International Journal of Cloud Computing and Database Management

E-ISSN: 2707-5915
P-ISSN: 2707-5907
IJCCDM 2023; 4(1): 09-11
Received: 08-11-2022
Accepted: 14-12-2022

Dr. Nesar Ahmad
HOD and Professor, Department of Statistics and Computer Science, Tilka Manjhi Bhagalpur University, Bhagalpur, Bihar, India

Supriya Raj
Research Scholar, Department of Statistics and Computer Science, Tilka Manjhi Bhagalpur University, Bhagalpur, Bihar, India

Corresponding Author:
Dr. Nesar Ahmad
HOD and Professor, Department of Statistics and Computer Science, Tilka Manjhi Bhagalpur University, Bhagalpur, Bihar, India

Modern symmetric searchable encryption technique for string classification in cloud technology

Dr. Nesar Ahmad and Supriya Raj

DOI: https://doi.org/10.33545/27075907.2023.v4.i1a.44

Abstract
In this paper, we give an effective and easy-to-use symmetric Searchable symmetric encryption (SSE) for string search. Unlike previous plans, we use hash affixing instead of a chain of encryption activities for record age, which makes it suitable for lightweight applications. Here is the first time that probabilistic trapdoors for string search in SSE are suggested. We give solid proof that our plan isn't secure against a real but curious server, depending on the definitions. Here, we also show another idea for search design protection, which protects some of the spillage from the trapdoor. I've shown that our plan is safe under the inquiry design in the definition of noticeability. I explain why SSE's scheme for string search can't meet the flexibility criteria mentioned, and I also suggest changes to our plan so that it can be used against moving enemies at the cost of more changes to communications and more memory space.

Keywords: Cloud storage, symmetric key, searchable encryption, hash-chain, and lightweight cryptography

1. Introduction
The cloud is meant to store a large number of encrypted reports. With the rise of distributed computing, more and more customers and driving organisations are getting used to the idea of private storage being re-appropriated. This makes it easy for customers with a lot of encrypted information to store it in the cloud in secret. No matter what, this makes it hard to look. This gives a boost to a new field of study called "Searchable Encryption" (SE). SE can be divided into symmetric accessible encryptions (SSE) and Asymmetric Searchable Encryption (ASE). We think about the SSE for string look in this paper. In the SSE, the customer encrypts the information and stores it in the cloud. It can be seen that the customer can organise the information on their own and keep up with more information structures to get to the information they need quickly. In this method, the calculations on the customer side are as big as the information, but the calculations to get to the information are smaller for both the customer and the cloud server.

Since a cloud server stores a lot of archives, searching for a catchphrase may bring up a large number of documents, most of which were not expected. This causes unnecessary traffic on the system. This makes people look against a string, which makes the question more clear. Looking for string is a multi-catchphrase seek that makes it safe to ask for watchwords. So, even if all of these keywords are in a report, their order and proximity to each other should be taken into account when searching. The record table should be set up in a way that keeps the words' location information safe.

Even though there are a few works in writing that include string look (like [1, 8, 2, 3], and [5]), most of them need formal security evidence against the changed meanings of [1] and also send a lot of information to the server after the search. In the SSE plot, the server has to know nothing about how the questions and information are asked and collected. SSE does this by using symmetric cryptographic natives instead of open key encryption, which requires a lot of calculations but leaks little data [1]. Here, we use a model that will be used throughout the paper to explain how our calculations and data structures work.

With more and more documents being stored in the cloud, it can be hard and take a lot of time to find the one you want. One solution could be to use symmetric searchable encryption (SSE), which lets one party send its data to another party (like a cloud) for private storage and selective searching.

In this paper, we looked at the security definitions of [11] again and came up with a new lightweight SSE scheme called s;s for searching strings.

2. Review of the Literature

Mihir Bellare, Alexandra Boldyreva, and Adam ONeill. Encryption that is both sure and easy to look up. In the pages 535-552 of the Annual International Cryptology Conference. Springer.

When the encryption calculation is random, the traditional ideas of security for open-key encryption plans must be met. These are mainly the lack of definition or semantic security under a picked plaintext or picked ciphertext attack [4, 3, 8, 10]. This paper looks at the case where the calculation for encryption is deterministic. First, we talk about the application of the spurring. Quick pursuit. More and more people are interested in remote information storage as redistributed databases [2]. The information will be stored in a jumbled way. (The company that specialises in databases isn't believed.) We like the idea of an open key setting, where anyone can add encoded information to the database that a known "beneficiary" can find and decode. The encryption scheme must make it possible for the recipient to look for information. Public key encryption with keyword search (PEKS) [11, 1, 8] is a solution that gives strong protection and can be shown to do so. However, search requires a lot of direct investment in the size of the database. Since databases can be as big as terabytes, this isn't enough space. The commonsense network shows that they need a record with a certain field value that can be found in a time that is proportional to the size of the database. (For example, by using tree-based information structures correctly.) Only this is possible with deterministic encryption. The encoded fields can be stored in the data structure, and an objective ciphertext can be found in a time that is proportional to the size of the database. The question is what kind of safety one can expect. To answer this, we need to know what security means in the context of deterministic encryption. A definition. One option is to just ask for one-wayness, but we might want to make sure that as much information about the plaintext is shared as possible. To figure out how big this could be, we look at two rules that are built into deterministic encryption. First of all, there is no way to protect anything if the plaintext is known to come from a small space. In reality, if the enemy knows that c is the encryption under open key pk of a plaintext x from a set X, he or she can record the encryption cx of x under pk for all x X and return as the decryption of c the x that makes cx = c. We deal with this by possibly requiring protection when the plaintext comes from a space with a high min-entropy. Second, and this is getting less impressive, the ciphertext is half information about the plaintext. We deal with this by possibly requiring that halfway data not be spilled when the plaintext and incomplete data don't depend on the open key. This makes sense because, after all, open keys are hidden in our product and information doesn't depend on them. Even though our new idea of security for deterministic encryption, which we call PRIV, is more likely to be broken than the old ideas that used randomised plans, it is still very strong. This new idea can be reached through the next question. Developments. Our first step is normal and standard: Deterministically encrypt plaintext x by using the encryption calculation of a random plan but using a hash of (the open key and) x as coins. We show that this "Encode with-Hash" deterministic encryption scheme is PRIV secure in the Random Oracle (RO) model of [11], assuming that the beginning randomised plan is IND-CPA secure. The next thing we're making is an addition to RSA-OAEP [10, 1]. The change to the cushioning is fixed, but it uses three Feistel adjustments instead of the two that OAEP uses. In the RO model, where RSA is accepted, it has been shown that RSA-DOAEP is PRIV secure. The length-safety feature of this development is an appealing one. (The length of the ciphertext grows until it matches the length of the plaintext.) This is important when transfer speed is expensive—senders in a database setting could be power-dependent devices—and when you want to make sure that code is inherited.

3. Our Scheme

In this section we present our SSE scheme ss for string search, which is composed of four algorithms KeyGen, Build Index, Trapdoor and Search. First, we formally define the scheme. In the subsequent subsections, we will discuss these algorithms in detail with illustrations.

Scheme 1 (ss). The scheme ss is a collection of four polynomial-time algorithms (KeyGen, BuildIndex, Trapdoor, Search) such that:

- **Key Gen (1):** Key Gen is a probabilistic key generation algorithm that is run by the client to setup the scheme (see Algorithm 1). It takes a security parameter, and returns a secret master key km and a mask-key k0 which are to be kept privately at client’s end and a session key ks which is to be shared between client and the server. Client also shares a -bit prime p with the server. The length of km, k0 and ks are polynomial bounded
- **Build Index (km; k0; ks; p):** Build Index is a probabilistic algorithm run by the client to generate SI = (I; Ir; Ic). It takes km, k0, ks, p and returns SI. Since Build Index is randomized, we write this as SI Build Index (km; k0; ks;p)(s).
- **Trapdoor (km; ks; p; s):** Trapdoor is a probabilistic algorithm run by the client to generate a trapdoor for a given string of words s = (w1; w2; : : : ; wl). It takes km, k0, ks, p and s as input and outputs t = (t1; t2;: : : ; tl), where ti is the trapdoor corresponding to the word wi. Since trapdoor is randomized, we write this as t T rapdoor (km;k0;ks;p)(s)
- **Search (SI; t):** Search is run by the server in order to search for the documents in D that contain the string s. It takes ks, SI and trapdoor t of the string s as inputs, and returns D(s), the set of identifiers of documents containing the string s. Since this algo-rithm is deterministic, we write it as D(s) = Searchks (SI; t)

4. Conclusion

We've shown that our scheme is safe based on the definition of "non-adaptive in distinguishability" [11]. For an active adversary, we suggest that the scheme s;s be changed at the cost of more memory on the client's side and two rounds of communication for each change to the document collection. In this direction, more research can be done to come up with an effective SSE scheme that only needs one round of communication, if possible. With our plan, the server doesn't know anything about how often words are used and where they are placed other than what it can learn from the history.

We bring a new security idea to SSE for the first time. It's called "search pattern in distinguishability." It can be seen that with non-adaptive distinguishability security, the keywords are guaranteed to be safe from possible leakage from the index, but not from possible leakage from the trapdoor. In order to do this, we use a probabilistic trapdoor for the first time and show that our scheme is safe by this measure. We've used the TIMIT dataset to test our scheme and put it to use for the first time to search over phone symbols. We've also used our scheme on DNA data from [1] and were able to successfully do pattern matching over an encrypted domain.

5. Abbreviation

Searchable symmetric encryption (SSE)
Searchable encryption (SE)
Asymmetric Searchable Encryption (ASE)
Impulse under samecloseness-plan chose plaintext attacks (IND-CLS-CPA)
Iindistinguishability under selective chosen plaintext attacks (IND-SCPA)
Public key encryption with keyword search (PEKS)
Mandatory Access Control (MAC),
Role Based Access Control (RBAC)
Optimal Asymmetric Encryption Padding (OAEP)
RSA (Rivest–Shamir–Adleman)
Random Oracle (RO)
Searchable Public-Key Ciphertexts with Hidden Structures (SPCHS)
Secure multi-party computation (SMC)

6. References

1. https://github.com/iskana/pbwt sec/tree/master/sample dat.
2. http://www.fon.hum.uva.nl/david/massp/2007/timit/train/dr5/fsdc0/.
3. Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, *et al*. Searchable Encryption Revisited: Consistency Proper-ties, Relation to Anonymous IBE, and Extensions. 2008;21:350-391.
4. Mihir Bellare, Alexandra Boldyreva, Adam ONeill. Deterministic and Efficiently Searchable Encryption. In Annual International Cryptology Conference; c2007. p. 535-552.
5. Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano. Public Key Encryption with Keyword Search. In International Conference on the Theory and Applications of Cryptographic Techniques; c2004. p. 506-522.
6. Dan Boneh, Eyal Kushilevitz, Rafail Ostrovsky, William E Skeith III. Public Key Encryption That Allows PIR Queries. In Annual Interna-tional Cryptology Conference; c2007. p. 50-67.
7. Ning Cao, Cong Wang, Ming Li, Kui Ren, Wenjing Lou. Privacy-Preserving Multi-Keyword Ranked Search Over Encrypted Cloud Data. 2014;25:222-233. IEEE.
8. David Cash, Paul Grubbs, Jason Perry, Thomas Ristenpart. Leakage-Abuse Attacks Against Searchable Encryption. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM; c2015. p. 668-679.
9. David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit S Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, *et al*. Dynamic Search-able Encryption in Very-Large Databases: Data Structures and Imple-mentation; c2014, 853.
10. David Cash, Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel-Cat˘alin˘ Ros¸u, Michael Steiner. Highly-Scalable Searchable Sym-metric Encryption with Support for Boolean Queries. In Advances in Cryptology–CRYPTO; c2013. p. 353-373.
11. David Cash, Stefano Tessaro. The Locality of Searchable Symmetric Encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques; c2014. p. 351-368.
12. Kuyoro SO, Ibikunle F, Awodele O. Cloud computing security issues and challenges; c2011.
13. Dan Jerker B Svantesson. Data protection in cloud computing The Swedish perspective, computer law & security review. 2012;28:476-480.
14. Kavitha K. Assistant Professor, Department of MCA, Adhiparasakthi engineering college, Melmaruvathur, Tamilnadu, India, Study on cloud computing model and its benefits, challenges; c2014.
15. Mell Peter, Grance Timoth. The NIST definition of cloud computing draft; c2011.
16. Garga S, Versteeg S, Buyya R. A framework for ranking of cloud computing services. Journal of future generation computer systems. 2013;22:102-122.
17. Sadr Alsadati. Sayed Mohsen, Security challenges in cloud computing in order to improve security in the development of e-government services, The 8th symposium on advances in science and technology (8th SAS Tech), Mashhad, Iran, 8th SASTech.khi.ac.ir; c2013.
18. Soon-Keow Chong, Jemal Abawajy, Masitah Ahmad, Isredza Rahmi A. Hamid, Enhancing Trust Management in cloud environment, international conference on innovation, management and technology research, Malaysia; c2013. p. 22-23.
19. Masdari M, ValiKardan S, Shahi Z, Azar SI. Towards workflow scheduling in cloud computing: a comprehensive analysis, Journal of network and computer applications. 2016;31:66:64-82.
20. Masdari M, Nabavi SS, Ahmadi V. An overview of virtual machine placement schemes in cloud computing, Journal of network and computer applications. 2016;31(66):106-27.
21. Amol C. Adamuthe cloud computing – A market perspective and research directions, I.J Information technology and computer science, (http://www.mecs-press.org/); c2015. p. 42-53.
22. Arjunan G. An survey on cloud computing process and its applications, international journal of research in computer applications and robotics. www.ijrcar.com. 2015;3:10.
23. European Academic Research. ISSN: 2286-4822, www.euacademic.org, 2016;3:11.
24. Hossein Mohammadi. Considerations on models, algorithms and security challenges in cloud computing, IJISET - International journal of innovative science, engineering & technology. 2015;3(6):2348-7968.
25. Chandni Jain M. Cloud Computing: Network/security threats and counter measures, International journal of advanced research in computer and communication engineering. 2015;4:8.