Enabling Generic, Verifiable and Secure Data Search in Cloud Services

Mr.T.A. Mohanaprakash M.E, Ph.D[1], S.M.Hajee ijas mohamed[2], B.Naresh kumar[3], M.Naresh[4]

[1]ASSOCIATE PROFESSOR, DEPT OF CSE, PANIMALAR INSTITUTE OF TECHNOLOGY, CHENNAI
[2]B.E STUDENT, DEPT OF CSE, PANIMALAR INSTITUTE OF TECHNOLGY, CHENNAI
[3]B.E STUDENT, DEPT OF CSE, PANIMALAR INSTITUTE OF TECHNOLGY, CHENNAI
[4]B.E STUDENT, DEPT OF CSE, PANIMALAR INSTITUTE OF TECHNOLGY, CHENNAI

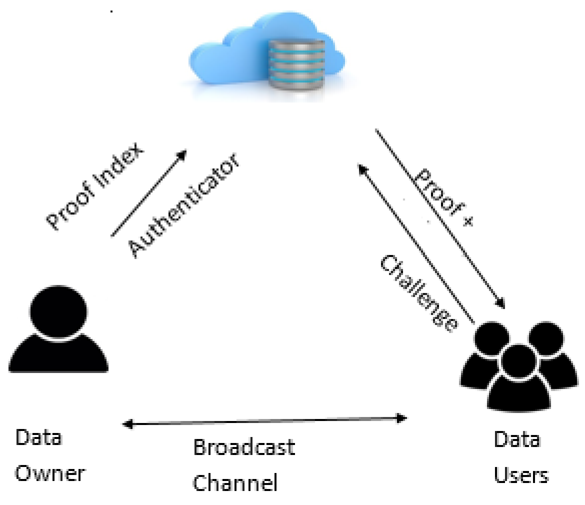
---***---

Abstract – *Cloud storage allows users to retrieve and share their data conveniently with well understood benefits, such as on demand access, reduced data maintenance cost, and service elasticity. Meanwhile, cloud storage also brings serious data privacy issues, i.e., the disclosure of private information. In order, to ensure data privacy without losing data usability, a cryptographic notion named searchable symmetric encryption (SSE), has been proposed. By using SSE, users can encrypt their data before uploading to cloud services, and cloud services can directly operate and search over encrypted data, which ensures data privacy. Searchable Symmetric encryption has been widely used in a cloud storage. It allows cloud services to directly search over the encrypted data. Most verifiable SSE scheme only enable verifiability for single user model. So, we propose a GSSE, a generic verifiable SSE framework to ensure search result integrity and freshness across multiple users GSSE provides verifiability for any SSE scheme and it supports data updates. It ensures both the freshness and integrity of search results across multiple users and data owner.*

Keywords **– Enabling generic, searchable symmetric encryption (SSE), Encryption, Cloud services**

I. INTRODUCTION

In cloud computing data were easily accessed anywhere and portable, though it is very scalable and ease of access it creates more security issues and threats. Even though it contains a encryption system privacy issues occurs. To avoid such consequences, it proposes a strong encryption techniques using generic technology to protect privacy measures as strong as possible. Cloud storage allows users to retrieve and share their data conveniently with well understood benefits, such as on demand access, reduced data maintenance cost, and service elasticity. Meanwhile, cloud storage also brings serious data privacy issues, i.e., the disclosure of private information. In order to ensure data privacy without losing data usability, a cryptographic notion named searchable symmetric encryption (SSE), has been proposed. By using SSE, users can encrypt their data before uploading to cloud services, and cloud services can directly operate and search over encrypted data, which ensures data privacy. Due to the increasing popularity of cloud computing, more and more Data owners are motivated to outsource their data to cloud servers for great convenience and reduced cost in data management. Data owners offer services to a large number of businesses and companies, they stick to high security standards to improve data security



by following a layered approach that includes data encryption, key management, strong access controls, and security intelligence. The cloud server executes the query and returns the encrypted documents with an additional proof according to the token generated by Data owners. The Data users will receive the result with the corresponding proof so they can verify the correctness and decrypt encrypted documents after the verification is correct.

II. EXISTING SYSTEM

The existing VSSE (Verifiable Searchable Symmetric Encryption) schemes exhibit very limited applicability, like only supporting static database, demanding specific SSE constructions, or only working within the single-user model. There is a setup phase that produces an encrypted index for a selected collection of documents and then phase, no additions or deletions of documents are often supported. It has some disadvantages such as

• Low search Efficiency
• The search delay of the scheme is proportional to the size of the database.
• It is not suitable for the large scale databases.
• Doesn't support verification upon file update.
• Data Integrity attacks.

III. PROPOSED SYSTEM.

Generic Search Symmetric Encryption (GSSE), which provides verifiability for any SSE schemes and further supports data updates. To generically support result verification, we first decouple the proof index in GSSE from SSE. The data owner first extracts the keywords of each document and builds a keyword index. He/she encrypts the documents as well as the keyword index. The data owner outsources the encrypted documents as well as the encrypted keyword index to the cloud. The Data user get the each result, the proof and the public verification key, they itself or others can verify the freshness, authenticity, and completeness of the search result even Without decrypting them.We also develop a time stamp chain for data freshness maintenance across multiple users.

IV. MODULE DESCRIPTION

1. Registration

It is a process of enrolling or being enrolled into the cloud. To utilize the cloud documents, every Data Owner and Data User should enroll. During this process your basic information like email, contacts etc., are collected and stored within the Cloud. The cloud id for a specific user will get automatically generated during the registration.

2. Data owner

Data Owner extracts the keywords of each document and also builds a keyword Index. It encrypts the documents and the keyword Index using a key and outsources in Cloud. Data Owner provides the Public Verification Key and Proof Index to the Data User via Cloud for document verification. It is used to only authorized person to add, modify, or delete the document(s) from the cloud..

3. Data user

Data User send a request to the cloud server. After request granted from the Cloud, the Data User receiving the Public Verification Key from the Cloud generated by Data Owner. The Data User now decrypt and download the encrypted documents, after verifying with the Public Verification Key. After receiving a verification from cloud, the data user will download the file within a particular time limit.

4. Cloud Service Provider

The Cloud Service Provider can view all the uploaded and downloaded documents in the Cloud. The CSP receives the document request from the Data User, verifies the authentication before granting permission. Then the CSP executes the query and returns the encrypted document consistent with the search token. And also returns a further proof with the document, to verify the search result.

V.EVALUATION

In order to demonstrate the feasibility of GSSE, we have implemented it by using Crypto++ 5.6.5. The prototype is written by about 2200 lines of code. We use 128-bit AES-CBC to encrypt the authenticators and sign it with RSA signature. We implement two random-oracles with HMAC-

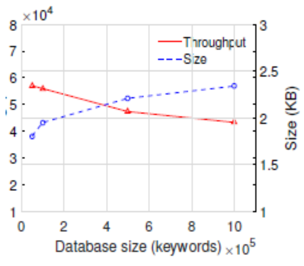


Fig. 7: *Prove* cost

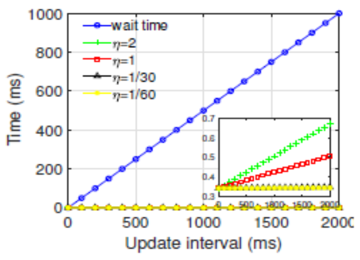


Fig. 11: *Verify* latency

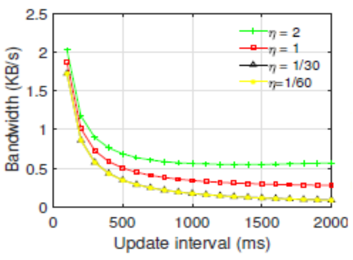


Fig. 12: Bandwidth consumption

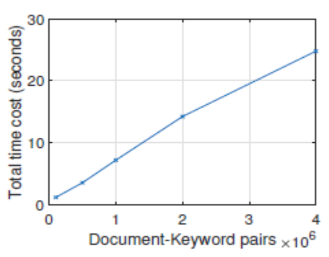


Fig. 5: *Init* delays

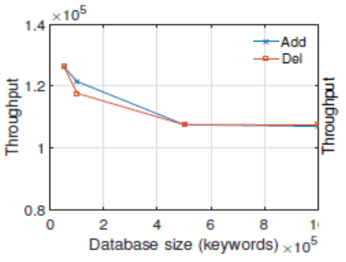


Fig. 6: *Update* throughput

SHA256 and the hash function is an implementation of SHA3-256 and the incremental hash function is MuHash. Our experiments were performed by using a machine with single thread on an Intel Core i5 2.5GHz processor with 4G RAM. We used the Enron email dataset [43] in our experiments. The used part of the dataset [43] is between "allen-p" and "kaminski-v". We extract document-keyword pairs from the dataset and construct our plaintext inverted index by using a python script. Note that the delays of extracting keywords from files are not included in our evaluation, since keyword extraction is independent with GSSE. We first measure the overheads of the algorithms proposed and then compare GSSE with a well-known SSE scheme [11] to demonstrate the small extra overhead introduced by result verification.

First, we measure the delays of the Init algorithm as shown in Fig. 5, which include the building of the proof index and the authenticator. All the delays and the subsequent measurements are the average results with ten runs of experiments. Note that the cost of building the authenticator is negligible. The delays of generating the proof index are proportional to the size of the document-keyword pairs, since GSSE performs the same number of insertions to the number of the document-keyword pairs. Overall, the initialization consumes around 25 seconds where the documents include four million keywords, which is acceptable. The update delays are decided by the size of the database that is measured by the number of keywords. Strictly speaking, the delays are directly related to the number of the layers in MPT. In order to show the relationship between the update delays and the database size, we use various numbers of keywords to measure the delays. Since the number of keywords varies from each file, we use throughput to measure the number of keyword-document pairs that can be updated per second (see Fig. 6). We observe that the throughput of adding and deletion operations are almost the same. The throughput decreases when the size of the database grows. They can support 110,000 updates per second with one million keywords database. Similarly, we observe that the bandwidth overhead incurred by update token is decided by the number of keywords contained in the file. Each update token takes about 32 bytes, which is acceptable as well. As shown in Fig. 7, the server can perform about 43,000 prove operations per second even when the size of the database is one million keywords, which indicates the server can simultaneously support 43,000 concurrent queries submitted by users. Note that this experiment only measures the cost of generating the proofs, not including the waiting time for the authenticator in the checkpoint. The communication overhead incurred by proof delivery is only a few kilobytes, which is decided by the number of layers in MPT, and gradually increases as the database grows (see Fig. 7 and Fig. 8). We measure the storage cost MPT as shown in Fig. 9. If we use a database with 1,000,000 keywords, the storage overhead is about 82MB. Compared with the size of the original dataset itself, 590 MB, the overhead is relatively small. Note that, if a data owner stores various media types of data set (e.g., images or music) with fewer keywords or attributes, the storage overhead of MPT will further reduce to be practically negligible, compared with the size of the data set itself. The performance of Generate algorithm performed by data users is presented in Fig. 10. We observe that the measured delays of generating root are all within 0.1 milliseconds and are acceptable. In Fig. 11, we evaluate the verification delays in data users. Note that an entire verification delay includes the delay of waiting for a checkpoint and the delay of executing the Check and the Generate algorithms. Since the execution delay of the Generate algorithm is relatively stable, around 0.1 milliseconds, we do not plot it in Fig. 11. Here, _ is the update frequency of the data owner. We assume that the time that a user initiates a query is uniformly distributed during an update interval, and then the user's waiting delays are also uniformly distributed. Therefore, the expected delay is half of the update interval and the verification delays are dominated by the waiting delays. The execution delay of the Check algorithm is negligible and is proportional to the update interval, which is mainly incurred by verifying the signature

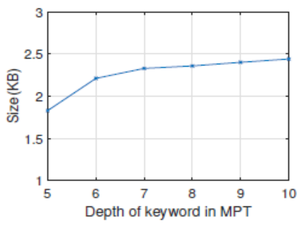


Fig. 8: Proof cost of MPT

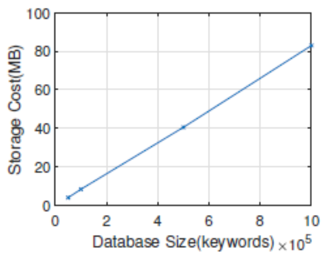


Fig. 9: Storage cost of MPT

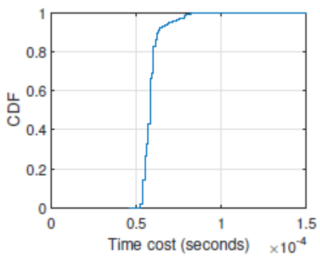


Fig. 10: *Generate* performance

and decrypting authenticators. Kindly note that in above measurement, we do not take into account the network transmission and propagation delays, as they vary in different specific network contexts and do not reflect the essential extra cost directly introduced by our verification design. We do, however, report the communication overhead in terms of the message size, as shown in Fig. 12. In a later experiment, we will also show that we can set an update interval so as to make a trade-off between verification delays and communication overhead. Fig. 12 shows the bandwidth costs for authenticator update. Here, the size of the first authenticator in each update interval is around 112 bytes, which includes 32 bytes of the root of MPT, 8 bytes of the timestamp, an 8 bytes AES-CBC extension and a 128 bytes RSA signature. Overall, the bandwidth of the authenticator includes two part: the overhead introduced by the fixed update time point and the overhead introduced by data update. We can observe that the bandwidth cost increases to about 2KB per second when the update interval decrease to zero, this is introduced by the fixed update time point which is inversely proportional to the bandwidth overhead. Moreover, the bandwidth gradually increases when the update interval becomes too long. This overhead is introduced by the length of the authenticator, because as the update interval grows, the length of the authenticator becomes larger. Overall, the cost should be acceptable to achieve GSSE. According to the results, in order to make a decent tradeoff between verification delays and bandwidth costs, we suggest choosing an update interval between 500 milliseconds and 1,500 milliseconds.

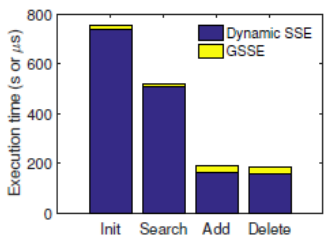


Fig. 13: Comparison with SSE [11]

VI. CONCLUSION

A dynamically verifiable SSE scheme, which can be applied to any SSE schemes with a three-party model and does not require modifications on them. By building authenticators and a proof index, GSSE provides efficient search result verification, while preventing data freshness attacks and data integrity attacks in SSE

REFERENCES

- S. Jarecki, C. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Outsourced symmetric private information retrieval," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013, pp. 875–888.
- S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable Symmetric encryption," in Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012, pp. 965–976.
- D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for Boolean queries," in Advances in Cryptology-CRYPTO 2013. Springer, 2013, pp. 353–373.
- M. Naveed, M. Prabhakaran, and C. A. Gunter, "Dynamic searchable encryption via blind storage." in Proceedings of the IEEE Symposium on Security and Privacy, San Jose, CA, May, 2014.
- C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure rankedkeyword search over encrypted cloud data," in Proc. of ICDCS'10,2010.
- P. Mell and T. Grance, "Draft nist working definition of cloudcomputing," Referenced on Jan. 23rd, 2010 Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2010.
- [M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. UCBEECS-2009-28, Feb 2009.
- Cloud Security Alliance, "Security guidance for critical areas of focus in cloud computing," 2009, http://www.cloudsecurityalliance.org.