iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** VI **Month of publication:** June 2018

DOI: http://doi.org/10.22214/ijraset.2018.6102

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VI, June 2018- Available at www.ijraset.com

A Survey on Various Search Techniques over Encrypted Data on Cloud Storage

Karan Kumar Singh[1], Kushal Lohana[2], Anmol Adukia[3], Supreetha S [4]
[1, 2, 3]*U.G Students, Department of CS&E, Sapthagiri College o Engineering, Chikkasandra,Bangalore,India*
[4]*Assistant Professor, Department of CS&E, Sapthagiri College of Engineering, Chikkasandra,Bangalore,India*

Abstract: ***Cloud computing has generated much interest in the recent years for its many advantages, but has also raise many security and privacy concerns. Storage and access of confidential documents have been identified as a major problem in the area. Many researchers investigated solutions to search over encrypted documents stored on remote cloud servers. Day to day usage of cloud has attracted attackers to break data security in the cloud data storage system. Security of cloud data is ensured by means of cryptographic keys which when exposed facilitates the attackers access the confidential data. In this paper, we present a Survey on various search technique along with the security issues raised by these technique on encrypted data.***

Index Terms*: Cloud, Storage, privacy and security*

I. INTRODUCTION

In recent times organizations and individuals adopt cloud technologies, many have become aware of the serious concerns regarding security and privacy of accessing personal and confidential information over the Internet. In particular, the recent and continuing data breaches highlight the need for more secure cloud storage systems. It is generally agreed that encryption is necessary, cloud providers often perform the encryption and maintain the private keys instead of the data owners. That is, the cloud can read any data it desired, providing no privacy to its users. The storage of private keys and encrypted data by the cloud provider is also problematic in case of data breach. Hence, researchers have actively been exploring solutions for secure storage on private and public clouds where private keys remain in the hands of data owners.

We describe modifications to the scheme to lower storagecost at a small cost in response time and to defend against cloud providers with statistical knowledge on stored data Although phrase searches are processed independently using our technique, they are typically a specialized function in a keyword search scheme, where the primary function is to provide conjunctive keyword searches.

II. BACKGROUND

Author in [1] work on Encrypted keyword search scheme based on public key encryption was among the most cited in the area. The author considered a scenario where a user wishes to have an email server verify messages associated with certain keywords without revealing the content of the emails. As sample application, the scheme would allow an urgent encrypted email to be flagged to the attention of a user while others sent to appropriate folders. The proposed solution uses identity based encryption and a variant using bilinear mapping. Authors in [2] proposes a novel approach to allow for phrase searching and query proximity ranking for search queries on encrypted data in the cloud. Neither the source documents, nor the search index database needs to be hosted on local or trusted servers. Both of them will be encrypted and hosted in remote public cloud servers.. Most proposed methods utilize advanced mathematical structures such as Bloom filters or trap-doors but they typically only allow for Boolean searches and do not support phrase searching. “Proximity ranked searching” implicitly ranks documents by a function, f, that is directly proportional to the distance the keywords in the search phrase appear from each other. In addition, the author implements search querying techniques presented in previous research such as Boolean searching and multi-keyword ranking. To allow for proximity ranking, the location information of keywords must be preserved in the encrypted index created from the data corpus. This is a challenge that previous research relying on Bloom filters cannot easily overcome. Instead of using a highly compressible index, such as a Bloom filter, we make use of a relational database to store the three valuable components to each document: document reference, keywords, and keyword locations. A main attack point on the proposed scheme thus far is that it is highly susceptible to statistical frequency analysis attacks. If each keyword were encrypted individually using a deterministic encryption method, a nosey cloud provider could compare the encrypted index with a language probability table to estimate which words map to which encrypted words. To combat this problem we truncate the encrypted words to a predefined number of bits β.

Therefore, numerous collisions are created since the entire encrypted keyword space size has been reduced to 2^{β} Authors in [3] proposed a provable secure-phrase-search scheme with symmetric searchable symmetric encryption, while maintaining a moderate computational cost in searching and at a moderate communication cost between the client and server, both linear in the phrase size. Additionally, the author proves that our construction satisfies non-adaptive security, the latest security definition proposed by R. Curtmola. It is well known that searching by phrases over plaintext enjoys a wide range of applications. In many cases the ability of searching by a single keyword or a Boolean combination of keywords over encrypted documents is good enough. However, oftentimes, it can be more efficient to search by phrases instead of discrete keywords in many cases. Most existing proposals for multi- keyword search fail to offer the ability to judge whether or not the keywords occurring in the document are consecutive. The reason being that most secure indexes are established based on the distinct word sets contained within the documents. To search for a phrase, two rounds of communication between the client and server are needed for each query. During the first round of interaction, the server performs the search over the index of the binary matrix with the trapdoor sent from the client and returns the document candidates which contain every word of the phrase. In the second interaction, another trapdoor will be generated and sent from the client to help the server test whether each document candidate contains all words occurring as a phrase. The scheme achieves non- adaptive security under the security definition we gave in section 4. But it is not the strongest, at least it does not meet the criteria of adaptive security, which is raised by R. Curtmola. Authors in [4] present a conjunctive keyword and phrase search scheme with low storage requirement. The scheme is capable of basic ranking and has the ability to search for non- indexed keywords. The key difference between conjunctive keyword search and phrase search is that, in addition to containing the requested keywords, they must also appear contiguously in the specified order in the document The scheme provides the basic ability to rank the returned results during the final step of the phrase matching process: The number of matched phrases per document can be tracked and used to rank the returned results. Though not discussed here, further ranking capability can be added by incorporating TF- IDF. The scheme relies on false positives to defend against statistical analysis The proposed solution uses truncation of encrypted keywords to generate false positives in query results to hide the true search terms. As a result, part of the index must be stored client-side and the search is also performed by the client rather than the cloud. The use of false positives to provide security can be unreliable since the number of false positives associated to individual search terms is random. Unlike in Zittrower's scheme, the technique allows the cloud to perform the majority of the computations. In terms of communication, the scheme includes returning irrelevant results since normalization requires insertion of random data into the index tables, although much less than in Zittrower's scheme. The algorithm provided in section although efficient, is vulnerable to statistical analysis. Namely, the number of entries for each word in a document is not hidden. Authors in [5] presents a low storage solution for encrypted phrase search. The scheme is capable of basic ranking and can be adapted to search for non-indexed keywords and defend against Inclusion-Relation attacks. The scheme requires two Bloom filters per document, one for mapping keywords to document and one for determining keyword location. The two sets of filters require N(bk + bl) bits of storage on the cloud server, where bk is the size of a conjunctive keyword Bloom filter, bl is the size of a keyword location Bloom filter and N is the number of documents in the corpus. The data owner needs only to store cryptography keys. The communication cost also compares favourably with existing schemes. The phrase search scheme based on Bloom filter that achieves 8 times lower storage cost in our experiment than the existing solutions while exhibiting similar or better communication and computational requirements. The proposed solution provides basic ranking capability, can be adapted to non-keyword search and is suitable against inclusion-relation attacks. The main difference between conjunctive keyword search and phrase search is that the queried keywords must appear contiguously in the specified order in addition to all being present in the document. At last Authors in [6] proposes a scheme which achieves a much faster response time than existing solutions. The scheme is also scalable, where documents can easily be removed and added to the corpus. The Author also describe modifications to the scheme to lower storage cost at a small cost in response time and to defend against cloud providers with statistical knowledge on stored data. To overcome the various issues faced in various search scheme, The Author in [6] proposed phrase search scheme based on bloom filters

A. Phrase Search Scheme Based On Bloom Filters

In a keyword search scheme, Bloom filters can be used to test whether a keyword is associated with a document. Many existing phrase search schemes use a keyword-to-document index and a location/chain index to map keywords to documents and match phrases. We describe an alternative approach using Bloom filters to support this functionality with an emphasis on response time. Our scheme can be summarized as the use of multiple n-gram Bloom filters, Bn Di, to provide conjunctive keyword search and phrase search. Bastion achieves this by combining the use of standard encryption functions with an efficient linear transform. In

this sense, Bastion shares similarities with the notion of all-or- nothing transform. An AONT is not an encryption by itself, but can be used as a pre-processing step before encrypting the data with a block cipher. This encryption paradigm—called AON encryption— was mainly intended to slow down brute-force attacks on the encryption key. However, AON encryption can also preserve data confidentiality in case the encryption key is exposed, as long as the adversary has access to at most all but one cipher text blocks Bloom filters are space-efficient probabilistic data structure used to test whether an element is a member of a set. A Bloom filter contains m bits, where k hash functions, Hi(x), are used to map elements to the m-bits in the filter. The Bloom filter is initially set to all zeros. To add an element, a, to the filter, we compute Hi(a) for i = 1 to k, and set the corresponding positions in the filter to 1. For example, for k = 2 and m = 7, to add 'Bell' to the filter, we compute H1(Bell) = 2 and H2(Bell) = 5. Setting the position 2 and 5, the Bloom filter becomes 0; 1; 0; 0; 1; 0; 0. To test for membership of an element, b, in a sample Bloom filter, we compute Hi(b) for i = 1 to k, the element is determined to be a member if all corresponding positions of the sample Bloom filter is set to 1. For example, 'Bell' would be a member of the Bloom filter, 0; 1; 1; 0; 0; 1; 1.

B. Inclusion-Relation Attacks

Cai et al. described an inclusion-relation (IR) attack, where two sets of query results, a and b, with a being a subset of b, would imply the queried keywords that led to b includes keywords that led to a. A cloud server with some knowledge on the statistical properties of the search terms and has access to sets of trapdoors and their search results can potentially discover some of the keywords. Technique can be adapted to defend against such attacks by allowing a set of keywords to map to many possible queries (trapdoors) and the inclusion of false positives in search results. Also, since different sets of keywords can have the same bits in the Bloom filter being set to 1, different Keyword sets can lead to the same query. Figure 1 shows different mappings of keywords to trapdoors and search results for various conjunctive keyword search schemes.

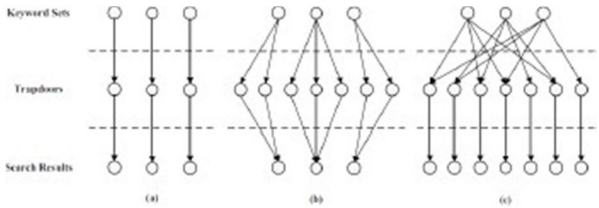


Fig-1: Relationship between keywords, trapdoors and search Results

C. Phrase Search Protocol

To provide phrase search capability, each document is parsed for lists of keyword pairs and triples. For example, 'Happy Day, Happy Night' would yield the pairs, 'Happy Day', 'Day Happy' and 'Happy Night', and the triples, 'Happy Day Happy' and 'Day Happy Night'. A keyed hash for each keyword pair is computed, Hkp (kwj|kwj+1), and passed into k hash functions and the result is used to set k bits in the Bloom filter, B^2Di. Keyword triples are similarly hashed to generate the Bloom filter, B^3Di. The resulting Bloom filters for pairs and triples are organized into matrices with the first rows containing the filters B^xD1 for the first document. The matrices are then transposed to produce the pairs and triples Bloom filter indexes, IBF^2 and IBF^3, which are stored alongside the encrypted documents on the cloud. To perform a phrase search for the keyword sequence, kw0= {kw1,kw2...kwq}, the data owner must first perform the Bloom filter hash computation of the pair, Hkp(kw1|kw2), to determine the set bits in the queryfilter if the phrase contains two keywords. If the phrase contains more than two keywords, the hashes of triples within the phrase,

Hkp(kwj|kwj+1|kwj+2)

Where j=1 to q− 2, are evaluated instead. These bit locations are sent to the server, who then computes T = IBF2 q1&IBF2, q2...&IBF2,qx, where IBF2,qi is the qth i row in IBF2 if the phrase contains two keywords, and similarly using IBF3 for longer phrases. The set bits in T identify the matched documents. That is, for each set bit index, i, in T, the following is true:

$\{Hkp(kw1|kw2)\} \in B^2$ Di (1) For pairs and

$\{Hkp(kwj|kwj+1|kwj+2)\} \in B^3$Di, where j= 1 to q−2, (2) For triples.

Once the matches are identified, the cloud server returns the matched document identifiers or the encrypted documents depending on the application requirements. Our phrase search scheme requires only 2 messages to be sent:

The initial message to the cloud server containing the set bit locations of the query Bloom filter T for pairs or triples and

The response to the data owner containing the query results from the phrase search performed locally by the cloud. Performing

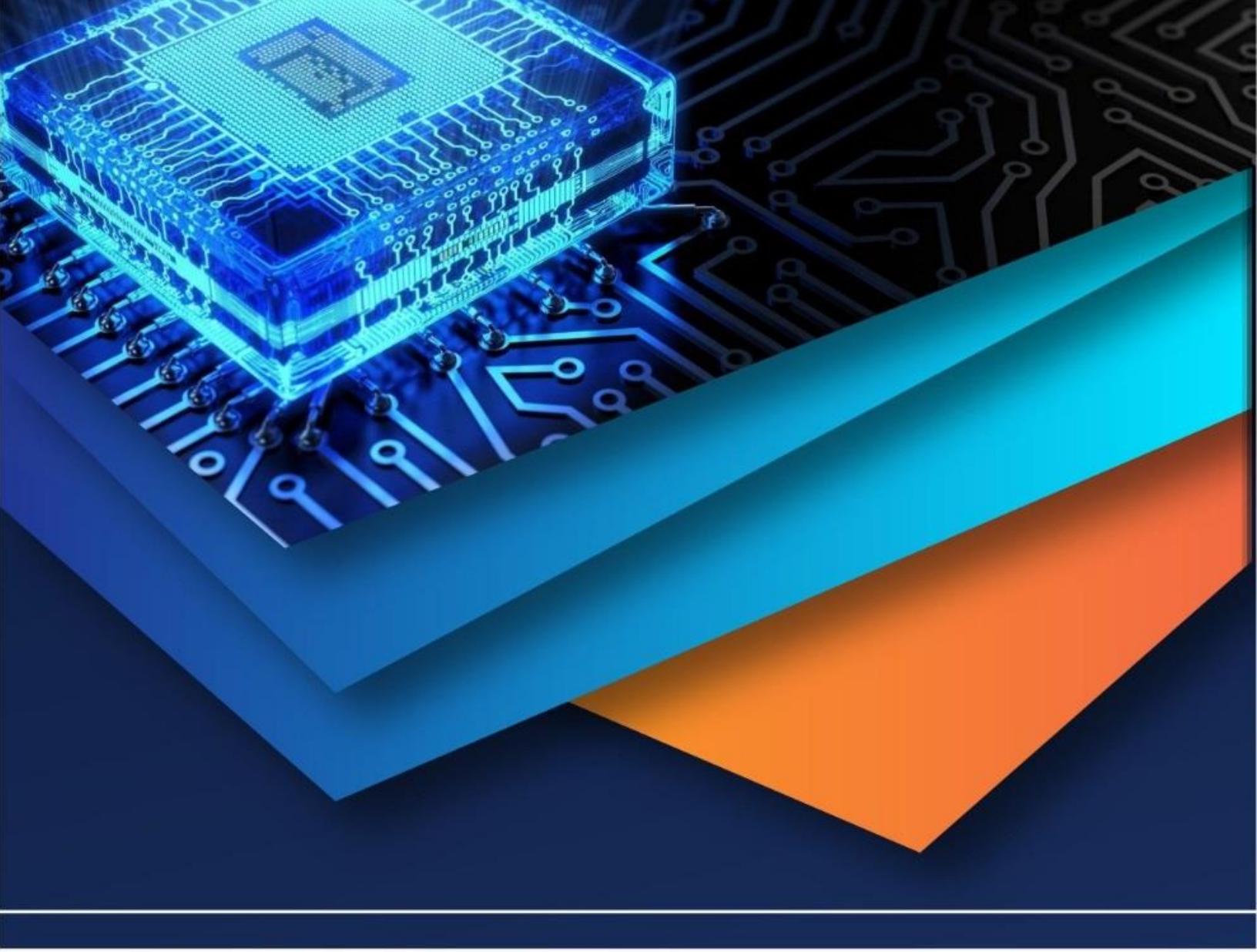
thephrase search requires k(q − 2) hash computations for phrases of length q > 2 and a simple bit-wise AND operations. The protocol is computationallyefficient. Its performance is dependent on the length of the phrase and largely independent of the size of the document set. Due to the space efficiency of Bloom filters, our scheme also requires less storage than index based schemes. Since filters are assigned per document, adding or removing documents consists simply of adding or removing the associated filters, providing a scalable solution. While a document containing a phrase will always be correctly identified as such, our scheme can falsely identify documents as containing a phrase when it doesn't. The source of the false positive is not only the natural property of Bloom filter, but also in how a phrase match is determined. If a user queries n-grams for n = 2 or n = 3, our scheme has no false positives other than ones arising from the use of Bloom filters. For n > 3, however, it is possible that keyword triples within a phrase appear in different parts of a document without the complete phrase being present. Using the previous example of 'Happy Day Happy Night', a false positive would occur if a document does not contain the phrase but instead contains 'Happy Day Happy Day' and 'Snowy Day Happy Night'. The validity of the scheme is based on low occurrence of such scenarios in practical settings.

III. CONCLUSION

After doing a survey on various search technique, the scheme used by authors in [6] overcomes the various issues faced in previous techniques. The author in[6] uses phrase search scheme based on Bloom filter that is significantly faster than existing approaches, requiring only a single round of communication and Bloom filter verifications. The solution addresses the high computational cost noted in [5] by reformulating phrase search as n-gram verification rather than a location search or a sequential chain verification. This approach is also the first to effectively allow phrase search to run independently without first performing a conjunctive keyword search to identify candidate documents. it also achieves a lower storage cost than all existing solutions except [5], where a higher computational cost was exchanged in favour of lower storage. While exhibiting similar communication cost to leading existing solutions, the proposed solution can also be adjusted to achieve maximum speed or high speed with a reasonable storage cost depending on the application.

REFERENCES

[1] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in In proceedings of Eurocrypt, 2004, pp. 506–522.
[2] S. Zittrower and C. C. Zou, "Encrypted phrase searching in the cloud," in IEEE Global Communications Conference, 2012, pp. 764–770
[3] Y.Tang,D.Gu,N.Ding,andH.Lu,"Phrase search over encrypted data with symmetric encryption scheme," in International Conference on Distributed Computing Systems Workshops,2012,pp.471–480.
[4] Poon and A. Miri, "An efficient conjunctive keyword and phrase search scheme for encrypted cloud storage systems," in IEEE International Conference on Cloud Computing, 2015.
[5] , "A low storage phrase search scheme based on bloom filters for encrypted cloud services," to appear in IEEE International Conference on Cyber Security and Cloud Computing, 2015.
[6] Fast Phrase Search for Encrypted Cloud Storag Hoi Ting Poon, Member, IEEE, and Ali Miri, Member, IEEE transaction paper,2017



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY