trapdoor \( TQ \) for the query.

Query(\( I, TQ \)):

It takes as input the the encrypted index \( I \) and trapdoor \( TQ \). It outputs \( R \subseteq \Sigma \) as the search result.

IV. CONCLUSION & FUTURE WORK

This system proposed a privacy preserving multikwesword search over encrypted data on inverted index. This scheme overcomes the one-time-only search limitation in the previous schemes used in Ginix[1]. The probabilistic trapdoor generation algorithm prevents the cloud server from linking the trapdoors. This scheme also hides the number of keywords in the query. Additionally, this scheme supports multi-keywords conjunctive search. This scheme uses the blind storage technique to protect the access patterns. As compared with another schemes which uses the pairing operations that are expensive but this only uses the multiplications and exponentiations. This system also be useful for the large documents.

REFERENCES


