Online Resource 4

1 Update Reliable SI-SSE-RV

Update Reliability [1]: For an adversary, the probability of successfully giving the non updated search result that can pass the verification test, is negligible.

To assure update reliability, we propose **Update Reliable SI-SSE-RV** by redefining the Encryption(), Search() and Verify() algorithm of SI-SSE-RV as shown in Table 1. From the formal construction of Update Reliable SI-SSE-RV, it could be observed that with every insert/delete/update operation, the data owner has to issue the new value of global counter Ver_{new} to server and force it to replace the existing value of Ver.

The Update Reliable SI-SSE-RV could support multiple users with additional communication overhead between data owner and data user. In particular, to work with Update Reliable SI-SSE-RV, each user must have to collect the latest value of global counter Ver from data owner. If we consider 2 messages (request and reply) per data user in such process of collecting counter value, then for u number of users, the overall communication cost would be $(2 \cdot u)$. However, with such additional overhead, the scheme could offer multi-user support.

2 Security against replay attack

with the following Theorem 1, we prove that the given extension Update Reliable SI-SSE-RV is indeed update reliable and it offers resistance against replay attack.

Theorem 1 *The proposed Update Reliable SI-SSE-RV assures update reliability and resists replay attack.*

Proof : Let the data owner maintains a global counter Ver at local-site as well as at the server-site. For each document insertion/deletion/modification, data owner updates Ver at local site. He then sends the updated Ver to server and asks it to replace the old Ver. Suppose the server as an adversary $\mathcal{A}$ does not follow the update request for Ver and afterwards performs replay attack [1] by forwarding the non-updated data as the search result. With the proposed Update Reliable SI-SSE-RV, the data owner could easily identify the dishonest behavior of $\mathcal{A}$ and avoids replay attack as follows:

Let Ver_{new} is the global counter value available to data owner as per the last update. The data owner also has the corresponding value for a local variable U_{new}. Suppose $\mathcal{A}$ at the end of the *UR_Search()* algorithm returns a global counter value Ver_{old} which is a non-updated value.

As per *UR_Encryption()* algorithm, Ver is updated for each new file. Thus, after $|D|$ number of documents inserted in data collection, the last global counter would be $Ver_{new} = (U_{new} + |D| + \sum_{i \leq |D|} S'_{2i})$ and the corresponding $U_{new} = (U + |D|)$. Such Ver_{new} should be available at server besides encrypted document collection.

Additionally, as per *UR_Search()* algorithm, suppose $\mathcal{A}$ computes search result Res and identifies p number of files matching with query and q number of unmatched files where $(p+q) = |D|$. $\mathcal{A}$ then computes the global proof components (P_1, P_2) where $P_1 = \sum_{i<=p} S'_{i2}$ and $P_2 = \sum_{j<=q} S'_{j2}$. Finally, $\mathcal{A}$ should return the search result Res along with the global proof components (P_1, P_2) and an available global counter value Ver_{new}.

Formal Construction: Update Reliable SI-SSE-RV

Algorithms

Let $SKE=(Enc, Dec, key)$ be a symmetric key encryption algorithm with key key. Let G_1 and G_2 be two groups of prime order p and e be a non-degenerate bilinear map, i.e. e: $G_1 \times G_2 \rightarrow G_T$ that holds XDH assumption across G_1 and G_2. Let g_K: $\{0,1\}^\lambda \times \{0,1\}^* \rightarrow Z_p$ be a keyed hash function and is denoted as $g_K(\cdot)$. **Let** $Ver, U_{old}, U_{new} \in G_1$ **are group elements. Let** $Ver = U_{old} = U$ **where** U **is a random element from group** G_1.

1. **UR_Setup(**$1^\lambda, n$**)** Same as *Setup()* of SI-SSE-RV.
2. **UR_Encryption(**D, W, SK**)** To compute ciphertext C from input payload data D having identifier $ID \in \{0,1\}^*$, set of keywords $W = \{w_1, w_2, ..., w_n\}$ and secret key SK.
   - **Document Encryption :**
     - Compute $D' = SKE.Enc_{key}(D||I_{ID})$ where I_{ID}=$rg_K(ID)P$.
   - **Build Simple Index :**
     - Compute $I_0 = rP$ where $r \in Z_p$.
     - For each $w_j \in W$, find $s_j = g_K(w_j)$.
     - For $1 \leq j \leq n$, compute $I_j = rs_jP$.
     - Set a simple index $I = \{I_0, I_1, ..., I_n\}$.
   - **Server-side Verification Component Computation :**
     - Compute $S'_1 = \sum_{j=1}^{n} I_j + rg_K(s)P$ and $S'_2 = S'_1 + I_{ID}$.
   - **Global Counter computation :**
     - **Compute** U_{new}=$(U_{old}+1)$ **and** Ver=$Ver + U_{new} - U_{old} + S'_2$.
   - **Output :** ***A ciphertext*** $C = (D', S'_1, S'_2, I)$ ***along with the global counter*** Ver.
3. **UR_Token(**Q', SK**)** Same as *Token()* of SI-SSE-RV.
4. **UR_Search(**C, T**)** To perform search across ciphertext $C_i \in C$ using token T where $C_i = (D'_i, S'_{i1}, S'_{i2}, I_i)$ and $T = \{t_0, t_1, I'\}$. Let a result set $Res = \perp$. **Let** $P_1, P_2 \in G1$ **be two global proof components**.
   - **Conjunctive Search :** Check search relevancy with
   $$e(I_{i0}, t_1) = e(\sum_{j \in I'} I_{ij}, t_0) \quad (1)$$
   - **Proof Component computation :** If Eq. (1) holds,
     - Compute a proof component $\pi_i = (R_{i1}, R_{i2}, I_{i0})$ where $R_{i1} = S'_{i1}$ - R', $R_{i2} = S'_{i2}$ - R' and $R' = \sum_{j \in [1...n] \setminus I'} I_{ij}$.
     - Set $Res_i = (D'_i, \pi_i)$ and $Res = (Res||Res_i)$.
     - **Set** P_1=$P_1 + S'_{i2}$.
   - **If Eq. (1) does not hold,**
     - **Set** P_2=$P_2 + S'_{i2}$.
   - **Output : A result set** Res **along with global proof components** (P_1, P_2) **and global counter value** (Ver).
5. **UR_Verify(**Res, T, t'**)** To verify the search result Res for available token T and verification component t'.
   - Let a verification output $O = \perp$.
   - For each $Res_i \in Res$ where $Res_i = (\cdot, \pi_i = (R_{i1}, R_{i2}, I_{i0}))$
     - **Correctness Check :**
       - If $Res_i = (D'_i, \pi_i)$, then compute $(D_i, I_{ID}) = SKE.D_{key}(D'_i)$ and $R = R_{i2}$ - I_{ID}.
       - If $Res_i = (\perp, \pi_i)$, then set $R = R_{i1}$ (In case of result alteration).
       - Find $SR = (e(R, t_0) = e(I_{i0}, t'))$.
     - **Generate verification output :**
       - If $(SR = 1)$ Then
         - If $(Res_i = (D'_i, \pi_i))$ Then $O_i = 1$.
         - Else If $(Res_i = (\perp, \pi_i))$ Then $O_i = 0$.
       - Else
         - If $(Res_i = (D'_i, \pi_i))$ Then $O_i = 0$.
       - Set O=$(O||O_i)$.
   - **Output :**
     - **If (((**$O_i = 1$**) for each** $O_i \in O$**) AND (**$(U_{new} + P_1 + P_2) = Ver$**))**
     - **Then Output 'Correct'**.
     - **else Output 'Incorrect'**.

As per *UR_Verify()* algorithm, data owner first checks the correctness of search result Res and accordingly generates output O_i for each result forwarded by $\mathcal{A}$. If $O_i = 1$ for each result, then from the available (U_{new}, P_1, P_2), the data owner checks equality $(U_{new} + P1 + P2) = Ver_{new}$. If $\mathcal{A}$ follows the protocol and updates the data as well as global counter Ver as per the requirement by data owner, such equality is being satisfied. However, if any dishonest activity is performed by $\mathcal{A}$, then the data owner could easily identify it using *UR_Verify()* algorithm as follows:

The potential dishonest activities performed by $\mathcal{A}$ are : (i) $\mathcal{A}$ updates the requested document but does not update Ver, (ii) $\mathcal{A}$ does not update the requested document but updates Ver, (iii) $\mathcal{A}$ neither updates the document nor the global counter Ver.

Let us assume that $\mathcal{A}$ has an old global counter value Ver_{old}. The data owner makes update request for document D_{new} with a new global counter value Ver_{new}. If $\mathcal{A}$ updates the requested file but does not update the global counter Ver (activity (i)) then at the end of the *UR_Search()* algorithm the global verification components P_1 or P_2 includes new S_2' component for D_{new}. However, the global counter value includes old S_2' component for D_{new}. Hence, during verification, the global component equality checking does not satisfy i.e. $(U_{new} + P1 + P2) \neq Ver_{new}$. Thus, data owner identifies the problem in data update. Similarly, if $\mathcal{A}$ updates Ver with Ver_{new} but does not update requested document (activity (ii)) then the verification components P_1 or P_2 includes old S_2' component for D_{new} whereas global counter includes new S_2' component for D_{new}. In such a case, data owner once again gets $(U_{new} + P1 + P2) \neq Ver_{new}$ and identifies update problem. Besides these, if $\mathcal{A}$ neither updates the document nor the global counter Ver (activity (iii)), then both the verification components and global counter value include old S_2' for D_{new}. At the same time, with the Ver_{old}, $\mathcal{A}$ has U_{old}. However, the latest U_{new} available to data owner includes the exact number of updates requested by data owner. Hence, $(U_{new} + P1 + P2) \neq Ver_{new}$ and data owner determines update problem.

From the above discussion, it can be said that with update reliable SI-SSE-RV, no adversary $\mathcal{A}$ could convince data owner for the correctness of the non-updated data. Hence, we claim that the extended construction of SI-SSE-RV ensures update reliability and resists replay attack.

References

[1] Xinrui Ge, Jia Yu, Hanlin Zhang, Chengyu Hu, Zengpeng Li, Zhan Qin, and Rong Hao. Towards achieving keyword search over dynamic encrypted cloud data with symmetric-key based verification. *IEEE Transactions on Dependable and Secure Computing*, 2019.