# Online Resource 3

## 1 SI-SSE-RV : Formal Construction

The proposed algorithms are formally defined in Table 1. With *Setup()* algorithm, the data owner sets up a secret key $SK$ used for document encryption, index building, token generation as well as for result verification. He then distribute $SK$ to each authorized data user using any key distribution algorithm (that is either using broadcast encryption [1] used in [2] or using access control based mechanism used [3, 4]). Once key has been setup, the data owner computes a ciphertext $C$ for the given document $D$ and its associated set of keywords $W$ using *Encryption()* algorithm. The ciphertext includes encrypted document $D'$, simple index $I$ as well as server-side verification components $(S'_1, S'_2)$. Such verification components are used by the server to generate a proof component for each output ciphertext. A data user (or data owner) possessing $SK$ can compute a search token $T$ from the chosen conjunctive query $Q$ using *Token()* algorithm. Besides token, this algorithm generates a local verification component $t'$ that further used by *Verify()* algorithm. The server on the availability of token $T$, performs search across all available ciphertexts $C_i \in \mathcal{CC}$ using *Search()* algorithm. With this algorithm, the server identifies the set $Res$ of ciphertexts matching with the requested query. For each ciphertext in $Res$, the server computes a proof component using $(S'_1, S'_2)$ associated with that ciphertext. With the proposed *Verify()* algorithm, the data user can check the correctness of the returned search result $Res$. To check such correctness, it uses the token $T$, local verification component $t'$ and secret key $SK$. The output generated by *Verify()* algorithm includes 1/0 for each result in $Res$ according to their correctness.

**Correctness:** For a given ciphertext $C_i = (D'_i, S'_{i1}, S'_{i2}, I_i)$ where $I_i = (I_{i0} = r_i P, I_{ij} = r_i s_j P = r_i g_k(w'_j)P)$ with $j \in [1...n]$ and a token $T = (t_0 = \rho Q, t_1 = \rho(\sum g_k(w'_j))Q, I')$ with $w'_j \in W'$ and $j \in I'$, the correctness of *Search*$(C_i, T)$ algorithm is proved as follows:

L.H.S of Eq. (3)

$$\begin{aligned} e(I_{i0}, t_1) &= e(r_i P, \rho(\sum g_k(w'_j))Q) \\ &= e(P, (\sum g_k(w'_j))Q)^{r_i \rho} \\ &= e(P, Q)^{r_i \rho(\sum g_k(w'_j))} \end{aligned} \tag{1}$$

R.H.S of Eq. (3)

$$\begin{aligned} e(\sum_{j \in I'} I_{ij}, t_0) &= e(\sum r_i s'_j P, \rho Q) \\ &= e(r_i \sum g_k(w'_j)P, \rho Q) \\ &= e(\sum g_k(w'_j)P, Q)^{r_i \rho} \\ &= e(P, Q)^{r_i \rho(\sum g_k(w'_j))} \end{aligned} \tag{2}$$

From Eq. (1) and Eq. (2), the correctness of the proposed *Search()* algorithm is proved. Note that the correctness of the proposed *Verify()* algorithm is proved with Theorem 2 in *Section 5.1.*

Table 1: Formal Construction: 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 $g_K(\cdot)$.

1. **Setup($1^\lambda, n$)** To generate a secret key $SK$.
   - Let $(s, K) \leftarrow \{0,1\}^\lambda$ are two random, secret key components. Let $P \in G_1$ and $Q \in G_2$ are generators of groups. Let $\mathcal{KS}$ is a keyword space for $n$ keyword fields where initial value of each keyword is NULL.
   - The algorithm outputs a secret key $SK = \{s, K, Q, key\}$.
2. **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 $S'_2 = S'_1 + I_{ID}$.
   - **Output :** A ciphertext $C = (D', S'_1, S'_2, I)$.
3. **Token($Q', SK$)** To compute token $T$ from secret key $SK$ and conjunctive query $Q' = (W', I')$ where $W' = \{w'_{\ell_1}, w'_{\ell_2}, ..., w'_{\ell_t}\}$ is a list of keywords, $I' = \{\ell_1, \ell_2, ..., \ell_t\}$ is a list of positions of keywords in $\mathcal{KS}$ for $1 \leq t \leq n$.
   - **Token Generation :**
     - Compute $t_0 = \rho Q$, $t_1 = \rho(\sum g_K(w'_j))Q$ where $w'_j \in W'$, $j \in I'$ nd $\rho \in Z_p$.
   - **Verification Component computation :**
     - Compute a verification component $t' = t_1 + \rho g_K(s)Q$.
   - **Output :** A token $T = (t_0, t_1, I')$.
4. **Search($\mathcal{CC}, T$)** To perform search across ciphertext $C_i \in \mathcal{CC}$ 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 $Res = \perp$.
   - **Conjunctive Search :** Check search relevancy with
   $$e(I_{i0}, t_1) = e(\sum_{j \in I'} I_{ij}, t_0) \tag{3}$$
   - **Proof Component computation :** If Eq. (3) 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 $Res = (Res||Res_i)$.
   - **Output :** A result set $Res$.
5. **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 :** A verification output $O$.

# 2 SI-SSE-RV : Security Provision

We first emphasize that the primary objective behind the proposed work is to enable data user to check the correctness of result returned from the malicious server in SI based SSE scheme. With such an objective, the proposed SI-SSE-RV offers result verifiability whereby the server is forced to compute a proof component along with the search result. On the availability of result, data user utilizes a proof component to verify the correctness of result and identifies malicious activity (if any) performed by server in terms of search result alteration or search document tampering.

## 2.1 Attack Model and Security Definition

For security definition of SI-SSE-RV, we define an attack model whereby we consider SS as an adversary $\mathcal{A}$ having capabilities to perform the following attacks.

1. **Chosen Keyword Attack** $\mathcal{A}$ can perform chosen keyword attack to deduce plaintext (keywords) from the available ciphertexts (lists of encrypted keywords) and tokens.
2. **Alteration of query result** $\mathcal{A}$ can perform malicious alteration of query result.
3. **Replay Attack** In case of dynamic data update, $\mathcal{A}$ can perform replay attack by forwarding non-updated data as a search result [5].

We additionally assume that the other potential attacks (viz. leakage due to search pattern, statistical analysis performed by server) can be mitigated by the schemes explicitly defined for mitigation of such attacks.

### 2.1.1 Security against chosen keyword attack

We define semantic security against chosen keyword attack in terms of a game played between polynomial-time algorithms for adversary $\mathcal{A}$ and challenger $\mathcal{C}$. Following [6,7], we present game ICR (Indistinguishability of Ciphertext from Random) as mentioned below:

**Game ICR**

1. Initially, $\mathcal{C}$ runs *Setup*$(1^\lambda)$ to generate a secret key $SK$.
2. $\mathcal{A}$ adaptively requests $\mathcal{C}$ for ciphertext $C$ of his chosen sets of keywords. $\mathcal{A}$ also asks for token $T_{W'}$ for his chosen conjunctive sets of keywords $W'$.
3. $\mathcal{A}$ sends $\mathcal{C}$ a list $W_0$ of his chosen keywords and $W_1$ of some random keywords with the restriction $|W_0| = |W_1|$ and none of the tokens $T_{W_0}$ and $T_{W_1}$ asked previously in step 2. $\mathcal{C}$ randomly selects $b \in \{0,1\}$ and gives $\mathcal{A}$ a challenge ciphertext $C' = Encryption(W_b, SK)$.
4. $\mathcal{A}$ again asks for ciphertext and token for his chosen set of keywords $W''$ such that $W'' \notin \{W_0, W_1\}$. A total number of ciphertexts and token requests is polynomial in $\lambda$.
5. Eventually, $\mathcal{A}$ outputs $b' \in \{0,1\}$ and wins the game if $b' = b$.

The adversary $\mathcal{A}$ wins game ICR if he can correctly guess whether he was given $C'$ for $W_0$ or $W_1$. We define $\mathcal{A}$'s advantage as

$$Adv_{\mathcal{A}}^{ICR}(1^\lambda) = |Pr[b' = b] - \frac{1}{2}|$$

**Definition 1**: We say that the proposed SI-SSE-RV is semantically secure against adaptive chosen keyword attack, according to ICR if for any polynomial time adversary $\mathcal{A}$, we have that $Adv_{\mathcal{A}}^{ICR}(1^\lambda)$ is a negligible function.

#### 2.1.2 Security against malicious alteration of query result

We assume that adversary $\mathcal{A}$ can modify query result in either way:

- *Search Result modification*
  **Case 1** For an unmatched ciphertext $C_i$ (i.e. $C_i$ does not satisfy token $T$), $\mathcal{A}$ returns $Res_i = (D'_i, \pi_i)$ instead of $Res_i = (\perp, \pi_i)$.
  **Case 2** For a matched ciphertext $C_i$ (i.e. $C_i$ satisfies token $T$), $\mathcal{A}$ returns $Res_i = (\perp, \pi_i)$ instead of $Res_i = (D'_i, \pi_i)$.
- *Search Document tampering*
  For any matched/unmatched ciphertext $C_i$, $\mathcal{A}$ returns $Res_i = (D'_k, \pi_i)$ instead of $Res_i = (D'_i/\perp, \pi_i)$ where $k \neq i$.

We define a verification component $\pi_i$ in such a way that any deliberate alteration of $Res_i$ can be identified by DU.

**Definition 2**: We say that the proposed SI-SSE-RV is secure against malicious alteration of query result performed by adversary $\mathcal{A}$ if proof $\pi$ is sufficient to verify the correctness of returned result.

#### 2.1.3 Security against replay attack

With dynamic data update, the data owner attempts to update (insert or delete or modify) a document residing at the server. We assume that the adversary $\mathcal{A}$ does not follow the protocol honestly and avoids the requested data update. Furthermore, he performs replay attack [5] by forwarding the non-updated data in the search result.
To avoid such an attack, we extend the basic construction of SI-SSE-RV and propose Update Reliable SI-SSE-RV where no adversary would be able to convince data verifier for the correctness of the non-updated data [5].
**Definition 3**: We say that the proposed extension of SI-SSE-RV is indeed update reliable and hence resists replay attack.

### 2.2 Security Analysis

**Theorem 1** *The proposed SI-SSE-RV is semantically secure against chosen keyword attack, assuming MXDH holds.*

**Proof :** Let $\mathcal{A}$ be an adversary that wins game ICR with advantage $\epsilon$. We define an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as subroutine to break MXDH assumption with non-negligible advantage. Let $P \in G_1$, $Q \in G_2$, $e$: $G_1 \times G_2 \rightarrow G_T$ be an XDH setting and let $(P, aP, bP, cP, aQ, bQ)$ be $\mathcal{B}$'s MXDH challenge. For $\mathcal{B}$, a challenge is to determine whether $c = ab$, or to solve DDH problem in $G_1$. The algorithm $\mathcal{B}$ starts simulating $\mathcal{A}$ as follows:

- **Encryption Queries** $\mathcal{A}$ makes polynomial number of encryption queries to $\mathcal{B}$. $\mathcal{B}$ answers them as follows:
  - Let $\mathcal{A}$'s query includes a set $W_i = (w_{i1}, w_{i2}, ..., w_{in})$ where $0 \leq i \leq poly(\lambda)$.
  - $\mathcal{B}$ chooses random values $r_i \in Z_p$ and computes $I_{i0} = r_i P$.
  - Then for each $w_{ij} \in W_i$, $\mathcal{B}$ randomly selects $z_{ij} \in Z_p$ for all $j \in [1, n]$ and computes $I_{ij} = r_i z_{ij} bP$.
  - $\mathcal{B}$ maintains consistency amongst values of $z_{ij}$ for different $w_{ij}$ i.e. if $w_{ij} = w_{i'j}$ (where $i \neq i'$) then the same $z_j$ is chosen.
  - Finally, $\mathcal{B}$ returns $C_i$ with $I_i = (I_{i0}, I_{i1}, ..., I_{in})$.
- **Token Queries** $\mathcal{A}$ makes token request for query $Q' = (W', I')$ where $W' = (w'_{\ell_1}, w'_{\ell_2}, ...w'_{\ell_t})$ and $I' = (\ell_1, \ell_2, ..., \ell_t)$. $\mathcal{B}$ responses as follows:
  - $\mathcal{B}$ randomly chooses $\rho \in Z_p$ and computes $t_0 = \rho Q$, $t_1 = \rho(\sum \gamma_k bQ)$ for all $k \in [1, t]$.
  - Here $\gamma_k = z_{ij}$ if $w'_k$ is already asked by $\mathcal{A}$ in any of the previous ciphertext or token queries, otherwise $\gamma_k \in Z_p$.
  - Finally, $\mathcal{B}$ returns a token $T = (t_0, t_1, I')$.

Here, we observe that $T$ is a valid token for $W'$ since $\mathcal{B}$ consistently uses the same value $z_{ij}$ for word $w_j$. The $Search(C_i, T)$ operation will be successful if and only if $W_i$ includes all keywords that are present in $W'$ at the location specified by $I'$.

- **Challenge** $\mathcal{A}$ issues two lists of keywords $W_0 = (w^*_{01}, w^*_{02}, ..., w^*_{0n})$ and $W_1 = (w^*_{11}, w^*_{12}, ..., w^*_{1n})$. Here $W_0$ is a list of known keywords whereas $W_1$ is a list of random keywords. $\mathcal{B}$ responses as follows:
  - $\mathcal{B}$ randomly selects $\beta$=$\{0, 1\}$ and sends a challenge ciphertext $C_\beta = (I_{\beta 0}, I_{\beta 1}, I_{\beta 2}, .., I_{\beta n})$ where $I_{\beta 0} = aP$ and $I_{\beta k} = \gamma^*_k cP$ for all $k \in [1, n]$. Here, $\gamma^*_k = z_{ij}$ if $w_k$ is previously asked by $\mathcal{A}$ in any of the ciphertexts or token queries, and $\gamma^*_k \in Z_p$ otherwise.
  - $C_\beta$ is the correct ciphertext for $W_0$ if $c = ab$. On the other hand, if $c \neq ab$, then $C_\beta$ is a correct ciphertext for $W_1$. This implies that if $c$ is random then $C_\beta$ involves an index for random set of keywords.
- **Repeat Encryption and Token Queries** Once a challenge ciphertext $C_\beta$ is available, $\mathcal{A}$ can make another encryption queries and token queries and $\mathcal{B}$ answers as above.
- **Guess** Finally, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ that shows its decision as to whether $C_\beta$ is a ciphertext for $W_0$ or for $W_1$. $\mathcal{A}$ takes this decision as follows:

From the available token $T = (t_0, t_1, I')$ and a challenge ciphertext $C_\beta = (I_{\beta 0}, I_{\beta 1}, I_{\beta 2}, .., I_{\beta n})$, $\mathcal{A}$ checks the search equality as follows:
From Eq. (3),

$$e(I_{\beta 0}, t_1) = e(\sum_{j \in I'} I_{\beta j}, t_0) \tag{4}$$

L.H.S. of Eq. (4),

$$e(I_{\beta 0}, t_1) = e(aP, \rho(\sum \gamma_k bQ)) \tag{5}$$

$$= e(P, Q)^{ab\rho \sum \gamma_k} \tag{6}$$

R.H.S. of Eq. (4),

$$e(\sum_{j \in I'} I_{\beta j}, t_0) = e(\sum_{j \in I'} \gamma^*_k cP, \rho Q) \tag{7}$$

$$= e(P, Q)^{c\rho \sum \gamma^*_k} \tag{8}$$

From Eq. (6) and Eq. (8),

$$e(P, Q)^{ab\rho \sum \gamma_k} = e(P, Q)^{c\rho \sum \gamma^*_k} \tag{9}$$

Here, if $ab = c$, then $C_\beta$ is considered as correct ciphertext for $W_0$ and so $\mathcal{A}$ outputs $b' = 0$, otherwise $C_\beta$ is considered as a ciphertext for random keyword set $W_1$ and $\mathcal{A}$ outputs $b' = 1$. $\mathcal{B}$ returns this answer of $\mathcal{A}$ (i.e. $b'$) as its own answer to its MXDH challenge.

However, according to the discussion of [6], $\mathcal{B}$'s advantage of breaking MXDH challenge is equal to $\mathcal{A}$'s advantage of breaking ICR game. If MXDH assumption holds, then the advantage of $\mathcal{A}$ i.e. $\epsilon$ is negligible and so the advantage of $\mathcal{B}$ is also negligible. Thus, we say that the proposed SI-SSE-RV is semantically secure against chosen keyword attack. This completes the proof of *Theorem 1.*

**Theorem 2** *The proof $\pi$ is sufficient to verify the correctness of result returned by malicious adversary $\mathcal{A}$.*

We give two separate proofs for two types of result alteration performed by $\mathcal{A}$, that is **Proof 1** : Search result modification, **Proof 2** : Search Document tampering.

**Proof 1:**

***Case 1:*** Let us assume that a ciphertext $C_i$ does not satisfy the requested conjunctive search query i.e. the equality check of Eq. (3) does not hold. In such a case, consider a malicious action of $\mathcal{A}$ whereby it outputs $Res_i$=$(D'_i, \pi_i)$ instead of $Res_i$=$(\perp, \pi_i)$ and tries to convince DU for the result correctness.

As per the proposed *Search()* algorithm, $\mathcal{A}$ uses an available $C_i$=$(D'_i, S'_{i1}, S'_{i2}, I_i)$ and $T = (t_0, t_1, I')$ to compute proof component $\pi_i = (R_{i1}, R_{i2}, I_{i0})$ as follows:

- $\mathcal{A}$ computes

$$R' = \sum I_{ij} = r \sum g_K(w_{ij}),\ j \in [1...n] \setminus I' \tag{10}$$

- Then from available

$$\begin{aligned} S'_{i1} &= \sum I_{ij} + r g_K(s)P, \quad j \in [1...n] \\ &= r[\sum g_K(w_{ij}) + g_K(s)]P \\ S'_{i2} &= S'_{i1} + I_{ID_i}, \quad j \in [1...n] \\ &= r[\sum g_K(w_{ij}) + g_K(s)]P + r g_K(ID_i)P \end{aligned}$$

- $\mathcal{A}$ computes

$$\begin{aligned} R_{i1} &= S'_{i1} - R', \quad j \in I' \\ &= r[\sum g_K(w_{ij}) + g_K(s)]P \\ R_{i2} &= S'_{i2} - R', \quad j \in I' \\ &= r[\sum g_K(w_{ij}) + g_K(s)]P + I_{ID_i} \end{aligned}$$

As $g_K(s)$ involves a secret key $s \in Z_p$ (known to DO or DU only), there is no alternate way for $\mathcal{A}$ to find the above $(R_{i1}, R_{i2})$.

At the requesting user's site, there exist token components $(t_0, t_1)$ as well as verification component $t'$ as follows:

$$\begin{aligned} t_0 &= \rho Q, \quad t_1 = \rho(\sum g_K(w_j))Q \\ t' &= t_1 + \rho g_K(s)Q, \quad j \in I' \\ &= \rho(\sum g_K(w'_j) + g_K(s))Q \end{aligned}$$

The verification of search result is as follows:

- As $Res_i$ includes document $D'_i$, first $I_{ID_i}$ is extracted by $(D_i, I_{ID_i}) = SKE.Dec(D'_i)$.
- $R = R_{i2} - I_{ID_i}$
- Then according to *Verify()* algorithm, the check is performed as

$$e(R, t_0) = e(I_0, t') \tag{11}$$

L.H.S. of Eq. (11)

$$\begin{aligned} e(R, t_0) &= e(r[\sum g_K(w_{ij}) + g_K(s)]P, \rho Q) \\ &= e(P, Q)^{r\rho(\sum g_K(w_{ij}) + g_K(s))} \end{aligned} \tag{12}$$

R.H.S. of Eq. (11)

$$e(I_0, t') = e(rP, \rho(\sum g_K(w'_j) + g_K(s))Q)$$
$$= e(P, Q)^{r\rho(\sum g_K(w'_j)+g_K(s))} \quad (13)$$

Here, Eq. (11) holds (i.e. L.H.S = R.H.S.), if and only if $w_{ij} = w'_j$ for $j \in I'$. However, as per initial assumption in this proof (i.e. the search query is not satisfied), Eq. (3) does not hold and hence equations (11) can not hold. As per *Verify()* algorithm, if Eq. (11) does not hold and still $Res_i$ includes $D_i$, the output $O = 0$. This proves the incorrectness of the returned result.

***Case 2:*** Let us assume that the requested conjunctive search query is satisfied by ciphertext $C_i$. However, $\mathcal{A}$ returns $Res_i$=$(\perp, \pi_i)$ instead of $Res_i$=$(D'_i, \pi_i)$.

Here, adversary $\mathcal{A}$ computes $\pi_i = (R_{i1}, R_{i2}, I_{i0})$ as in *Case 1.* Additionally, DU has $(t_0, t_1, t')$. The *Verify()* algorithm works as follows:

- As $Res_i$ includes $\perp$, $R = R_{i1}$
- Then the check is performed as

$$e(R, t_0) = e(I_0, t') \quad (14)$$

Here, as per our assumption in this case i.e. $w_{ij} = w'_j$ and so Eq. (14) holds. However, $Res_i$ includes $\perp$ and thus the proposed *Verify()* algorithm sets $O = 0$. This proves incorrectness of result.

**Proof 2:**

Let us assume that for search across any $C_i$ (i.e. matched or unmatched), adversary $\mathcal{A}$ outputs $Res_i$=$(D'_k, \pi_i)$ instead of $Res_i$=$(D'_i/\perp, \pi_i)$ where $k \neq i$.

Here, according to the proposed *Search()* algorithm, $\mathcal{A}$ computes $\pi_i = (R_{i1}, R_{i2}, I_{i0})$ as in *Case 1.* As per the initial assumptions (Section **??**), $\mathcal{A}$ follows the protocol i.e. it does not change the operations inside any algorithm. Thus, $\pi_i$ is computed using $(S'_{i1}, S'_{i2})$ only.

Having token components $(t_0, t_1)$ and verification component $t'$, the proposed *Verify()* algorithm works at the user's site as follows:

- As $Res_i$ includes document $D'_k$, it extracts $I_{ID_k}$ with $(D_k, I_{ID_k}) = SKE.D(D'_k)$.
- Then computes $R$

$$R = R_{i2} - I_{I_{ID_k}}$$
$$R = \sum_{j=1}^{n} I_j + rg_K(s)P + rg_K(ID_i)P - rg_K(ID_k)P$$

- Then checks

$$e(R, t_0) = e(I_0, t') \quad (15)$$

- L.H.S. of Eq. (15)

$$e(R, t_0) = e(r[\sum g_K(w_{ij}) + g_K(s) + g_K(ID_i) + g_K(ID_k)]P, \rho Q)$$
$$= e(P, Q)^{r\rho(\sum g_K(w_{ij})+g_K(s)+g_K(ID_i)+g_K(ID_k))} \quad (16)$$

- R.H.S. of Eq. (15)

$$\begin{aligned} e(I_0, t') &= e(rP, \rho(\sum g_K(w'_j) + g_K(s))Q) \\ &= e(P, Q)^{r\rho(\sum g_K(w'_j) + g_K(s))} \end{aligned} \tag{17}$$

Here, Eq. (15) doesn't hold since L.H.S. $\neq$ R.H.S. . However, still $Res_i$ includes $D'_k$, and so *Verify()* algorithm sets $O = 0$. This proves the incorrectness of the returned result.

From **Proof 1** and **Proof 2**, we say that adversary $\mathcal{A}$ with proof component $\pi$ can not convince DU for the correctness of result for any unmatched ciphertext or any tampered document. This proves that proof $\pi$ is sufficient to verify the correctness of query result.

# References


[1] Amos Fiat and Moni Naor. Broadcast encryption. In *Annual International Cryptology Conference*, pages 480–491. Springer, 1993.

[2] Reza Curtmola and Ostrovsky Garay, Kamara Seny. Searchable symmetric encryption: improved definitions and efficient constructions. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 79–88. ACM, 2006.

[3] Jun Ye, Jianfeng Wang, Jiaolian Zhao, Jian Shen, and Kuan-Ching Li. Fine-grained searchable encryption in multi-user setting. *Soft Computing*, pages 1–12, 2016.

[4] Shangping Wang, Xiaoxue Zhang, and Yaling Zhang. Efficiently multi-user searchable encryption scheme with attribute revocation and grant for cloud storage. *PloS one*, 11(11):e0167157, 2016.

[5] 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.

[6] Lucas Ballard, Seny Kamara, and Fabian Monrose. Achieving efficient conjunctive keyword searches over encrypted data. In *Information and Communications Security*, pages 414–426. Springer, 2005.

[7] Philippe Golle, Jessica Staddon, and Brent Waters. Secure conjunctive keyword search over encrypted data. In *Applied Cryptography and Network Security*, pages 31–45. Springer, 2004.