ID-based deletion searchable encryption scheme

ID-based deletion searchable encryption scheme

Xiaoyan Deng [1], Hao Cheng [2], Bo Sun [3,*], Lulin Ni [4] and Qingbing Ji [4]

[1]Chengdu University of Information Technology, China
[2]Beijing Municipal Public Security Bureau, China
[3]National computer network emergency response technical team/coordination center of China, China
[4]China Electronics Technology Group Corporation, 30 Group, China
*jqbdxy@163.com

Abstract. Searchable encryption technology frees up user's local storage resources and also facilitates user's search on files in the server. At present, general searchable encryption schemes are merely static searches and cannot dynamically manipulate data already stored in server. In order to solve this problem, this paper proposes a dynamic searchable encryption scheme based on bilinear pairing, which can delete the specified identity file. When delete operation is performed, rights authentication is performed on user who issues delete request, and malicious deletion of illegal users is effectively prevented. At the same time, the security of the scheme is proved. Based on the implementation of delete function, the scheme achieves security against the choose plaintext attack.

1 Research status

With the development of cloud technology, the search function[1,2] of searchable encryption search function can no longer meet the needs of users. More and more users want to be able to remotely manipulate data stored on server. Literature [3] uses the ID of files and keyword [4] to generate searchable cipher-text. When user wants to delete a file, he only needs to send corresponding ID to server, and the server deletes the file after match is successful. Literature [5] proposed an asymmetric searchable encryption scheme. The scheme aims to improve the following loopholes: anyone can generate trapdoors, cipher-text can be tampered casually, key pairs are generated severally by users, encryption of identity, S is useless. Literatures [6-9] implement the dynamic operation of cipher-text on server in symmetric searchable encryption scheme. Among them，literature [6] adds a delete group to the server to implement the delete function of cipher-text. On the other hand, because data stored in server is out of the physical control of user, delete operation is performed by server. At this time, if an illegal user can impersonate legitimate user to “cheat” the trust of server, then illegal users can delete data from the server. This may result in the loss of data and affect user's use of data. In this paper, before user performs remote operation on data on server, server authenticates the user who issued operation request. Only user with the operation authority can operate on the data, otherwise it does not change any data on the server. In current research on authentication technology, literature [10] proposes a centralized (t, m, n)-AS group authentication scheme. The program can authenticate whether there are illegal participants in all m participants within $O(1)$ time; If there are illegal participants, the scheme can identify all illegal participants within $O(m)$ time without additional communication. Literature [11] combines the basic ideas of public key encryption and SSL protocol, and designs a two-way authentication one-time password authentication scheme. The paper gives a detailed description of specific implementation of new scheme, and adopts B/S model to implement the scheme.

2 Our Contribution

We make some contributions in the paper as follows:
-We introduced an ID-based deletion searchable encryption scheme (ID-DSE). Upon deletion, we first authenticated whether the user who issued the delete request has delete permission. A deletion request from user who does not have the deletion permission cannot delete the corresponding cipher-text.
-We give security proof of the scheme. It is proved that the scheme is against choose plaintext attack.
-Finally, comparative analysis between our scheme and other schemes is given. Draw conclusion: Under the premise of implementing delete function, the scheme does not affect the user's search operation, and does not occupy a large amount of server storage space.

3 Basic Knowledge

3.1 Bilinear Pairing

Let e: $G_1 \times G_1 \rightarrow G_T$ be a bilinear pairing, mapping from groups G_1 and G_1 to G_T, where G_1 and G_T are cyclic groups of the same prime order p. It has some properties as follows:
-Bilinearity. For any $g \in G_1$ and $a, b \in Z_p$, $e(g^a, g^b) = e(g, g)^{ab}$.
-Non-degeneracy. For any generator $g \in G_1$, $e(g, g) \in G_T$ is a generator of G_T.
-Computability. For any $g \in G_1$, there is an efficient algorithm to compute $e(g, g)$.

3.2 DBDH Assumption

Let e: $G_1 \times G_1 \rightarrow G_T$ is a bilinear map. We define the advantage function

$$Adv_{G_1,A}^{DBDH}(\lambda)$$

of an adversary A as

$$|P_r[A(g, g^a, g^b, g^c, e(g,g)^{abc})=1]-P_r[A(g, g^a, g^b, g^c, g^*)=1]|$$

Where $a, b, c \in Z_p$ are randomly chose. We say that the decisional bilinear Diffie Hellman assumption relative to generator G_1 holds if $Adv_{G_1,A}^{DBDH}(\lambda)$ is negligible for all PPT A.

4 ID-DSE

Setup(λ):the algorithm initializes the global system parameter (p, g, G_1, G_T, e), where G_1, G_T are cyclic groups of prime order p, g is the generator of G_1, and e: $G_1 \times G_1 \rightarrow G_T$ is a bilinear pairing, H_1:$\{0,1\}^* \rightarrow G_1^*$, H_2:$\{0,1\}^* \rightarrow G_1^*$, H_3:$G_2 \rightarrow \{0,1\}^{logp}$, H_4:$G_2 \rightarrow \{0,1\}^n$ are cryptographic hash functions.
KeyGen$_U$(*parma*) : Randomly select $x \leftarrow Z_p$ and set $Pk_U = g^x$ and $Sk_U = x$. Return (Pk_U, Sk_U).
KeyGen$_S$(*parma*) : Randomly select $y \leftarrow Z_p$ and set $Pk_S = g^y$ and $Sk_S = y$. Return (Pk_S, Sk_S).
PEKS(w,Sk_U, Pk_S) : Randomly select a $r \leftarrow Z_p$, and compute

$$C_1=H_3(e(g^x,H_1(w))^r),\ C_2=H_4(e(g^x, g^y)^r) \oplus H_4(e(g^x, H_2(ID))^r),\ C_3=g^r .$$

Trapdoor(w, Sk_U, ID) : Output the search trapdoor $T_w = H_1(w)^x$, and deletion trapdoor $T_d = H_2(ID)^x$.
TEST(T_w,C_1,T_d,C_2,C_3) : Output 1 if $H_3(e(g^r, T_w))=C_1$, and 0 otherwise. Output 1 if $H_4(e(g^r, T_d))=C_2 \oplus H_4(e(g^x, g^r)^y)$, and 0 otherwise.
Correctness. Let the user's key pair be (Pk_U, Sk_U) = (g^x, x) and server's key pair be (Pk_s, Sk_s)=(g^y, y). Let w, ID be the keyword contained in C and w' be that in T_w, ID' be that in T_d. Then we have the followings.
Search phase:

$$C_1=H_3(e(g^x, H_1(w))^r)$$

$$T_w = H_1(w')^x,\ C_3=g^r$$

$$H_3(e(C_3, T_w))= H_3(e(C_3, H_1(w')^x))= H_3(e(g^x, H_1(w'))^r)$$

If $w=w'$, then $H_3(e(g^x, H_1(w'))^r)= H_3(e(g^x, H_1(w))^r)$. And if $w \neq w'$, then

$$H_3(e(g^x, H_1(w'))^r) \neq H_3(e(g^x, H_1(w))^r).$$

Deletion phase:

$$C_2=H_4(e(g^x, g^y)^r)\oplus H_4(e(g^x, H_2(ID))^r)$$

$$T_d = H_2(ID')^x,\ C_3=g^r$$

$$H_4(e(g^x, C_3)^y)\oplus H_4(e(C_3\ T_d))= H_4(e(g^x, C_3)^y)\oplus H_4(e(C_3, H_2(ID')^x))$$

$$= H_4(e(g^x, g^r)^y)\oplus H_4(e(g^r, H_2(ID'))^x)$$

$$= H_4(e(g^x, g^y)^r)\oplus H_4(e(g^x, H_2(ID'))^r)$$

If $ID=ID'$, then $H_4(e(g^x, H_2(ID))^r)= H_4(e(g^x, H_2(ID'))^r)$. And if $ID\neq ID'$, then

$$H_4(e(g^x, H_2(ID))^r) \neq H_4(e(g^x, H_2(ID'))^r).$$

Security Proof

Themorem1. Our scheme is against choose plaintext attack assuming the DBDH problem is hard.

Proof. Given two random oracles $H_1:\{0,1\}^*\rightarrow G_1^*$ and $H_3:G_T\rightarrow\{0,1\}^{logp}$. Suppose A be an IND-CPA adversary that has the advantage ϵ_1 in breaking our scheme, and suppose makes at most $q_{H_3}>0$ hash queries to H_3 and at most $q_T>0$ trapdoor queries. Then, there is an algorithm B that solves the DBDH problem with the advantage at least $\varepsilon_1^* = 2\varepsilon_1/\{e\cdot q_{H_3}\cdot(1+q_T)\}$. Now we show that ε_1^* is negligible.

Appose (p, g, G_1, G_T, e) be the DBDH parameters, where p is the prime order of G_1 and G_T. Randomly select an element g from G_1. B is given $v_0=g$, $v_1=g^{\alpha}$, $v_2=g^{\beta}$, $v_3=g^{\gamma}\in G_1$ where α, β, γ are randomly select from Z_p^*. Its goal is to output $D=e(g, g)^{\alpha\beta\gamma}\in G_T$. Let D is the solution to the DBDH proplem. B finds D by interacting with A as follows:

Keygen: B send (v_0, v_1) as public key to A.

H1-Queries: B maintain a list of tuples called H_1-*List*, in which each entry is a tuple of the form $<w, h, a, c>$. The list is initially empty. When A queries the random oracle H_1 at a point $w_i\in\{0,1\}^*$, B responds as follows:

Step 1: If w_i already appears on H_1-*List* in a tuple $<w_i, h_i, a_i, c_i>$, B responds with H1.

Step 2. Otherwise, B generates a random $coin\in\{0, 1\}$, so that $Pr[coin=0] = \delta$ for some δ that will be determined later.

Step3: B select randomly an element $a\in Z_p^*$. If $coin=0$, B computes $h_i=v_2\cdot g^a=g^{\beta_1}\cdot g^a\in G_1^*$. If $coin=1$, B computes $h_i= g^a\in G_1^*$.

Step 4: B adds the tuple $<w_i, h_i, a, coin>$ to H_1-*List* and responds to A with $H_1(w_i)=h_i$.

H_3-Queries: B maintains a list of tuples called H_3-*List*, in which each entry is a tuple of the form $<t, v>$. The list is initially empty. When A issues a query to H_3, B checks if t_i is already on H_3-*List* in the form of $<t_i, v_i>$. If so, B responds to A with $H_3(t_i)=v_i$. Otherwise, B picks a random string $v_i\in\{0,1\}^{\log p}$, adds the tuple $<t_i, v_i>$ to H_3-*List*, and responds to A with $H_3(t_i)=v_i$.

Phase 1: When A issues a query for the trapdoor of keyword w_i, B responds as follows:

Step 1: B initiates H_1-Queries to obtain $h_1\in G_1^*$, where $H_1(w_i)=h_i$. Let $<w_i, h_i, a_i, c_i>$ be the corresponding tuple on H_1-*List*.

1) If $c_i=0$, then B reports a failure and terminates.

2) If $c_i=1$, then $H_1(w_i)=h_i=g^a\in G_1^*$. We define that $T_{w_i} = (v_1)^{a_i} = (g^a)^{a_i}$, $T_{w_i} = (g^{\alpha})^{a_i} = (g^{a_i})^{\alpha} = H_1(w_i)^{\alpha}$. B gives T_{w_i} to A.

Challenge: Once A decides that Phase 1 is over, it outputs a pair of keywords w_0 and w_1 on which it wishes to be challenged. B responds as follows:

Step 1: B initiates H_1-Queries twice to obtain h_0 and $h_1\in G_1^*$, where $H_1(w_0)=h_0$ and $H_1(w_1)=h_1$. If $c_0=1$ or $c_1=1$, then B reports a failure and terminates.

Step 2: If both $c_0=0$ and $c_1=0$, B randomly picks a $b_1\in\{0,1\}$.

Step 3: B picks a random string $S_1\in\{0,1\}^{\log p}$, and gives the cipher-text $C_1=(v_3, S_1)$ to A. Easy to verify, C_1 is a valid cipher-text for w_{b_1} as required.

Phase 2: *A* can continue issuing more trapdoor queries for keyword w_i, where the only restriction is that $w_i \neq w_0$ and $w_i \neq w_1$. *B* responds as in Phase 1.
Guess: *A* outputs its guess $b_1' \in \{0,1\}$for b_1. *B* picks a random pair $<t_i, v_i>$ from H_3-*List* and outputs t_i as the solution to D_1.
To complete the proof, we now show that *B* correctly outputs D_1 with the probability at least $\varepsilon_1' = 2\varepsilon_1/\{e \cdot q_{H_3} \cdot (1+q_T)\}$. In the first place, we calculate the probability that *B* does not abort during the above process. Suppose *A* makes a total of q_T trapdoor queries. Then the probability *B* does not abort in Phase 1 or 2 is δ^{q_T}. And the probability that it does not abort during the challenge step is 1-δ. Therefore, the probability that *B* does not abort during the whole process is $\delta^{q_T} \cdot (1-\delta)$. This value is maximized at $\delta_{opt} = 1 - 1/(q_T+1)$. Using δ_{opt}, the probability that *B* does not abort is at least $1/\{e \cdot (q_T+1)\}$. In the second place, we calculate the probability that *B* outputs the correct result in case that *B* does not abort. Let Q_1 be the event that *A* issues a query for *v*. If $\neg Q_1$, we know that the decryption of the cipher-text is independent of *A*'s view. Let $\Pr[b_1=b_1']$ be the probability that *A* outputs the correct result, therefore in the real attack $\Pr[b_1=b_1'|\neg Q_1]=1/2$. Since has *A* the advantage ε_1, $|\Pr[b_1=b_1'|\neg Q_1]-1/2| \geq \varepsilon_1$. According to the following formulae, we know $\Pr[Q_1] \geq 2\varepsilon_1$.

$$\Pr[b_1=b_1'] = \Pr[b_1=b_1'|\neg Q_1]\Pr[\neg Q_1] + \Pr[b_1=b_1'|Q_1]\Pr[Q_1]$$

$$\leq \Pr[\neg Q_1]/2 + \Pr[Q_1] = (1+\Pr[Q_1])/2$$

$$\Pr[b_1=b_1'] \geq \Pr[b_1=b_1'|\neg Q_1]\Pr[Q_1] = \Pr[\neg Q_1]/2 = (1-\Pr[Q_1])/2$$

Therefore, we have that $\Pr[Q_1] \geq 2\varepsilon_1$ in the real attack. Now we know that *A* will issue a query for *v* with the probability at least $2\varepsilon_1$. That is to say, the probability that *v* appears in some pair on H_3-*List* is at least $2\varepsilon_1$. *B* will choose the correct pair with the probability at least $1/q_{H_3}$ and thus *B* produces the correct answer with the probability at least $2\varepsilon_1/q_{H_3}$. Since *B* does not abort with the probability at least $1/\{e \cdot (q_T+1)\}$, we see that *B*'s success probability is at least $\varepsilon_1' = 2\varepsilon_1/\{e \cdot q_{H_3} \cdot (1+q_{H_3})\}$ as required.

5 Functional Comparison

We compare this article with the literature [3,15,16] in two aspects shown in Table 1: functional and storage costs. Functional is compared from two aspects: whether there is a deletion operation and whether it has an authentication function. and the storage cost refers to the cipher text component uploaded by user to server.

Table 1. Scheme comparison table

scheme/ functional	deletion	authentication	Ciphertext
literature[3]	Y	N	ID_m, C_w, C_m
literature[15]	N	N	C_T, C_w, C_m
literature[16]	N	N	C_w, C_m
Our scheme	Y	Y	C_1, C_2, C_m

6 Outlook

This paper study dynamic searchable encryption scheme in the public key cryptosystem, and introduce authentication technology in the process of implementing the dynamic searchable encryption to prevent malicious users from malicious deletion. The scheme proposed in this paper is only limited to the implementation of dynamic deletion and search functions, and has not yet fully implemented the dynamic operation of the update function. The next step will be to study the update function of dynamic searchable encryption based on public keys, and truly implement the function of deletion and updata files to users.

7 References

[1] Li JW, Jia CF, Liu ZL, Li J, Li M. Journal of Software, **26(1)**,109(2015).

[2] Shen ZR, Xue W, Shu JW. Journal of Software.**25(4)**,880 (2014).
[3] Jin H, Gao X, Xu P. CN105553660A(2016).
[4] Dan B, Crescenzo G D, Ostrovsky R, et al. International Conference on the Theory and Applications of Cryptographic Techniques.506(2004).
[5] Wu Qi. Computer Engineering, **42(8)**,123(2016).
[6] Kamara S, Papamanthou C, Roeder T. ACM, 965(2012).
[7] Yang Y, Li H, Liu W, et al. Global Communications Conference. 775 (IEEE,2015).
[8] Kamara S, Papamanthou C. International Conference on Financial Cryptography and Data Security. 258(Springer 2013).
[9] Yavuz A A, Guajardo J. International Conference on Selected Areas in Cryptography. 241(Springer 2015).
[10] He XT, Miao FY, Fang L. Computer Engineering, **44(1)**,154(2018).
[11] Wang LH. Nanjing University of Science and Technology(2007).
[12] XU L,XU C G,YU X L. Journal of Cryptologic Research,**3(4)**,330(2016).
[13] ZHANG F G. Journal of Cryptologic Research, **3(3)**,211(2016).
[14] Fang LM. Nanjing University of Aeronautics and Astronautics(2012).
[15] Liu Q, Wang G, Wu J. International Conference on Computational Science and Engineering. 715(IEEE, 2009).
[16] Song D X, Wagner D, Perrig A. IEEE Symposium on Security and Privacy.44(IEEE Computer Society,2000).