ISSN (Print): 0974-6846
Indian Journal of Science and Technology, *Vol 11(48), DOI: 10.17485/ijst/2018/v11i48/137381, December 2018* ISSN (Online): 0974-5645

An Efficient Phrase Search using Bloom Filters over Cloud Data

S. K. Shaheena Begum* and Prathi Lakshmi Venkata Pavani Sujitha

Department of CSE, Visvodaya Engineering College, JNTUA, Kavali – 524201, Andhra Pradesh, India; Shaheena.vits@gmail.com, sujithaprathi509@gmail.com

Abstract

Objective: To study the efficient phrase search technique on the basis of Bloom filters that is considerably faster compared to existing solutions. **Methods/Statistical Analysis**: Nowadays, keyword search is very popular technique to retrieve the data from clouds, in which the keyword can be processed over encrypted data and retrieve most of irrelevant results. Many authors investigate the different searching techniques to retrieve sensitive information from the clouds. But few of them only focus on conjunctive keyword search. Hence, in this paper we present An Efficient Phrase Search technique using Bloom filters that are very fast compared with existing techniques and our scheme reduce storage and computational overheads. **Findings**: To evaluate our scheme we collect the set of 1500 documents and these documents are pre-processed to avoid headers and footers to decrease the distorted in the statistics of the given dataset. Then we apply the Bloom filters technique to improve the performance in our scheme. Finally, our scheme achieves lower storage and computational overheads rather than existing techniques. **Applications/Improvements**: Our proposed technique explores a series of n-gram filters to uphold the functionality. The scheme manifests a trade-off between storage and false positive rate.

Keywords: Bloom Filters, Cloud Computing, Multiple Keyword Search, Phrase Search

1. Introduction

As associations and people actualize cloud advances, many have turned out to be aware of the genuine concern in regards to security and privacy of accessing private information over the Internet. To solve this problem generally encryption is required and cloud providers use the full encryption and protect the private keys of owners. While encrypting the private keys by cloud providers will arises new security issue like information leakage. Consequently, specialists have been seeking after arrangements with the end goal to give secure storage on private and public clouds where private keys kept up by the owners.

In[1] presented one of the most primitive works for searching of keyword. Here scheme explores public key encryption technique to make keywords viable to search without enlightening data. In[2] explored the predicament to search over the audit logs which are encrypted.

The majority of the customary searches pointed on single keyword search. Recently, specialists have anticipated arrangements on conjunctive keyword search, which contains complex catchphrases[3,4]. Other fascinating quandaries, for example, giving positions for list items[5–7] and looking with catchphrases named fuzzy keyword search through that may contain blunders[8,9], have likewise been estimated. The capacity of expression looking was additionally recently investigated[10–15].

In this study, we are showing a phrase search plot which accomplishes quicker reaction time contrasted with ordinary arrangements. Likewise the plan is extensible and added to the corpus where reports can without much of a stretch be expelled. We additionally shown changes to the plan to diminish the storage cost at a little expense accordingly time and to ensure against cloud providers with measurable information on confidential information.

**Author for correspondence*

2. Research Method

2.1 Communication Framework

We'll explicate our keyword search framework using two parties: The data owner and an untrusted cloud server. Our algorithms will be implemented easily to the scenario of an organization wishing to setup a cloud server for its representatives by conveying a proxy server instead of the owner and having the workers/clients approve to the proxy server. A standard keyword search protocol is depicted in Figure 1. During setup, the required encode keys for hashing and encode tasks are created by owner. Then, all the documents which are in database are searched for keywords. Bloom filters and n-grams are tied to hashed keywords. The files are transferred to the cloud server that is symmetrically encoded. The owner in the wake of breaking down the documents as in setup transfers them with Bloom filters appended to the cloud server with the end goal to transfer records into the database. To seize a document, the owner needs to send the demand to the cloud server that removes the file along with the attached Bloom filters. The search to be carried out, the data owner computes and sends the queried keywords with a trapdoor encryption to the cloud to set off a etiquette for the requested keywords to search in the corpus. Ultimately, the cloud provides the requested documents with identifiers.

Our scheme changes from a portion of the regular works[1,2]. Here keywords for the most part contain

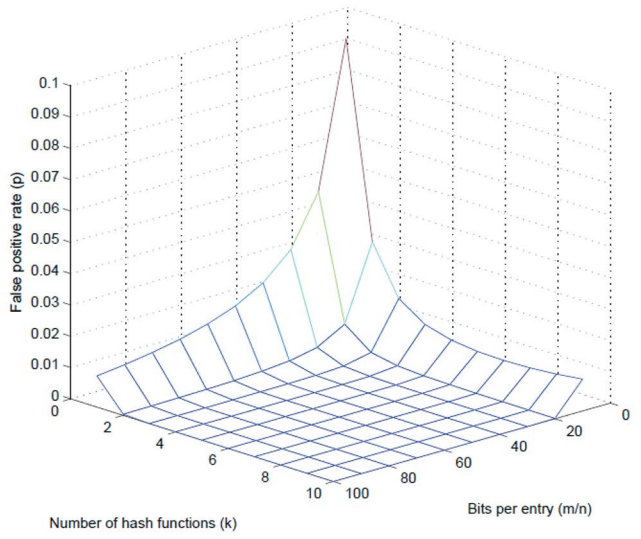

Figure 1. False positive rate (p) as a function of the number of hash function (k) and bits per entry (m/n).

meta-information rather than the substance of the documents and where a trustworthy key escrow authority is misused because of the use of identity-based encryption. Exceptional to recently works[10], our setup is practically identical to an association where it looked to re-appropriate registering assets to a distributed storage provider along these lines permitting a scan for its representatives and like[7] where the point is to return precisely ranked documents. The other late works which concern encoded information search have estimated comparative models[11].

Note that, based on the application, the encrypted documents may or may not need recovery once the query is determined. Further privacy issues may occur. These issues were considered in impervious storage and private information retrieval schemes.

The protocol leading to the query resolution will mainly limited by our discussions. Direct retrieval is supposed where suitable to use rather than prevailing solutions to perform phrase search.

In the view of security, we imagine a semi-honest cloud server, which is involved in learning about stored data but it follow our keyword based search protocol as mentioned and will not alter or misrepresent any data for the sake of acquiring an advantage. Two of the main security issues with respect to the keyword searches are the privacy of the document sets and the confidentiality of the queried keywords. In brief, a secure keyword based search protocol could avert the cloud server from getting non-negligible amount of information from the stored documents or the keywords based on the query requests.

Note that, users are employees of the data owner's organization in our target application, who can exploit the data set to search for any documents in the data set. An application necessitates that users be constrained from accessing certain files, an access control system would be required to secure the matched results and return only those which the user has the requisite credential to access.

2.2 Bloom Filters

Researchers have misused the Bloom filters to achieve conjunctive keyword search to reduce storage cost and give security as false positives. Blossom filters are space-proficient probabilistic information structures. These are utilized to scrutinize whether a component is an individual from a set. A Bloom filter contains m bits, where k hash capacities, $H_i(x)$ are used to map elements to the m-bits in the filter. The Bloom filter is set to each of the

zeros at first. To entirety up a component, a, to the filter, we figure $Hi(a)$ for i= 1 to k, and make the respective positions to 1 in the filter. For example, for $k = 2$ and $m = 7$, to add 'Bell' to the filter, we enumerate $H_1(Bell) = 2$ and $H_2(Bell) = 5$. Setting the location 2 and 5, the Bloom filter turn out to be 0; 1; 0; 0; 1; 0; 0. To analyze the sponsorship of an element, b, in an example Bloom filter, we determine $H_i(b)$ for i = 1 to k, on the off chance that all the individual places of the Bloom filter is embedded to 1 the component is specified to be a part.

Bloom filters can erroneously break down a component as an individual from a set, as they have no false-negatives. Given k hash functions, n things embedded and m bits which are utilized in the filter, the likelihood of false positives is near $p = \left(1-e^{-kn/m}\right)^k$ and least amount false positive rate is accomplished when $k=\frac{m}{n}ln2$.

2.3 Fast Phrase Search based on Bloom Filters

In a keyword search conspire, Bloom filters can be utilized to break down either a keyword is identified with a record or not. Many existing expression look plans utilize a keyword to-archive list and an area/chain record to delineate to documents and match phrases. To support this usefulness a methodology utilizing Bloom filters outlined with noticeable quality on reaction time. Our plan can be pruned as the abuse of numerous n-gram Bloom filters, $B^n_{D_i}$, to give conjunctive keyword search and phrase search.

2.3.1 Conjunctive Keyword Search Protocol

To give conjunctive keyword search capacity, each document, D_i, is dissected for a rundown of keyword kw_j. A Bloom filter of size m is made to zeros. Every keyword is hashed utilizing a secret key to create $H_{kc}(kw_j)$ and g one into k Bloom filter hash functions to set k bits in the Bloom filters. These outcomes in *1*-gram Bloom channel for all archives: $B^1_{D_i}=\{b_1,b_2,\ldots,b_m\}$ where $b_i\in\{0,1\}$. Other than the Bloom filters, the archive documents, $D=\{D_1,D_2,\ldots,D_n\}$, is encoded and gets transferred to the server of the cloud . The main line of the Bloom filters contains the filter $B^1_{D_1}$ for the first document and the last line contains $B^1_{D_N}$ which are then organized into a matrix. Its transpose is put away as a Bloom filter list I_{BF} where each line important to a bit in the Bloom filters. Note that the ith row in I_{BF} contains data on which document's filter has its i^t it set. This game plan enables us to quickly distinguish the documents for a particular query by working just with bits that are set.

The owner plays out the Bloom filter hash calculation to execute a conjunctive keyword search for an arrangement of keywords $kw^{'}=\{kw_1,kw_2\ldots kw_q\}$ and to verify the set of bit locations, $Q=\{q_1,q_2,\ldots,q_x\}$, that would be set in the query filter and sends them to the server, where server at that point processes $T=I_{BF,q_1}\ \&\ I_{BF,q_2}\ldots\&\ I_{BF,q_x}$ where I_{BF,q_i} is the q_i^{th} qth i row in I_{BF}. The indexes of bits that are set in T are perceived as the coordinated documents. The cloud server will restore the document identifiers which are coordinated or the encoded documents dependent on the application requirements once the relating matches are perceived. Note that the span of the set Q is lesser than m, A conjunctive keyword Bloom filter contains every one of the keywords in a document as the filter of query includes just a couple of keywords. In this manner, this strategy can recognize the documents that are coordinated, performing fewer activities as opposed to singular filter check.

2.4 Phrase Search Protocol

To give phrase search ability, all the records are parsed for arrangements of keyword combines and triples. A keyed hash for every keyword pair is figured, $H_{k_p}(kw_j\,|\,kw_{j+1})$, also, go into k hash functions and in addition to set k bits in the Bloom filter $B^2_{D_i}$, by utilizing the outcome. Keyword triples are correspondingly hashed to count the Bloom filter, $B^3_{D_i}$.

The following Bloom filters for sets and triples are organized into matrices with the main lines containing the filters $B^x_{D_1}$ for the main document. The frameworks are then transposed to result the sets and triples Bloom filter lists I_{BF^2}, and I_{BF^3}, which are store close by the encoded reports on the cloud.

With the end goal to play out an phrase search for the keyword sequence, $kw^{'}=\{kw_1,kw_2\ldots kw_q\}$, the information owner should initially execute the Bloom filter hash calculation of the pair, $H_{k_p}(kw_1\,|\,kw_2)$, to identify the set-bits in the question filter if the expression contains two keywords. In the phrase that the expression contains in excess of two keywords, the hashes of triples within the expression, $H_{k_p}(kw_1|kw_{j+1}|kw_{j+2})$ where $j=1toq-2$, are assessed. The set bit areas are sent to the server, who at that point calculates $T=I_{BF^2,q_1}\ \&\ I_{BF^2,q_2}\ldots\&\ I_{BF^2,q_x}$, where I_{BF^2,q_i} is the q_i^{th} row in I_{BF^2} in the phrase that the expression contains two keywords and similarly utilizing I_{BF^3} for longer expressions. The set bits in T perceive the coordinated records. That is, for each set bit file, i, in T, the following is true:

$$\left\{H_{k_p}\left(kw_1 \mid kw_2\right)\right\} \in B_{D_i}^2 \quad (1)$$

For pairs and

$$\left\{H_{k_p}\left(kw_j \left| kw_{j+1} \right| kw_{j+2}\right)\right\} \in B_{D_i}^2, where\ j = 1toq-2, \quad (2)$$

For triples.

When the matches are known, the cloud server restores the archive identifiers which are coordinated or the encoded reports based on the application imperatives. Our phrase search conspire needs just 2 messages to be sent: 1. The underlying message to the cloud server comprises of the set bit areas of the question Bloom filter T for sets or triples and 2. The reaction to the information–provider containing the inquiry results from the phrase search performed locally by the cloud. Achieving the phrase search requires $k(q - 2)$ hash calculations for phrases of length $q > 2$ furthermore, a simple Bitwise AND tasks. The convention is computationally capable. Its execution is dependent on the phrase length and generally autonomous of the extent of the report set. As a result of the space proficiency of Bloom filters, our plan additionally requires less capacity than file based plans. Since filters are dispensed per document, addition or cancellation of archives comprises essentially of summation or removal of the related filters, giving an adaptable solution.

While a document contains a phrase will effectively perceived in that capacity, our plan can wrongly distinguish documents as containing a phrase when it doesn't. The source of the false - positive isn't just the normal property of Bloom filter, yet in addition in how a phrase matches is resolved. When a client questions n-grams for $n = 2$ *or* $n = 3$, our plan has no false positives other than ones stimulated by utilizing Bloom filters. For $n > 3$, likely the keyword triples inside a phrase show up in different parts of an archive without the entire phrase being available.

3. Results and Analysis

To assess our outcomes against existing phrase search plans, we specify the calculation on a corpus comprising of 1500 reports made accessible by Project Gutenberg. The archives were pre-prepared to dispense with headers and footers, which incorporate copyright, contact and source data to diminish skewing in the insights relating to the informational index. Stop words are likewise discarded. To count the factual properties of the corpus and the effectiveness for the distinctive plans, the Natural Language Toolkit was utilized. The structure of Bloom filters is noteworthy to our plan's effectiveness. Particularly, the utilization of a Bloom filter record requires the channels to be of the comparative length. Keep in mind that the condition for false-positive rate as pursues:

$$p = \left(1 - e^{-k\frac{n}{m}}\right)^k \quad (3)$$

Figure 1 indicates false-positive rates somewhere in the range of 1% and 10% similar to the quantity of hash functions and the quantity of bits required per passage. A little filter estimate is practically identical as far as capacity. A low false-positive rate would reduce correspondence and computational expense. Particularly, the execution time will be altogether enhanced by utilizing few hash functions, since the computational expense is similar to the quantity of hash work utilized. Particularly, the quantity of hash capacities, k, expected to decrease false-positive rate isn't regularly utilized since there is extremely slight enhancement in false positive rate as we can heighten the quantity of hash functions past a specific limit. Utilizing a solitary hash work, k = 1, would reduce the computational expense, yet in addition dramatically increases the capacity cost to achieve the equivalent false positive rate. The high inconsistency in false positive rate when k = 1 can likewise be troublesome for corpus with high change in record sizes. As portrayed in the Figure 2, the quantity of bits per section and the false positive rate is genuinely steady for k ≥ 2 and $m/n \geq 10$.

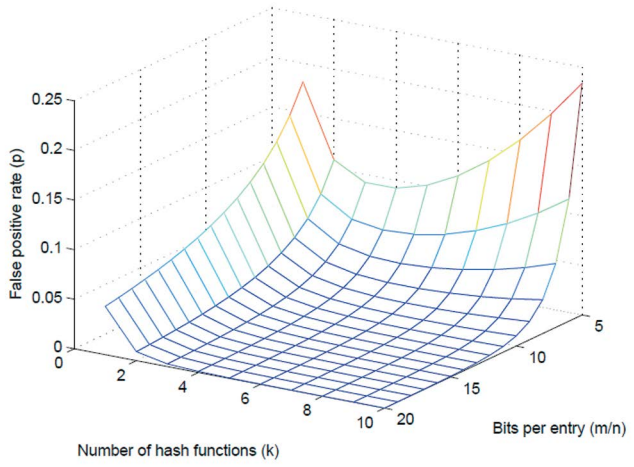

Figure 2. False positive rate (p) as a function of the number of hash function (k) and bits per entry (m = n) (close up).

4. Conclusion

In this article, we anticipated a phrase search plot dependent on Bloom filter that is significantly quicker than winning strategies as it needs just a solitary round of correspondence and Bloom filter checks. The arrangement addresses the high computational expense by reconsidering phrase search as n-gram confirmation instead of an area search or a successive chain check. In contrast to different plans, our plan respects just the subsistence of a phrase, taking out any data of its area. There is no need of utilizing consecutive check in our plans, and is parallelizable and has a practical stockpiling necessity. This methodology is likewise the first to assign effective phrase search which is autonomous without utilizing conjunctive keyword search to distinguish competitor reports. As per our examination, it curbs the expense of capacity than every single existing arrangement, where an advantaged computational expense was traded for lower stockpiling. While displaying same correspondence cost to essential existing arrangements, the proposed arrangement can likewise be adjusted to obtain greatest speed or rapid with a reasonable storage cost by the application.

5. References

1. Boneh D, Crescenzo GD, Ostrovsky R, Persiano G. Public key encryption with keyword search. Proceedings of Eurocrypt; 2004. p. 1–15.
2. Waters R, Balfanz D, Durfee G, Smetters DK. Building an encrypted and searchable audit log. Network and Distributed System Security Symposium; 2004. p. 1–10. PMid: 15057567.
3. Ding M, Gao F, Jin Z, Zhang H. An efficient public key encryption with conjunctive keyword search scheme based on pairings. IEEE International Conference on Network Infrastructure and Digital Content; 2012. p. 526–30. https://doi.org/10.1109/ICNIDC.2012.6418809
4. Kerschbaum F. Secure conjunctive keyword searches for unstructured text. International Conference on Network and System Security; 2011. p. 285–9. https://doi.org/10.1109/ICNSS.2011.6060016
5. Hu C, Liu P. Public key encryption with ranked multi keyword search. International Conference on Intelligent Networking and Collaborative Systems; 2013. p. 109–13.
6. Fu Z, Sun X, Linge N, Zhou L. Achieving effective cloud search services: Multi-keyword ranked search over encrypted cloud data supporting synonym query. IEEE Transactions on Consumer Electronics. 2014; 60(1):164–72. https://doi.org/10.1109/TCE.2014.6780939
7. Clarke CLA, Cormack GV, Tudhope EA. Relevance ranking for one to three term queries. Information Processing and Management. 2000; 36(2):291–311. https://doi.org/10.1016/S0306-4573(99)00017-5
8. Tuo H, Wenping M. An effective fuzzy keyword search scheme in cloud computing. International Conference on Intelligent Networking and Collaborative Systems; 2013. p. 786–9. https://doi.org/10.1109/INCoS.2013.150
9. Zheng M, Zhou H. An efficient attack on a fuzzy keyword search scheme over encrypted data. International Conference on High Performance Computing and Communications and Embedded and Ubiquitous Computing; 2013. p. 1647–51. https://doi.org/10.1109/HPCC.and.EUC.2013.232
10. Zittrower S, Zou CC. Encrypted phrase searching in the cloud. IEEE Global Communications Conference; 2012. p. 764–70. https://doi.org/10.1109/GLOCOM.2012.6503205
11. Tang Y, Gu D, Ding N, Lu H. Phrase search over encrypted data with symmetric encryption scheme. International Conference on Distributed Computing Systems Workshops; 2012. p. 471–80. https://doi.org/10.1109/ICDCSW.2012.89
12. Poon H, Miri A. An efficient conjunctive keyword and phrase search scheme for encrypted cloud storage systems. IEEE International Conference on Cloud Computing; 2015. p. 1–155. PMCid: PMC4688098.
13. Poon H, Miri A. A low storage phrase search scheme based on Bloom filters for encrypted cloud services. IEEE International Conference on Cyber Security and Cloud Computing; 2015. p. 1–53. PMCid: PMC4688098.
14. Rhee HS, Jeong IR, Byun JW, Lee DH. Difference set attacks on conjunctive keyword search schemes. Proceedings of the Third VLDB International Conference on Secure Data Management; 2006. p. 64–74. https://doi.org/10.1007/11844662_5
15. Cai K, Hong C, Zhang M, Feng D, Lv Z. A secure conjunctive keywords search over encrypted cloud data against inclusion-relation attack. IEEE International Conference on Cloud Computing Technology and Science; 2013. p. 339–46.