A note on “faster and efficient cloud-server-aided data de-duplication scheme with an authenticated key agreement for Industrial Internet-of-Things”

Zhengjun Cao[1], Lihua Liu[2]

Abstract. We show that the data de-duplication scheme [Internet of Things, 2021(14): 100376] is flawed. (1) There are some inconsistent notations and false equations, which should be corrected. (2) The scheme fails to keep user anonymity, not as claimed. (3) The scheme could fail to keep data confidentiality.

Keywords: Data confidentiality, Data duplication, Security and privacy, Cloud computing, User anonymity

1 Introduction

Recently, Olakanmi and Odeyemi [1] have presented a cloud-server-aided data de-duplication scheme, in which there are four entities: data owner (DO), key management center (KMC), cloud server (CS), and data user (DU). The KMC only generates the system public parameters. The CS is considered as a semi-trusted server. DO’s Industrial Internet of Things (IIoT) devices are connected to a gateway through where each device’s data is outsourced to the CS.

The considered scenario consists of company A, company B, and cloud server C. C uses the proposed scheme for de-duplication while B may use the data of A if and only if A authorized B. A and B are assumed to have an IIoT sub-network, respectively. Each has a gateway replacing the local key server. Each device uploads its tokenized-tags to C through the corresponding gateway. The gateway first generates and uploads the tokenized tag to C. C performs the pre-de-duplication. Then the gateway encrypts and signs the device’s data, computes the device-linkable-searching-trapdoor. It finally uploads the ciphertexts, signature, and the device-linkable-searching-trapdoor, and C performs integrity and consistency test. To use the uploaded data of A, B first searches for the corresponding data of A by generating and uploading authentication token to C. C searches for the data using the authentication token of B, and returns the corresponding ciphertext. The design objectives include confidentiality, anonymity, cross de-duplication, consistency, and data integrity. For conveniences, we now revisit the scheme [1] as follows (see Table 1).

Though the proposed scenario and scheme are interesting, we find the scheme is flawed because of some inconsistent notations and wrong equations. We also find it cannot keep the data owner anonymity and data confidentiality.

[1]Department of Mathematics, Shanghai University, Shanghai, 200444, China

[2]Department of Mathematics, Shanghai Maritime University, Shanghai, 201306, China.

Email: liulh@shmtu.edu.cn

Table 1: The Olakanmi-Odeyemi data de-duplication scheme

KMC selects groups G, G_1 of prime order q. $P \in G$ is a generator. $e : G \times G \mapsto G_1$ is a bilinear map. $\vartheta = e(P, P)$.
$H_y(\cdot)$ and $H(\cdot) : \{0,1\}^* \mapsto Z_q^*$ are key and non-key based hash functions, respectively.
$Enc(\cdot), Dec(\cdot)$ are symmetric key encryption, decryption algorithms, respectively.

Data owner's device (id_i)	Gateway (id_{gw})	Cloud server (CS)	Data user (DU)
$\xrightarrow[\text{[register]}]{id_i}$	Pick $s_i \in Z_q^*$, store $\{id_i, s_i\}$.		
$\xrightarrow{id_i,\ D_{i,t}}$	Pick $f_{i,t}, x_i \in Z_q^*$. $R_1 = x_iP, R_2 = s_iP, R_3 = x_if_{i,t}P$, $T_{sp} = e(H(id_{gw})P, H(id_i)P)^{x_if_{i,t}(s_i+x_i)}$, $\kappa_1 = e(H(id_{gw})P, H(id_i)P)^{f_{i,t}x_i(s_i+x_i)}$, $\cdot e((f_{i,t}+s_i)P, H(id_i)P)^{(s_i+x_i)}$, $\kappa_2 = H(T_{sp}), T_{cdt} = \kappa_2 T_{sp}$, $\psi = Enc_{\kappa_1}(D_{i,t}), \sigma = H_{\kappa_2}(\psi)$. $\delta_i = e(R_1, [H(id_i)+1]R_2)$, $\delta_{gw} = e(R_1, [H(id_{gw})+1]P)$, $T_1 = e(\psi P, R_1)$, $T_{tk} = e\,(R_1, \sigma +$ $[s_i(H(id_i)+1) + H(id_{gw}) + 1]P)$.	Compare the T_{tk} with other data's T_{tk}.	
	$\xrightarrow{T_{tk}, R_3}$	If no similarity, accept it.	
	Upload the 6-tuple data.	$\xleftarrow{\text{OK}}$	
	$\xrightarrow{\{\psi,\sigma,\delta_i,\delta_{gw},T_{sp},T_{cdt}\}}$	Compute the key $\kappa_2' = H(e(H(id_{gw})R_3, H(id_i)(R_1 + R_2)))$, For each T_{sp} stored, check $T_{cdt} = T_{sp}\kappa_2'$. If so, discard the data. Otherwise, check if $\sigma = H_{\kappa_2'}(\psi)$, and $T_{tk} = T_1\delta_i\delta_{gw}$. If so, store the data.	
	$\longleftarrow$	$\xleftarrow[\text{request for } D_{i,t}]{}$	Send the request for the data $D_{i,t}$.
	$\Longrightarrow$	$\xRightarrow[\text{secure channel}]{f_{i,t}}$	Compute the authentication token $Ak = e(H(id_{gw})R_1, H(id_i)(R_1 + R_2))^{f_{i,t}}$,
		Check if there exists $T_{spi} = Ak$. If so, the requested data is available.	$\xleftarrow{Ak}$ Compute the key $\kappa_1' =$
		$\xrightarrow{\psi}$	$Ak \cdot e(f_{i,t}P + R_2, H(id_i)(R_1 + R_2))$, $D_{i,t} = Dec_{\kappa_1'}(\psi)$.

2 Inconsistent notations and false equations

— To confirm the consistency of the tokenized-tag, the CS needs to check the equation $T_{tk} = T_1\delta_i\delta_{gw}$. But we find *it is a false equation*. In fact,

$$R_1 = x_iP,\ R_2 = s_iP,\ \psi = Enc_{\kappa_1}(D_{i,t}),\ \sigma = H_{\kappa_2}(\psi),\ T_1 = e(\psi P, R_1),$$
$$\delta_i = e(R_1, [H(id_i)+1]R_2),\ \delta_{gw} = e(R_1, [H(id_{gw})+1]P),$$
$$T_{tk} = e(R_1, \sigma + [s_i(H(id_i)+1) + H(id_{gw}) + 1]P).$$

We have

$$\begin{aligned}
T_1\delta_i\delta_{gw} &= e(\psi P, R_1)e(R_1, [H(id_i)+1]R_2)e(R_1, [H(id_{gw})+1]P) \\
&= e(\psi P + [H(id_i)+1]R_2 + [H(id_{gw})+1]P, R_1) \\
&= e(\psi P + [H(id_i)+1]s_iP + [H(id_{gw})+1]P, R_1) \\
&= e((\psi + H(id_i)s_i + s_i + H(id_{gw})+1)P, R_1) \\
&\neq e(\sigma + [s_i(H(id_i)+1) + H(id_{gw})+1]P, R_1)
\end{aligned}$$

Hence, the proof of correctness 2 (see page 6, Ref.[1]) is wrong, in which $\sigma \in Z_q^*, \sigma \notin G$, the bilinear map $e(\sigma, xP)$ is meaningless. To revise, it needs to specify that

$$T_1 = e(\sigma P, R_1), T_{tk} = e(R_1, \sigma P + [s_i(H(id_i)+1) + H(id_{gw})+1]P).$$

— *The data user cannot complete the related computations.* In fact, the user needs to compute the authentication token

$$Ak = e(H(id_{gw})R_1, H(id_i)(R_1+R_2))^{f_{i,t}} \tag{1}$$

where $f_{i,t}$ is securely sent by the gateway. But it does not specify the way to enable the user to get other parameters id_i, R_1, R_2. The original description of the scheme is not explicitly organized.

— *There is a mere repetition computation* (see page 5, Ref.[1]). Since

$$\begin{aligned}
&T_{sp} = e(H(id_{gw})P, H(id_i)P)^{x_i f_{i,t}(s_i+x_i)}, \\
&\kappa_1 = e(H(id_{gw})P, H(id_i)P)^{f_{i,t}x_i(s_i+x_i)} \cdot e((f_{i,t}+s_i)P, H(id_i)P)^{(s_i+x_i)},
\end{aligned}$$

we have

$$\kappa_1 = T_{sp} \cdot e((f_{i,t}+s_i)P, H(id_i)P)^{(s_i+x_i)} \tag{2}$$

So, it is unnecessary to specify the repeated factor

$$e(H(id_{gw})P, H(id_i)P)^{f_{i,t}x_i(s_i+x_i)}$$

in the formula of κ_1.

— *There are some never invoked parameters*

$$\vartheta = e(P,P),\ \phi_i = \vartheta^{s_i} = e(s_iP, P)$$

which are generated by the KMC and the gateway, respectively, in the setup phase. Clearly, it is irrational to set some system public parameters while never invoking them.

3 The loss of data owner anonymity

The data owner anonymity is a basic design objective for the scheme. Namely, data users must be able to perform a data search *without compromising the identity of the data owner.* But we find it fails to keep data owner anonymity.

In fact, the user needs to compute the authentication token Ak, which means the user knows either the gateway's identity id_{wg} (or its hash value $H(id_{wg})$) or the device's identity id_i (or its hash value $H(id_i)$). Though the scheme has not specified the data owner's identity id_{DO} at all, we find the gateway's identity id_{wg} (or its hash value $H(id_{wg})$) is equivalently viewed as the identifier for the data owner, because any data user interested in data $D_{i,t}$ (generated by the IIoT device i, at time t) should ask a target data owner gateway to securely send the corresponding secret exponent $f_{i,t}$ as well as other parameters id_i, R_1, R_2. If the user fails to identify who is a target data owner, he cannot discriminate IIoT sub-networks (see Fig.1, Ref.[1]), and cannot make a proper request.

4 The loss of data confidentiality

As we see, the data $D_{i,t}$ is encrypted as $\psi = Enc_{\kappa_1}(D_{i,t})$ using the symmetric key encryption $Enc_{\kappa_1}(\cdot)$ with the key $\kappa_1 \in G_1$. The group G_1 has a sophisticated structure [2]. The computation of the bilinear map e is always done over the extension field $\mathbb{F}_{p^k}$, where p is a prime and k is the embedding degree. So, κ_1 cannot be directly used as a key for the encryption algorithm. By default, the first 256-bit (or 512-bit) substring of binary representation of κ_1 could be taken as the true key. The randomness of the truncated substring could be worse, which results in the loss of data confidentiality. To overcome this flaw, it needs to use the hash value $H(\kappa_1)$ as the symmetric key, not κ_1 itself. By the way, the practical representation of an element in an extension field is often neglected by some researchers.

5 Conclusion

In this note, we show that the Olakanmi-Odeyemi data de-duplication scheme has some flaws. The findings could be helpful for the future works on designing such schemes.

References

[1] O. Olakanmi, K. Odeyemi: Faster and efficient cloud-server-aided data de-duplication scheme with an authenticated key agreement for Industrial Internet-of-Things. Internet of Things, 14: 100376 (2021)

[2] R. Balasubramanian, N. Koblitz: The improbability that an elliptic curve has subexponential discrete Log problem under the Menezes-Okamoto-Vanstone algorithm. Journal of Cryptology, 11(2): 141-145 (1998)