A Note on "Privacy-Preserving Multi-Keyword Searchable Encryption for Distributed Systems"

Zhengjun Cao[1], Lihua Liu[2]

Abstract. We show that the searchable encryption scheme [IEEE Trans. Parallel Distrib. Syst., 32 (3), 2021, 561–574] cannot work because the Data Provider's secret key sk_{DP} and the Request User's secret key sk_{RU} are not available to the Cloud Platform (CP) or the Internal Server (IS). The CP and IS cannot finish the secure bit-decomposition protocol, which requires CP or IS to decrypt the blinded integer so as to securely handle the least significant bit of the target integer.

Keywords: Encryption of least-significant bit, Paillier cryptosystem, Searchable Encryption, Secure bit-decomposition.

1 Introduction

Secure bit-decomposition (SBD) is a process of securely converting an encrypted value of x into encryptions of the individual bits of x, which requires at least two semi-honest parties. In 2013, Samanthula *et al.* [1] designed a SBD protocol based on Paillier cryptosystem [2], which assumes that Alice generates a Paillier public/secret key pair (pk, sk) and Bob holds the encrypted value $E(x)$, where x is not known to Alice and Bob. Suppose $\langle x_0, \cdots, x_{m-1}\rangle$ denotes the binary representation of x where x_0 and x_{m-1} are the least and most significant bits respectively. At the end of the SBD protocol, the values $E(x_0), \cdots, E(x_{m-1})$ are known only to Bob and nothing is revealed to Alice.

Recently, Liu *et al.* [3] have presented a privacy-preserving multi-keyword searchable encryption scheme based on the Samanthula *et al.*'s SBD protocol. The scheme requires the Cloud Platform (CP) to store documents and searchable ciphertexts uploaded by each data provider (DP), and handle search queries with the assistance from each Internal Server (IS) to generate the encrypted searching result for each Request User (RU). In the process CP needs to run the SBD protocol with IS, under the target DP's public key pk_{DP} and the RU's public key pk_{RU}. We find the scheme cannot work because the corresponding secret keys sk_{DP} and sk_{RU} are not available to CP or IS. It has misused the Samanthula *et al.*'s SBD protocol.

2 Preliminaries

In 1999, Paillier [2] proposed a cryptosystem (Table 1), in which the function $E(m)$ has the additively homomorphic property, i.e., $E(m_1)E(m_2) = E(m_1 + m_2)$.

[1]Department of Mathematics, Shanghai University, Shanghai, 200444, China
[2]Department of Mathematics, Shanghai Maritime University, Shanghai, 201306, China.
Email: liulh@shmtu.edu.cn

The Samanthula *et al.*'s SBD protocol [1] can be described as follows. It needs to invoke a sub-protocol (denoted by Encrypted-LSB) to iteratively compute the encryption of least significant bit from current $E(x)$, and invoke the sub-protocol of Secure Verification of Result (denoted by SVR).

Table 1: Paillier's encryption

Setup	Pick an RSA modulus $N = pq$. Set $\lambda = \text{lcm}(p-1, q-1)$. Select $g \in \mathbb{Z}^*_{N^2}$ such that the order of g modulo N^2 is divisible by N. Publish N, g and keep λ in secret.
Enc.	For $m \in \mathbb{Z}_N$, pick $r \in \mathbb{Z}_N$, compute the ciphertext $c = E(m) = g^m r^N \bmod N^2$.
Dec.	Recover the plaintext $m = D(c) = \left(\frac{c^\lambda - 1 \bmod N^2}{N}\right) / \left(\frac{g^\lambda - 1 \bmod N^2}{N}\right) \bmod N$

Algorithm 1: $SBD(E(x)) \longrightarrow \langle E(x_0), \cdots, E(x_{m-1})\rangle$

Require: Bob has Paillier encrypted value $E(x)$, where x is not known to both parties and $0 \le x < 2^m$. (The secret key sk is known only to Alice)

1: $l \leftarrow 2^{-1} \bmod N$
2: $T \leftarrow E(x)$
3: **for** $i = 0 \rightarrow m-1$ do
4: $\quad E(x_i) \leftarrow$ **Encrypted-LSB**(T, i)
5: $\quad Z \leftarrow T \times E(x_i)^{N-1} \bmod N^2$
$\quad$ {update T with the encrypted value of q_i}
6: $\quad T \leftarrow Z^l \bmod N^2$
7: **end for**
8: $\gamma \leftarrow$ **SVR**$(E(x), \langle E(x_0), \cdots, E(x_{m-1})\rangle)$
9: **if** $\gamma = 1$ **then** return
10: **else** go to Step 2
11: **end if**

Table 2: Encrypted-LSB$(T, i) \rightarrow E(x_i)$

Alice (has the secret key sk)	Bob (has T for the iteration i)
	Pick $r \in \mathbb{Z}_N$ to compute $Y = T \times E(r) \bmod N^2$.
Compute $y = D(Y)$.	$\xleftarrow{Y}$
If y is even then set $\alpha = E(0)$, else $\alpha = E(1)$.	
$\xrightarrow{\alpha}$	If r is even then set $E(x_i) = \alpha$, else $E(x_i) = E(1) \times \alpha^{N-1} \bmod N^2$. Output $E(x_i)$.

3 Review of the scheme

The scheme involves five entities: KGC (Key Generation Center), CP (Cloud Platform), DP (Data Provider), IS (Internal Server), and RU (Request User). The proposed system model can be depicted as below (Fig. 1).

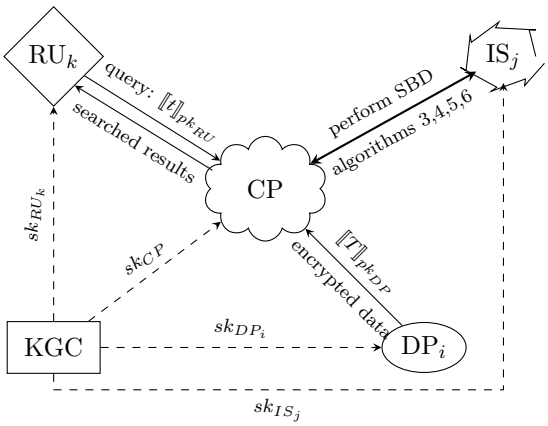

Figure 1: The proposed system model

KeyGen. For the security parameter k, KGC picks two k-bit primes p and q for Paillier cryptosystem to generate the private key λ and two partial private keys λ_1, λ_2. Securely send $sk_{CP} = \lambda_1$ to CP, and $sk_{IS} = \lambda_2$ to IS, respectively. It also initializes a keyword set $\mathcal{W}$ which contains μ keywords. Publish $(\mathcal{W}, \mu, N, g)$. Each DP generates the public key $pk_{DP} = (N, g, h_{DP})$ and the private key sk_{DP}. Each RU generates the public key $pk_{RU} = (N, g, h_{RU})$ and the private key sk_{RU}.

Store. Given DP's public key pk_{DP} and a document keyword set $\mathcal{W}_T \subseteq \mathcal{W}$, DP computes the searchable ciphertext $T \in \{0, \cdots, 2^\mu - 1\}$ whose binary representation is $(T_{\mu-1}, \cdots, T_0)$. Compute the searchable ciphertext $[\![T]\!]_{pk_{DP}}$ and send it to CP.

Trapdoor. Given RU's public key pk_{RU} and the keyword set $\mathcal{W}_t \subseteq \mathcal{W}$ of interest, RU computes the trapdoor $t \in \{0, \cdots, 2^\mu - 1\}$ whose binary representation is $(t_{\mu-1}, \cdots, t_0)$. Compute the encrypted trapdoor $[\![t]\!]_{pk_{RU}}$ and send it to CP.

Test. Given DP's public key pk_{DP}, RU's public key pk_{RU}, CP's secret key sk_{CP}, IS's secret key sk_{IS}, RU's encrypted trapdoor $[\![t]\!]_{pk_{RU}}$, and a searchable ciphertext $[\![T]\!]_{pk_{DP}}$, CP and IS perform the following steps (see Table 3 for Step-1). As for the other steps, we refer to the descriptions (Algorithm 4, 5, 6, §5.3, [3]).

Table 3: Step-1

Input:	$[\![t]\!]_{pk_{RU}}$, $[\![T]\!]_{pk_{DP}}$, $pk_{DP}, pk_{RU}, sk_{CP}, sk_{IS}$.
Output:	$[\![\neg T_i]\!]_{pk_{DP}}$ and $[\![t_i]\!]_{pk_{RU}}$ for $i \in \{0, \cdots, \mu - 1\}$. (where $\neg T_i = T_i \oplus 1$)
1)	CP computes $[\![\neg T]\!]_{pk_{DP}} = [\![2^\mu - 1]\!]_{pk_{DP}} \cdot ([\![T]\!]_{pk_{DP}})^{N-1}$.
2)	CP runs SBD protocol with IS: $\text{SBD}([\![\neg T]\!]_{pk_{DP}}) \leftarrow ([\![\neg T_{\mu-1}]\!]_{pk_{DP}}, \cdots, [\![\neg T_0]\!]_{pk_{DP}})$, $\text{SBD}([\![t]\!]_{pk_{RU}}) \leftarrow ([\![t_{\mu-1}]\!]_{pk_{RU}}, \cdots, [\![t_0]\!]_{pk_{RU}})$.

4 The flaws

4.1 The flawed system's setup

In the phase of **KeyGen**, KGC is responsible for generating the system's parameters $\mathcal{W}, \mu, N, g$, where N is an RSA modulus. So, the master key is $\lambda = \text{lcm}(p-1, q-1)$. Notice that each DP also adopts N, g as its own public parameters. So does each RU. Namely,

$$pk_{DP} = (N, g, h_{DP}), \quad pk_{RU} = (N, g, h_{RU}).$$

We find both h_{DP} and h_{RU} are not specified and invoked.

Since the factors p, q of N are only known to KGC, the DP's secret key sk_{DP} must be generated and issued by KGC. That means sk_{DP} should be of the form $\varphi\lambda$ where the random blinder φ is used to protect the master key λ. In view of there are lots of DPs and RUs, and each party's secret key has the same multiplier λ, we find a group of colluded DPs and RUs can recover the master key λ, which contradicts the basic requirement for the confidentiality of master key.

By the way, May [4] has proven that computing the RSA secret key is deterministic polynomial time equivalent to factoring. So, a system with a same RSA modulus for different users with different secret keys is insecure against collusion attack.

4.2 The scheme cannot work

In the Step-1 (Table 3), CP has the ciphertext $[\![\neg T]\!]_{pk_{DP}}$ and wants to obtain its secure bit-decomposition SBD($[\![\neg T]\!]_{pk_{DP}}$) with the help of IS. The SBD process needs to invoke the sub-protocol Encrypted-LSB to compute the encryption of least significant bit from current $[\![\neg T]\!]_{pk_{DP}}$. Concretely, in the sub-process CP acts as the role of Bob, and IS acts as the role of Alice (see Table 2). Therefore, IS has to use the secret key sk_{DP} to decrypt the ciphertext

$$Y = [\![\neg T]\!]_{pk_{DP}} \times E(r) \bmod N^2$$

in order to recover $y = \neg T + r$. But the DP's secret key sk_{DP} is not accessible to the IS. Thus, we find CP and IS cannot finish the sub-protocol.

Note that an adversary cannot derive an equivalent key of sk_{DP} from the partial secret keys sk_{CP}, sk_{IS} (both are not explicitly specified). Otherwise, the IS can directly decrypt $[\![T]\!]_{pk_{DP}}$ to obtain T, which represents the inclusion relationship between each keyword of W and a document D. As a result, the search query privacy would be completely evaporated.

5 Conclusion

We show that the Liu *et al.*'s searchable encryption scheme is flawed. We want to stress that the integration of secure bit-decomposition (SBD) with other primitives should be carefully handled, because SBD itself is fairly sophisticated.

References

[1] B. Samanthula, C. Hu, and W. Jiang, An efficient and probabilistic secure bit-decomposition, in Proc. 8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS'13, Hangzhou, China, May 2013, pp. 541C546.

[2] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in Proc. International Conference on Theory and Application of Cryptographic Techniques, EUROCRYPT'99, Prague, Czech Republic, May 1999, pp. 223–238.

[3] X. Liu and et al., Privacy-preserving multi-keyword searchable encryption for distributed systems, IEEE Trans. Parallel Distributed Syst., vol. 32, no. 3, pp. 561–574, 2021.

[4] A. May, Computing the RSA secret key is deterministic polynomial time equivalent to factoring, in Proc. Advances in Cryptology - CRYPTO'04, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 2004, pp. 213–219.