International Journal of Cloud Computing and Database Management

E-ISSN: 2707-5915
P-ISSN: 2707-5907
IJCCDM 2020; 1(1): 43-45
Received: 22-11-2019
Accepted: 28-12-2019

Penumalli Divyalekhya
Department of Computer Science, Sri Venkateswara University, Tirupati, Andhra Pradesh, India

Corresponding Author:
Penumalli.Divyalekhya
Department of Computer Science, Sri Venkateswara University, Tirupati, Andhra Pradesh, India

A novel symmetric searchable encryption scheme for string identification in cloud computing

Penumalli Divyalekhya

DOI: https://doi.org/10.33545/27075907.2020.v1.i1a.8

Abstract
In this paper, we initially give a productive and simple to-actualize symmetric accessible encryption conspire (SSE) for string search, unlike past plans, we use hash affixing rather than chain of encryption activities for record age, which makes it reasonable for lightweight applications. Here it's first to propose probabilistic trapdoors in SSE for string search. We give solid evidence of non-versatile security of our plan against genuine yet inquisitive server dependent on the definitions. Here additionally present another idea of search design protection, which gives a proportion of protection from the spillage from trapdoor. I have demonstrated that our plan is secure under inquiry design in noticeability definition. I show why SSE conspire for string search can't achieve versatile in noticeability criteria as referenced and furthermore propose adjustments of our plan with the goal that the plan can be utilized against dynamic foes at the expense of more adjusts of correspondences and memory space.

Keywords: Cloud Storage, Symmetric Key, Searchable Encryption, Hash-Chain, Lightweight Cryptography

1. Introduction
The cloud is intended to hold a substantial number of encoded reports. With the approach of distributed computing, developing number of customers and driving associations have begun adjusting to the private stockpiling re-appropriating. This permits asset obliged customers to secretly store a lot of scrambled information in cloud effortlessly. Be that as it may, this keeps one from seeking. This offers ascend to a recently rising field of research, called accessible encryption (SE). SE can be ordered into symmetric accessible encryptions (SSE) and lopsided accessible encryptions (ASE). In this paper, we think about the SSE for string look. In the SSE, the customer encodes the information and stores it on the cloud. It might be noticed that customer can arrange the information in a self-assertive way and can keep up extra information structures to accomplish wanted information proficiently. In this procedure, the underlying customer side calculation is in this way as expansive as the information, however resulting calculations to get to information is less for both customer and the cloud server.

Since enormous volumes of archives are put away in a cloud server, seeking against a catchphrase may result into extensive number of documents, the vast majority of which are not expected, causing superfluous system traffic. This rouse looking against a string, which enables the inquiry to be progressively explicit. Looking for string is a multi-catchphrase seek where the requesting of watchwords is protected. So, notwithstanding the nearness of every one of these catchphrases in a report, their requesting and contiguousness are to be taken consideration off while searching the record table should be set up so that the nearness data of the words can be protected.

Albeit few works are accessible in the writing including string look (for example [1], [8], [2], [3], [5]), however the vast majority of them need formal security evidence against the overhauled meanings of [1] and furthermore uncover heaps of information's to the server following the pursuit In the SSE conspire, the server is required to get the hang of nothing about the inquiry inquiries and information accumulations. SSE accomplishes this by utilizing symmetric cryptographic natives rather than overwhelming calculations of open key encryption at the expense of little spillage of data [1]. Here we take a model which will be reached out all through the paper to outline our calculations and information structures

2. Literature Survey

Mihir Bellare, Alexandra Boldyreva, and Adam ONeill. Deterministic and Efficiently Searchable Encryption. In Annual International Cryptology Conference, pages 535–552. Springer

The traditional ideas of protection for open key encryption plans, principally lack of definition or semantic security under picked plaintext or picked ciphertext assault [4, 3, 8, 10], must be met when the encryption calculation is randomized. This paper treats the situation where the encryption calculation is deterministic. We start by talking about the spurring application. Quick pursuit. Remote information stockpiling as redistributed databases is of expanding interest [2]. Information will be put away in scrambled structure. (The database specialist organization isn't believed.) We are keen on an open key setting, where anybody can add to the database encoded information which a recognized "beneficiary" can recover and unscramble. The encryption conspire must allow search (by the recipient) for information recovery. Open key encryption with watchword search (PEKS) [11, 1, 8] is an answer that provably gives solid protection yet search requires significant investment direct in the size of the database. Given that databases can be terabytes in size, this is restrictive. The commonsense network demonstrates that they need record containing a given field worth can be a recovered in time logarithmic in the size of the database. (For instance, by means of proper tree-based information structures.) Deterministic encryption permits only this. The encoded fields can be put away in the information structure, and one can discover an objective ciphertext in time logarithmic in the size of the database. The inquiry is the thing that security one can anticipate. To answer this, we need a meaning of security for deterministic encryption. A definition. One chance is to simply request one-wayness, yet we might want to ensure halfway data about the plaintext to the most extreme degree conceivable. To measure what this could be, we note two intrinsic constraints of deterministic encryption. To begin with, no protection is conceivable if the plaintext is known to originate from a little space. In reality, realizing that c is the encryption under open key pk of a plaintext x from a set X, the enemy can register the encryption cx of x under pk for all x ∈ X, and return as the unscrambling of c the x fulfilling cx = c. We address this by possibly requiring protection when the plaintext is drawn from a space of huge min-entropy. Second, and progressively unpretentious, is that the ciphertext itself is halfway data about the plaintext. We address this by possibly requiring non-spillage of halfway data when the plaintext and incomplete data don't rely upon the open key. This is sensible on the grounds that, all things considered, open keys are covered up in our product and information doesn't rely upon them. While surely more vulnerable than the old-style ideas met by randomized plans, our subsequent thought of security for deterministic encryption, which we call PRIV, is still very solid. The following inquiry is the way to accomplish this new thought. Developments. Our first development is conventional and normal: Deterministically encode plaintext x by applying the encryption calculation of a randomized plan yet utilizing as coins a hash of (the open key and) x. We show this "Encode with-Hash" deterministic encryption conspire is PRIV secure in the irregular prophet (RO) model of [11] expecting the beginning randomized plan is IND-CPA secure. Our subsequent development is an augmentation of RSA-OAEP [10, 1]. The cushioning change is deterministic yet utilizes three Feistel adjusts instead of the two of OAEP. RSA-DOAEP is demonstrated PRIV secure in the RO model accepting RSA is single direction. This development has the appealing component of being length-safeguarding. (The length of the ciphertext rises to the length of the plaintext.) This is significant when transfer speed is costly — senders in the database setting could be power-obliged gadgets—and for making sure about inheritance code.

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):** KeyGen 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 k_m and a mask-key k^0 which are to be kept privately at client's end and a session key k_s which is to be shared between client and the server. Client also shares a -bit prime p with the server. The length of k_m, k^0 and k_s are polynomial bounded
- **Build Index (k_m; k^0; k_s; p):** Build Index is a probabilistic algorithm run by the client to generate SI = (I; I_r; I_c). It takes k_m, k^0, k_s, p and returns SI. Since Build Index is randomized, we write this as SI Build Index $_{(km;\ k}0_{;\ ks;p)}$(s) .
- **Trapdoor (k_m; k_s; p; s):** Trapdoor is a probabilistic algorithm run by the client to generate a trapdoor for a given string of words s = (w_1; w_2; : : : ; w_l). It takes k_m, k^0, k_s, p and s as input and outputs t = (t_1; t_2;: : : ; t_l), where t_i is the trapdoor corresponding to the word w_i. Since trapdoor is randomized, we write this as t T rapdoor$_{(km;k}$0$_{;ks;p)}$(s)
- **Search (SI; t):** Search is run by the server in order to search for the documents in D that contain the string s. It takes k_s, SI and trapdoor t of the string s as inputs, and returns D(s), the set of identifiers of documents containing the string s. Since this algo-rithm is deterministic, we write it as D(s) = Search$_{ks}$ (SI; t)

4. Conclusion

With the increasing number of documents stored in cloud, search-ing for the desired document can be a difficult and resource intensive task. One solution may be to use symmetric searchable encryption (SSE) which allows one party to outsource the storage of its data to another party (a cloud) privately while enabling to search selectively over it. In this paper we revisited the security definitions of [11] and proposed a new lightweight SSE scheme $_{s;s}$ for string search. We have shown that our scheme is secure under the non-adaptive in distinguishability definition [11]. For active adversary, we propose modification of the scheme $_{s;s}$ at the additional cost of memory at client's end and two rounds of communications for one modification of document collection. Towards this direction, future research can be performed to design efficient SSE scheme ideally with one round of communication. With our scheme, server does not

learn the information related to word frequency and word positions except what it can learn from the history.

We, for the first time, introduce new security notion in SSE, named, search pattern in distinguishability. It may be observed that with non-adaptive in distinguishability security, although the keywords are guar-anteed to be secure from the possible leakage from index, however it does not guarantee the security from the possible leakage from trapdoor. Towards this, we for the first time introduce probabilistic trapdoor and prove that our scheme is secure under such criterion. We have implemented our scheme for the first time to search over phone symbols and validated it using the TIMIT dataset. We have also implemented our scheme over DNA data of [1] and successfully achieve pattern matching functionality over encrypted domain

5. 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, 2007, 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. 2004, 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, 2007, 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, 2015, 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, 2014, 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, 2013, 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, 2014, 351-368.