IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 27, 2018, accepted November 22, 2018, date of publication November 27, 2018,
date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2883581

Cryptanalysis of a Lightweight Certificateless Signature Scheme for IIOT Environments

BO ZHANG[1,2], TIANQING ZHU[3], CHENGYU HU[4], AND CHUAN ZHAO[1,2]

[1]School of Information Science and Engineering, University of Jinan, Jinan 250022, China
[2]Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China
[3]School of Software, University of Technology Sydney, Ultimo, NSW 2007, Australia
[4]School of Software, Shandong University, Jinan 250101, China

Corresponding author: Bo Zhang (zhangbosdu@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702218, Grant 61672262, and Grant 61602275, in part by the International Joint Training Program for Young and Middle-aged Scholar of Shandong Province, Technology Program of University of Jinan, under Grant XKY1709, in part by the Doctoral Program of University of Jinan under Grant 160100224, in part by the Shandong Province Higher Educational Science and Technology Program under Grant J18KA349, and in part by the Open Project of Co-Innovation Center for Information Supply and Assurance Technology, Anhui University, under Grant ADXXBZ201702.

ABSTRACT As an extremely significant cryptographic primitive, certificateless signature (CLS) schemes can provide message authentication with no use of traditional digital certificates. High efficiency and provable security without random oracle are challenges in designing a CLS scheme. Recently, Karati *et al.* proposed an efficient pairing-based CLS scheme with no use of map-to-point hash function and random oracle model to provide data authenticity in Industrial Internet of Things (IIoT) systems. The security proof was given under several hardness assumptions. However, we notice that both public key replacement attack and known message attack are existing in Karati *et al*.'s scheme. Any adversary without knowledge of signer's private key is capable of forging valid signatures. This leads to several serious consequences. For example, anybody can sign IIoT data on behalf of IIoT data owner without being detected.

INDEX TERMS Public key replacement attack, known message attack, digital signature, certificateless.

I. INTRODUCTION

Certificateless signature (CLS) scheme is an extremely significant cryptographic primitive to provide message authentication. As a variant of identity-based signature (IBS), CLS scheme enables every user to generate a secret key for himself independently, in addition to the partial private key generated by the key generation center (KGC) from user's unique identifier information. CLS scheme successfully eliminates the problem of key escrow in IBS. More importantly, there is no need to certify the corresponding public key, so the public key management in CLS scheme is quite efficient. Due to above advantages, CLS scheme has received considerable attentions and become a hot topic in public key cryptography. However, many research works in the literature have failed to provide provable security in the standard model while achieving low computational cost at the same time.

Recently, Karati *et al.* [1] proposed a new pairing-based CLS scheme (Karati-Islam-Karuppiah scheme) to provide data authenticity in Industrial Internet of Things (IIOT) systems. The new scheme is quite efficient because no map-to-point (MTP) hash function is used in their construction. In order to obtain a convincing security for the new scheme, Goyal [2] and Karati and Biswas [3] proposed formal security proofs in the standard model under several hardness assumptions.

A. OUR CONTRIBUTIONS

Even though Karati-Islam-Karuppiah scheme is efficient and their formally security proofs are provided in the standard model,the proposed scheme is insecure under the *public key replacement attack* and *the known message attack* as shown in this paper. The public key replacement attack means that a forger can generate forged signatures on any messages without the private key of the victim. The known message attack means that a forger can generate a combined message signature pair by giving two valid signatures from a same victim. We present both attack algorithms and show that Karati-Islam-Karuppiah scheme for IIOT environments is not secure. Besides, we analyze the mistakes in the security proof of Karati-Islam-Karuppiah scheme.

B. RELATED WORK

The first concrete CLS scheme is proposed in [4] to eliminates the key-escrow problem for IBS and the formal definition of strict security for CLS schemes is considered in [5]–[7]. Since then, there have been several works on this subject. CLS may be combined with other special signatures to obtain new type of the certificateless signatures. Various schemes like certificateless signcryption [8]–[12], certificateless aggregate signature [13]–[17], certificateless ring signature [18], [19], certificateless threshold signature [20]–[22] etc. have been proposed.

Yum and Lee *et al*. [23] proposed a method to transfer any IBS scheme to a CLS scheme, but their constructions are vulnerable against public key replacement attack [6]. In 2006, [24] and [25] also showed the schemes in [26] and [27] are insecure against this kind of attacks. The MTP hash function and bilinear pairing are assumed to be the high computational cryptographic operations. Therefore, as an enhancement, Du and Wen [28] and He *et al*. [29] designed CLS schemes without MTP hash function or bilinear pairing, respectively. These constructions were good attempts to make the CLS scheme more efficient. Unfortunately, the concrete schemes in [28] and [29] were found insecure under the public key replacement attack [30] and the attack launched by a malicious KGC [31], [32], respectively.

In recent years, there were lots of CLS schemes without bilinear pairings [33]–[39]. However, almost all of them are only provably secure under the random oracle model (ROM) [40]. Even though ROM leads to efficient construction, it also has faced lots of criticism due to insecure guarantees in some scenarios. When ROM is implemented with a concrete hash function, the aforementioned schemes may be insecure [41]. The first concrete CLS scheme with no use of ROM was proposed by Liu *et al*. [42], but was proved insecure under the attack launched by a malicious KGC by Huang and Wong [43] and Xiong *et al*. [44], respectively. Then some modified schemes were put forward in [45]–[49]. But, the existing CLS schemes with no use of ROM are not secure and there are too many bilinear pairings in their concrete schemes. Therefore, constructing an efficient concrete CLS scheme provably secure against attacks with no use of ROM is still an unresolved problem.

C. PAPER ORGANIZATION

In Section *II*, we introduce preliminaries including complexity assumptions, bilinear pairing, formal definition of CLS scheme and Karati-Islam-Karuppiah scheme in detail. The concrete attacks on Karati-Islam-Karuppiah scheme, the performance analysis and the experiment results are presented in Section *III*. In Section *IV*, the security proving process of Karati-Islam-Karuppiah scheme is present and the mistakes in the process are pointed out. Section *V* provides a conclusion.

TABLE 1. List of the notations used.

Symbols	Meaning
p	Sufficient large prime number
G_1, G_2	Cyclic groups of same order p
g_1	Generator of G_1
$e(\cdot,\cdot)$	The bilinear pairing $e: G_1 \times G_1 \rightarrow G_2$
$H(\cdot)$	Cryptographic hash function:$H: \{0,1\}^* \rightarrow \mathbb{Z}_p^*$
MSK	The private key of the KGC
Y_{KGC}	The public key of the KGC
ID_i	i-th user identity
D_i	The partial private key of user i
x_i	The secret value selected by the user i
SK_i	The private key of user i
Y_i	The public key of user i
m	The message in $\mathbb{Z}_p^*$
σ	The signature of message

II. PRELIMINARIES

We briefly review related concepts in Karati-Islam-Karuppiah scheme in this section, including complexity assumptions, bilinear pairing, formal definition of CLS scheme. We also provide a complete description of the details of Karati-Islam-Karuppiah scheme to ensure this paper's integrity. As shown in Table 1, we use the same symbols as in Karati-Islam-Karuppiah scheme.

A. SECURITY PROBLEMS AND COMPLEXITY ASSUMPTIONS

• q-Bilinear Strong Diffie-Hellman (q-BSDH) Problem. Let G_1 be a prime p ordered cyclic group with a generator g. Given $(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)})$, computing a pair $(e(g,g)^{\frac{1}{\theta+r}}, r)$.

Definition 1: q-BSDH assumption [2]). Let G_1 be a prime p ordered cyclic group with a generator g. Given $(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)})$, the successful advantage is presented as $|Pr[\mathscr{A}(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)}) = (r, e(g,g)^{\frac{1}{\theta+r}})|$. If there exists no such $\mathscr{A}$ with non-negligible advantage ϵ within time t in solving the q-BSDH problem, then the q-BSDH assumption holds.

• q-Extended Bilinear Strong Diffie-Hellman (q-EBSDH) Problem. Let G_1 be a prime p ordered cyclic group, g be a generator of G_1. Given a vector $(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)})$, computing a pair $(r, e(g,g)^{\frac{\theta}{\theta+r}})$ where $r \in \mathbb{Z}_p^*$.

Definition 2: q-EBSDH assumption [3]). Let G_1 be a prime p ordered cyclic group with a generator g. Given $(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)})$, the successful advantage is presented as $|Pr[\mathscr{A}(g, g^{\theta}, g^{(\theta^2)}, \cdots, g^{(\theta^q)}) = (r, e(g,g)^{\frac{\theta}{\theta+r}})|$. If there exists no such $\mathscr{A}$ with non-negligible advantage ϵ within time t in solving the q-EBSDH problem, then the q-EBSDH assumption holds.

B. BILINEAR PAIRING

Let (G_1, G_2) be a prime p ordered cyclic group pair. Also, g, h be two generators of G_1. Then $e: G_1 \times G_1 \rightarrow G_2$ with computability, bilinearity and non-degeneracy is a bilinear pairing:

1) **Computability**: e can be calculated efficiently.
2) **Bilinearity**: $e(g^x, h^y) = e(g, h)^{xy}$ where $x, y \in_R \mathbb{Z}_p^*$.
3) **Non-degeneracy**: $e(g^x, h^y) \neq 1_{G_2}$.

C. FORMAL DEFINITION OF CLS

The formal structure of a CLS scheme considers six different algorithms as mentioned below:

- **Setup**(k): Generates long time master key MSK and the public parameters $params$.
- **Set-PPK**(MSK, ID_i): Returns D_i to user i as the partial private key (PPK). D_i can be verified at anytime.
- **Set-Secret-Value**(ID_i): Sends a secret value x_i.
- **Set-Public-Key**(x_i, ID_i): Generates its full public key Y_i.
- **CLS-Sign**(SK_s, m): Transmits signature σ to the verifier where SK_s is the signer's private key.
- **CLS-Verify**(ID_s, PK_s, m, σ): If the signature σ is valid, then outputs VALID, otherwise, outputs INVALID. Here PK_s is the signer's public key.

D. SECURITY MODELS OF CLS

The security models of CLS can be described via the following games. A challenger $\mathscr{C}$ played these games with the forgers $\mathscr{A}_I$ and $\mathscr{A}_{II}$, respectively.

1) Type-*I* Model:
   - **Setup**: Challenger $\mathscr{C}$ generates MSK and $params$. It keeps MSK secret and sends $params$ to forger $\mathscr{A}_I$.
   - **Queries**: Forger $\mathscr{A}_I$ asks the following queries adaptively.
     - **Set-Secret-Value** (ID_i): $\mathscr{A}_I$ receives the secret value x_i of ID_i from $\mathscr{C}$.
     - **Set-PPK** (ID_i): $\mathscr{A}_I$ gains knowledge of D_i of ID_i.
     - **Set-Public-Key**(ID_i): $\mathscr{A}_I$ collects Y_i of ID_i from $\mathscr{C}$.
     - **Replace-Public-Key** (ID_i, Y_i): $\mathscr{A}_I$ replaces Y_i with a newly chosen Y_i'.
     - **Sign**(ID_i, m): $\mathscr{A}_I$ gains a valid signature σ for a chosen (ID_i, m).
   - **Output**: $\mathscr{A}_I$ produces a tuple $(ID', PK_{ID'}, m', \sigma)$. $\mathscr{A}_I$ wins the game if the tuple is not generated by Sign(ID', m') query and the output of CLS-Verify($ID', PK_{ID'}, m', \sigma$) is VALID and $\mathscr{A}_I$ has made no Set-PPK(ID') query.
     Note: $PK_{ID'}$ may have been changed by $\mathscr{A}_I$.

   Definition 3: A CLS scheme is Type-I secure against polynomially bounded forgers $\mathscr{A}_I$ if the success probabilities of $\mathscr{A}_I$ is negligible.
2) Type-*II* Model:
   - **Setup**: Challenger $\mathscr{C}$ generates MSK and $params$. Then, it sends MSK and $params$ to forger $\mathscr{A}_{II}$.
   - **Queries**: $\mathscr{A}_{II}$ adaptively asks one of the queries below.
     - **Set-Secret-Value** (ID_i): $\mathscr{A}_{II}$ receives the secret value x_i of ID_i from $\mathscr{C}$.
     - **Set-PPK** (ID_i): $\mathscr{A}_{II}$ gains knowledge of D_i of ID_i.
     - **Set-Public-Key**(ID_i): $\mathscr{A}_{II}$ collects Y_i of ID_i from $\mathscr{C}$.
     - **Sign**(ID_i, m): $\mathscr{A}_{II}$ gains a valid signature σ for a chosen (ID_i, m).
   - **Output**: $\mathscr{A}_{II}$ produces a tuple $(ID', PK_{ID'}, m', \sigma)$. $\mathscr{A}_{II}$ wins the game if the tuple is not generated by the Sign(ID', m') query and the output of CLS-Verify($ID', PK_{ID'}, m', \sigma'$) is VALID and $\mathscr{A}_I$ has made no Set-Secret-Value(ID') query.

   Definition 4: A CLS scheme is Type-II secure against polynomially bounded forgers $\mathscr{A}_{II}$ if the success probabilities of $\mathscr{A}_{II}$ is negligible.

E. KARATI-ISLAM-KARUPPIAH SCHEME

Karati-Islam-Karuppiah Scheme is a concrete CLS scheme. As shown below, it includes six algorithms.

- **Setup**(k): KGC chooses a prime p with k bits length, p ordered cyclic group pair (G_1, G_2) where g_1 is a generator of G_1. KGC also chooses its private key $y \in_R \mathbb{Z}_p^*$, a secure hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ and an efficient $e : G_1 \times G_1 \rightarrow G_2$. After that, KGC computes $g_2 = e(g_1, g_1)^y$ and the public key $Y_{KGC} = (g_1)^y$. Finally, KGC keeps $MSK = (y)$ safely, and publishes public parameter as $params =< G_1, G_2, p, e, g_1, g_2, Y_{KGC}, H >$.
- **Set-PPK**(MSK, ID_i): On receiving the parameter $params$ and the identity ID_i of user i, KGC with the master key y computes $h_i = H(ID_i)$ and $y_i = (g_1)^{\frac{y \cdot h_i}{h_i + r_i + y}}$ where $r_i \in_R \mathbb{Z}_p^*$. Then, KGC sends $D_i = (y_i, R_i = g_1^{r_i})$ to user i. After receiving D_i from the KGC securely, if the equation
$$e(g_1, Y_{KGC}^{h_i}) \stackrel{?}{=} e(y_i, (g^{h_i} \cdot R_i \cdot Y_{KGC}))$$
holds, D_i is genuine.
- **Set-Secret-Value**(ID_i): Given the public parameter $params$, user i chooses two random numbers (x_i, c_i) and sets its secret value $SK_i = (c_i, x_i, R_i)$.
- **Set-Public-Key**(x_i, ID_i): On receiving the parameter $params$ and x_i, user i sets the public key $Y_i = (Y_{i1} = (y_i)^{\frac{1}{x_i}}, Y_{i2} = g_2^{c_i})$.
- **CLS-Sign**(SK_s, m): On receiving the public parameter $params$ and a message $m \in \mathbb{Z}_p^*$, the signer with the private key $SK_s = (x_s, c_s, R_s)$ computes $h_s = H(ID_s)$, $\sigma_1 = g_2^t$ and $\sigma_2 = (g_1^{h_s} \cdot R_s \cdot Y_{KGC})^{(\frac{c_s}{m} - t)x_s}$ where $t \in_R \mathbb{Z}_p^*$. Finally, it outputs the signature $\sigma = (\sigma_1, \sigma_2)$ of message m.
- **CLS-Verify**(ID_s, Y_s, σ, m): Given the public parameter $params$, signer's identity ID_s with public key Y_s and (m, σ), this algorithm returns VALID, if $(\frac{Y_{s2}^{\frac{1}{m}}}{\sigma_1})^{h_s} \stackrel{?}{=} e(Y_{s1}, \sigma_2)$ holds, where $h_s = H(ID_s)$; otherwise, σ is considered as false signature and thus, it returns INVALID.

III. CRYPTANALYSIS OF KARATI-ISLAM-KARUPPIAH SCHEME

As shown in the above section, in order to design a secure concrete scheme, two types of attackers must be considered. $\mathscr{A}_I$ models an adversary who cannot obtain the long term master key of KGC, but can replace the public keys. $\mathscr{A}_{II}$ models a malicious KGC, who knows the long term master key, but cannot launch an attack through replacing users' public keys. A CLS scheme is required to satisfy existential unforgeability against both adversaries. But as shown below, we claim that Karati-Islam-Karuppiah scheme is insecure against either of the adversaries.

A. PUBLIC KEY REPLACEMENT ATTACK ON KARATI-ISLAM-KARUPPIAH SCHEME

First, we show $\mathscr{A}_I$ can replace A's public key by a particular value and generate user A's signature as follows.

Attack Algorithm 1 (AA1):

1) Choose $c_A \in \mathbb{Z}_p^*$ at random.
2) Compute $h_A = H(ID_A)$.
3) Replaces A's public key with the value $Y_A = (Y_{A1} = g_1^{h_A}, Y_{A2} = g_2^{c_A})$.
4) Choose $t \in \mathbb{Z}_p^*$ at random.
5) Compute $\sigma_1 = g_2^t$.
6) Compute $\sigma_2 = Y_{KGC}^{(\frac{c_A}{m}-t)}$.

Finally, $\mathscr{A}_I$ outputs the pair $\sigma = (\sigma_1, \sigma_2)$ as forged signatures.

Given the tuple $(m, ID_A, \sigma = (\sigma_1, \sigma_2), Y_A)$, the verification algorithm of CLS scheme will work as follows:

1) Compute $h_A = H(ID_A)$.
2) Check whether $(\frac{Y_{A2}^{\frac{1}{m}}}{\sigma_1})^{h_A} = e(Y_{A1}, \sigma_2)$, If it holds, the tuple is accepted as a valid signature.

Since $Y_A = (Y_{A1} = g_1^{h_A}, Y_{A2} = g_2^{c_A})$, $\sigma_1 = g_2^t$ and $\sigma_2 = Y_{KGC}^{(\frac{c_A}{m}-t)}$, one can derive that

$$\begin{aligned}(\frac{Y_{A2}^{\frac{1}{m}}}{\sigma_1})^{h_A} &= (\frac{g_2^{\frac{c_A}{m}}}{g_2^t})^{h_A} \\ &= g_2^{(\frac{c_A}{m}-t)h_A} \\ &= e(g_1, g_1)^{(\frac{c_A}{m}-t)h_A y} \\ &= e(g_1^{h_A}, g_1^{(\frac{c_A}{m}-t)y}) \\ &= e(g_1^{h_A}, Y_{KGC}^{(\frac{c_A}{m}-t)}) \\ &= e(Y_{A1}, \sigma_2) \end{aligned} \quad (1)$$

Tuple $(m, ID_A, \sigma = (\sigma_1, \sigma_2), Y_A)$ can always be accepted as a valid signature.

From (1), we can derive that all tuples $(Y_{A1}, Y_{A2}, \sigma_1, \sigma_2) \in \{(g_1, g_2^{c_A}, g_2^t, Y_{KGC}^{(\frac{c_A}{m}-t)h_A}), (Y_{KGC}, g_2^{c_A}, g_2^t, g_1^{(\frac{c_A}{m}-t)h_A}), (Y_{KGC}^{h_A}, g_2^{c_A}, g_2^t, g_1^{(\frac{c_A}{m}-t)})\}$ are valid signatures on message m since all of them can pass the verification algorithm. The attacks can always succeed in forging signatures but may easily be detected. This is because the replacements of public key $Y_{A1} \in \{g_1, Y_{KGC}, Y_{KGC}^{h_A}\}$ is easy to identify. $\{g_1, Y_{KGC}, Y_{KGC}^{h_A}\}$ are the typical elements in group G_1, where g_1,Y_{KGC} and $Y_{KGC}^{h_A}$ are the generator, the public key of KGC and the result of a deterministic algorithm, respectively.

Next, we give a more generic attack in which the replacement of the public key and the valid public key are indistinguishable. They have the same probability distribution.

$\mathscr{A}_I$ can forge user A's signature on message m by replacing A's public key to a random value.

Attack Algorithm 2 (AA2):

1) Choose $f_A, c_A \in \mathbb{Z}_p^*$ randomly.
2) Compute $h_A = H(ID_A)$.
3) Replace the public of U_A with the value $Y_A = (Y_{A1} = g_1^{h_A \cdot f_A}, Y_{A2} = g_2^{c_A})$.
4) Choose $t \in \mathbb{Z}_p^*$ at random.
5) Compute $\sigma_1 = g_2^t$.
6) Compute $\sigma_2 = Y_{KGC}^{(\frac{c_A}{m}-t)\cdot\frac{1}{f_A}}$.

Finally, $\mathscr{A}_I$ outputs the pair $\sigma = (\sigma_1, \sigma_2)$ as the forged signature.

Since $Y_A = (Y_{A1} = g_1^{h_A \cdot f_A}, Y_{A2} = g_2^{c_A})$, $\sigma_1 = g_2^t$ and $\sigma_2 = Y_{KGC}^{(\frac{c_A}{m}-t)\cdot\frac{1}{f_A}}$, one can derive that

$$\begin{aligned}(\frac{Y_{A2}^{\frac{1}{m}}}{\sigma_1})^{h_A} &= (\frac{g_2^{\frac{c_A}{m}}}{g_2^t})^{h_A} \\ &= g_2^{(\frac{c_A}{m}-t)h_A} \\ &= e(g_1, g_1)^{(\frac{c_A}{m}-t)h_A y \cdot f_A \cdot \frac{1}{f_A}} \\ &= e(g_1^{h_A \cdot f_A}, g_1^{(\frac{c_A}{m}-t)y \cdot \frac{1}{f_A}}) \\ &= e(g_1^{h_A \cdot f_A}, Y_{KGC}^{(\frac{c_A}{m}-t)\cdot\frac{1}{f_A}}) \\ &= e(Y_{A1}, \sigma_2) \end{aligned} \quad (2)$$

Therefore, tuple $(m, ID_A, \sigma = (\sigma_1, \sigma_2), Y_A)$ can always be accepted as a valid signature. More important, faked public key maintains the same distribution to the true public key with the help of random factor f_A. An adversary can generate signatures without the victim's private key on any messages. Thus, Karati-Islam-Karuppiah scheme is insecure if an adversary launches these attacks as described above.

B. KNOWN MESSAGE ATTACK ON KARATI-ISLAM-KARUPPIAH SCHEME

This section gives a detailed description about known message attack to Karati-Islam-Karuppiah scheme. In this attack, if $\mathscr{A}_I$ and $\mathscr{A}_{II}$ already obtain two valid message signatures $(m_1, \sigma_1, \sigma_2)$ and $(m_2, \sigma_1', \sigma_2')$, both of them can forge victim's A's signature on a combined message $m = \frac{m_1 \cdot m_2}{m_1 + m_2}$ as follows:

Attack Algorithm 3 (AA3):

1) Compute $\sigma_1'' = \sigma_1 \cdot \sigma_1'$.
2) Compute $\sigma_2'' = \sigma_2 \cdot \sigma_2'$.

Finally, (σ_1'', σ_2'') is outputted as the forged signature.

Since $h_A = H(ID_A)$, $\sigma_1 = g_2^{t_1}$, $\sigma_2 = (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{m_1}-t_1)x_A}$, $\sigma_1' = g_2^{t_2}$, $\sigma_2' = (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{m_2}-t_2)x_A}$, we

can deduce $\sigma_1'' = g_2^{t_1} \cdot g_2^{t_2} = g_2^{t_1+t_2}$ and

$$\begin{aligned}\sigma_2'' &= (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{m_1}-t_1)x_A} \\ &\quad \cdot (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{m_2}-t_2)x_A} \\ &= (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{m_1}-t_1)x_A+(\frac{c_A}{m_2}-t_2)x_A} \\ &= (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}-(t_1+t_2))x_A} \end{aligned} \tag{3}$$

One can deduce that

$$\begin{aligned}&(\frac{Y_{A2}}{\sigma_1''})^{\frac{1}{\frac{m_1 \cdot m_2}{m_1+m_2}}h_A} \\ &= (\frac{g_2^{\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}}}{g_2^{t_1+t_2}})^{h_A} \\ &= e(g_1, g_1)^{y(\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}-(t_1+t_2))h_A} \\ &= e(g_1, g_1)^{y \cdot h_A(\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}-(t_1+t_2)) \cdot x_A \cdot \frac{1}{x_A}} \\ &= e((g_1)^{\frac{y \cdot h_A}{(h_A+r_A+y)x_A}}, (g_1^{h_A+r_A+y})^{(\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}-(t_1+t_2)) \cdot x_A}) \\ &= ve((g_1)^{\frac{y \cdot h_A}{(h_A+r_A+y)x_A}}, (g_1^{h_A} \cdot R_A \cdot Y_{KGC})^{(\frac{c_A}{\frac{m_1 \cdot m_2}{m_1+m_2}}-(t_1+t_2))x_A}) \\ &= e(Y_{A1}, \sigma_2'') \end{aligned} \tag{4}$$

The tuple $(\frac{m_1 \cdot m_2}{m_1+m_2}, ID_A, \sigma = (\sigma_1'', \sigma_2''), Y_A)$ can always pass the verification algorithm. Thus, Karati-Islam-Karuppiah scheme is insecure against this attack as described above. In fact, in the definitions of security models, the adversaries can launch a more powerful adaptive chosen-message attack. They can ask the challenger to answer their Sign Queries for any well-constructed messages m_1, m_2 to make sure that the combination $\frac{m_1 \cdot m_2}{m_1+m_2}$ is a meaningful message.

C. PERFORMANCE ANALYSIS OF TWO ATTACKS

We compare the attack algorithms in terms of signature generation time with Karati-Islam-Karuppiah scheme (KIK for short)in this section.

We only focus on those time consuming cryptographic operations such as exponentiation operations, modular inversion operation and multiplication operations. In order to obtain the runtime of those operations and get an accurate comparison, we choose the same curve type and security parameter length as [1] and use PBC (Pairing-Based Crypto) library [50], which have been shown in Table 2. The hardware environment configuration is also shown in Table 2. We test every basic cryptographic operation 1000 times and the average runtime is shown in Table 3.

Based on the setting in Table 2 and 3. we estimate the total costs of KIK and the attack algorithms. The results have been shown in Table *IV*. Taking KIK as an example, to generate a valid signature, KIK has two element exponentiation in G_1, one element exponentiation in G_2, two element multiplication in G_1 and one element inversion. The total time costs of KIK is $10.73 \times 2 + 0.89 + 0.04 \times 2 + 0.015 \approx 22.445$ ms. According to this method, those three proposed attack algorithms require 23.255 ms, 23.27 ms and 0.049 ms, respectively.

Figure 1 shows the comparison between different attack algorithms and KIK. The figure shows that AA1 and AA2 need almost the same computational time as KIK in signature generation, while AA3 needs much less time. The results show that all of those attack algorithms can generate

TABLE 2. Basic information of implementation.

Ellipse Curve Type	Type A
Ellipse Curve	$y^2 = x^3 + x$
Security Level	$log_2 p = 512$
Platform	Personal Computer
CPU Series	Inter Core i5-4210H 2.9 GHz
RAM	8GB
Operate system	Windows 8
Crypto library	jPBC 2.0.0

TABLE 3. Runtime of the basic cryptographic operations (in millisecond).

Operations	Run time
Element exponentiation in G_1	$T_E^{G_1} \approx 10.73$ ms
Element exponentiation in G_2	$T_E^{G_2} \approx 0.89$ ms
Element multiplication in G_1	$T_M^{G_1} \approx 0.04$ ms
Element multiplication in G_2	$T_M^{G_2} \approx 0.009$ ms
Element inversion	$T_I \approx 0.015$ ms

TABLE 4. Computational costs of algorithms (in millisecond).

Algorithms	Signature Generation Time
KIK	$2T_E^{G_1} + T_E^{G_2} + 2T_M^{G_1} + T_I \approx 22.445$ ms
AA1	$2T_E^{G_1} + 2T_E^{G_2} + T_I \approx 23.255$ ms
AA2	$2T_E^{G_1} + 2T_E^{G_2} + 2T_I \approx 23.27$ ms
AA3	$T_M^{G_1} + T_M^{G_2} \approx 0.049$ ms

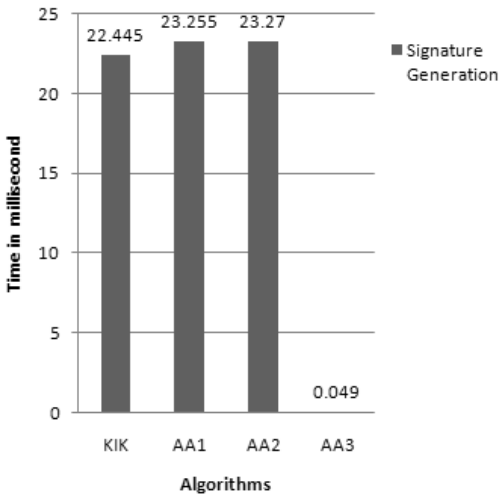

FIGURE 1. COST COMPARISONS BETWEEN OUR ATTACK ALGORITHMS AND KIK.

a forged signature efficiently. The results means that in the IIoT environment using KIK, the adversaries can succeed with breaking the authentication of IIoT data with limited computation costs.

IV. THE MISTAKES IN THE SECURITY PROOF OF KARATI-ISLAM-KARUPPIAH SCHEME

These attacks can be successfully proceeded due to the weaknesses in Theorem 1 and Theorem 2 in Karati-Islam-Karuppiah scheme. At first, we concisely summarize security proofs of Karati-Islam-Karuppiah Scheme so as to make the paper consistent.

A. SECURITY PROOFS OF KARATI-ISLAM-KARUPPIAH SCHEME

Theorem 1 (Type-I Security:): If there exists a forger $\mathscr{A}_I$ that breaks Type-I security of the Karati-Islam-Karuppiah Scheme, then there exists a solver $\mathscr{F}_I$ that breaches q-EBSDH assumption.

Proof: Given $\Psi =< G, g, g^x, g^{x^2}, \cdots, g^{x^p} >$, the challenger $\mathscr{C}$ wants to compute the solution $Z = e(g,g)^{\frac{x}{x+\theta}}$ of the q-EBSDH instance for some known $\theta \in \mathbb{Z}_p^*$, where G, g and q are the multiplicative group, its generator and the maximum number of queries, respectively. Suppose that there exists $\mathscr{A}_I$ who can break the Type-I security of the CLS. $\mathscr{C}$ can utilize $\mathscr{A}_I$ to solve the hard problem by playing the following interactive game with $\mathscr{A}_I$. For simplicity, the authors have considered that $A_i = g^{x^i}, \forall i \in [1, q]$ in Ψ.

Setup: $\mathscr{C}$ maintains lists $\mathscr{L}$ and $\mathscr{R}$ of tuples $(ID, Y_1, Y_2, x, c, r, h, y, R)$ and (ID, x, c, Y_1, Y_2), respectively. $\mathscr{L}$ and $\mathscr{R}$ are empty before the beginning of interactive game. $\mathscr{C}$ chooses $P(y)$ and $\Phi(y)$ of degree q as $P(y) = \sum_{i=0}^{q-1} \theta_i y^i$ and $\Phi(y) = \sum_{i=0}^{q-1} \beta_i y^i$, $\forall i \in [0, q-1]$, $(\theta_i, \beta_i) \in_R (\mathbb{Z}_p^*)^2$. It computes $g_1 = \prod_{i=0}^{q-1}(A_i)^{\theta_i} = g^{P(x)}$, $Y_{KGC} = \prod_{i=0}^{q-1}(A_{i+1})^{\theta_i} = g_1^x$ and $g_2 = e(g_1, Y_{KGC}) = e(g_1, g_1)^x$. $\mathscr{C}$ sends the system parameter $params = (G_1, G_2, q, e, g_1, g_2, Y_{KGC}, H)$ where $H : \{0,1\}^* \to \mathbb{Z}_p^*$ to $\mathscr{A}_I$.

Training phase: $\mathscr{C}$ responds $\mathscr{A}_I$'s queries as follows:

- **Create User(ID_i)**: $\mathscr{A}_I$ may submits a Create User query to $\mathscr{C}$. $\mathscr{C}$ selects $r_i = \phi(ID_i)$ and computes $R_i = g^{r_i}$. As the user's identity is public known, $P(y)$ can be represented as $\prod_{j=0}^{q-1}(y + r_j + h_j)$. Let, $P_i(y)$ be polynomial for ID_i and it is defined for $h_i = H(ID_i)$ and coefficients $(\mu_1, \cdots, \mu_1) \in_R (\mathbb{Z}_p^*)^q$ as

$$\begin{aligned} p_i(y) &= \frac{y \cdot P(y)}{y + r_i + h_i} + \mu_0 \\ &= \frac{y \cdot \prod_{j=0}^{q-1}(y + r_j + h_j)}{y + r_i + h_i} + \mu_0 \\ &= y \cdot \prod_{j=0, j\neq i}^{q-1}(y + r_j + h_j) + \mu_0 \\ &= \sum_{j=0}^{q-1} \mu_j y^j \end{aligned} \tag{5}$$

  It sets $y_i = \{\frac{\prod_{i=0}^{q-1}(A_i)^{\mu_i}}{g^{\mu 0}}\}^{h_i} = g^{\frac{x \cdot P(x) \cdot h_i}{x + r_i + h_i}} = g_1^{\frac{x \cdot h_i}{x + r_i + h_i}}$. Finally, it stores $(ID_i, \perp, \perp, \perp, \perp, r_i, h_i, y_i, R_i)$ in $\mathscr{L}$. It is noted that PPK verification condition $e(g_1, Y_{KGC})^{h_i} = e(y_i, g_1^{h_i \cdot r_i} \cdot Y_{KGC})$ holds.
- **Set-PPK(ID_i)**: $\mathscr{A}_I$ asks the query on ID_i and $\mathscr{C}$ returns $D_i = (y_i, R_i)$ if ID_i existed in $\mathscr{L}$. Otherwise, $\mathscr{C}$ calls Create User for $ID_i \neq ID^*$ and outputs $D_i = (y_i, R_i)$.
- **Set-Secret-Value(ID_i)**: $\mathscr{C}$ aborts the simulation if $ID_i = ID^*$. Otherwise, $\mathscr{C}$ searches for $ID_i (\neq ID^*)$ in $\mathscr{L}$. $\mathscr{C}$ returns (x_i, c_i, R_i) if it exists; otherwise, $\mathscr{C}$ chooses $(x_i', c_i' \in_R (\mathbb{Z}_p^*)^2$. Then, $\mathscr{C}$ updates only x_i as $x_i = x_i'$ and similarly for c_i with c_i' if an entry exists for $x_i = \perp$; otherwise, updates $x_i = x_i'$, $c_i = c_i'$ in $\mathscr{L}$ after calls Create User.
- **Set-Public-Value(ID_i)**: $\mathscr{A}_I$ asks the query on ID_i, if ID_i is found in $\mathscr{L}$, then $\mathscr{C}$ returns $Y_i = (Y_{i1}, Y_{i2})$. Otherwise, $\mathscr{C}$ calls Create User for $ID_i \neq ID^*$ and proceeds as
  - Checks for ID_i in $\mathscr{L}$ where $x_i \neq \perp$ and $c_i \neq \perp$. If such x_i and c_i do not exist, then it chooses $(x_i, c_i) \in_R (\mathbb{Z}_p^*)^2$. After that, it computes $Y_i = (Y_{i1}, Y_{i2})$, where $Y_{i1} = (yi)^{\frac{1}{x_i}}$ and $Y_{i2} = (g_2)^{c_i}$.
  - After that, it replaces the tuple $(\perp, \perp, \perp, \perp)$ by $(Y_{i1}, Y_{i2}, x_i, c_i)$, i.e., updates $(ID_i, Y_{i1}, Y_{i2}, x_i, c_i, r_i, h_i, y_i, R_i)$ in $\mathscr{L}$.

  Finally, $\mathscr{C}$ sends public key $Y_i = (Y_{i1}, Y_{i2})$ to $\mathscr{A}_I$.
- **Replace-Public-Key(ID_i, Y_i')**: Now, for invoked query $(ID_i, Y_i' = (Y_{i1}', Y_{i2}'))$, $\mathscr{C}$ sets $Y_{i1} = Y_{i1}', Y_{i2} = Y_{i2}'$, $x_i = x_i'$ and $c_i = c_i'$ which reflects in $\mathscr{L}$ if the corresponding tuple is present in $\mathscr{L}$. Finally, $\mathscr{C}$ inserts $(ID_i, x_i, c_i, Y_{i1}, Y_{i2})$ to list $\mathscr{L}$.
- **Sign(ID_i, m)**: $\mathscr{A}_I$ asks the query $q_s = (ID_i, m)$ and if it is not found in $\mathscr{L}$, $\mathscr{C}$ generate the signature as defined in original scheme. Otherwise, $\mathscr{C}$ considers list $\mathscr{R}$ and proceeds as follows:
  - Collects the secret key pair (x_i, c_i) from list $\mathscr{L}$.
  - Computes $\sigma_1 = (g_2)^k$ where $k \in_R \mathbb{Z}_p^*$.
  - Computes $\sigma_2 = \{(g_1)^{h_i} \cdot R_i \cdot Y_{KGC}\}^{(\frac{c_i}{m} - t)x_i}$ for $h_i = H(ID_i)$ and returns $\sigma = (\sigma_1, \sigma_2)$ to $\mathscr{A}_I$

Forgery phase: $\mathscr{A}_I$ stops asking queries and generates $\sigma' = (\sigma_1', \sigma_2')$ as a forged signature. Now, $\mathscr{C}$ aborts simulation if victim's identity $ID' \neq ID^*$. Otherwise, it considers a polynomial $\psi(y) = \sum_{i=0}^{q-2} \tau_i y^{i+1}$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-1}) \in (\mathbb{Z}_p^*)^{q-1}$ and expands the polynomial $P(y)$ as $P(y) = y^{-1} \cdot \psi(y) \cdot [(r_i + h_i) + y] + c$ where c is chosen selectively from $\mathbb{Z}_p^*$ so that the above equal holds successfully. Then, it finds $Y_{ID'1}$ from $\mathscr{L}$ where $Y_{ID'1} = (g_1)^{\frac{xh_i}{[(r_i+h_i)+x]x_i}}$. After that, $\mathscr{C}$ computes Υ as

$$\begin{aligned} \Upsilon &= [(Y_{ID'1})^{\frac{x_i}{h_i}} \cdot \prod_{i=1}^{q-1}(A_{i+1})^{-\tau_i}]^{\frac{1}{c}} \\ &= [g^{\frac{xP(x)}{[(r_i+h_i)+x]}} \cdot g^{-\Phi(x)}]^{\frac{1}{c}} \\ &= [g^{\Phi(x) + \frac{cx}{[(r_i+h_i)+x]}} \cdot g^{-\Phi(x)}]^{\frac{1}{c}} \\ &= [g^{\frac{cx}{[(r_i+h_i)+x]}}]^{\frac{1}{c}} \\ &= g^{\frac{x}{[(r_i+h_i)+x]}} \end{aligned} \tag{6}$$

$\mathscr{C}$ computes $Z = e(g, \Upsilon) = e(g, g)^{\frac{x}{r_i + h_i + x}}$. Now, if we view $\theta = r_i + h_i$, then $Z = e(g, g)^{\frac{x}{\theta + x}}$. Thus, it breaks q-EBSDH assumption.

Theorem 2 (Type-II Security):): If an adversary $\mathscr{A}_{II}$ who can break Type-II security of the Karati-Islam-Karuppiah Scheme exists, then there exists a solver $\mathscr{F}_{II}$ that breaches q-BSDH assumption.

Proof: Given $\Psi =< G, g, g^x, g^{x^2}, \cdots, g^{x^q} >$, the challenger $\mathscr{C}$ wants to compute the solution $Z = e(g, g)^{\frac{1}{x+\theta}}$ of the BSDH instance for some known $\theta \in \mathbb{Z}_p^*$ where G, g and q are the multiplicative group, its generator and the maximum number of queries, respectively. Suppose that there exists $\mathscr{A}_{II}$ who can break the Type-*II* security of the CLS. $\mathscr{C}$ can utilize $\mathscr{A}_{II}$ to solve the hard problem by playing the interactive game with $\mathscr{A}_{II}$ as following. For simplicity, the authors have considered that $A_i = g^{x^i}$, $\forall i \in [1, q]$ in Ψ.

Setup: $\mathscr{C}$ maintains $\mathscr{L}$ of tuple $(ID, Y_1, Y_2, x, c, r, h, y, R)$. $\mathscr{C}$ chooses $P(y)$ and $\Phi(y)$ of degree q as $P(y) = \sum_{i=0}^{q} \theta_i y^i$ and $\Phi(y) = \sum_{i=0}^{q} \beta_i y^i$, $\forall i \in [0, q]$, $(\theta_i, \beta_i) \in_R (\mathbb{Z}_p^*)^2$. It select $s \in_R \mathbb{Z}_p^*$, computes $g_1 = \prod_{i=0}^{q} (A_i)^{\theta_i} = g^{P(x)}$, $Y_{KGC} = g_1^s$ and $g_2 = e(g_1, g_1)^s$. $\mathscr{C}$ sends the system parameter $params = (G_1, G_2, q, e, g_1, g_2, Y_{KGC}, H)$ where $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$ to $\mathscr{A}_{II}$.

Training phase:$\mathscr{C}$ responds $\mathscr{A}_I$'s queries as follows:

- **Create User**(ID_i): $\mathscr{A}_{II}$ may submits a Create User query to $\mathscr{C}$. $\mathscr{C}$ selects $r_i = \phi(ID_i)$ and computes $R_i = g^{r_i}$. If $ID_i \neq ID^*$, then $\mathscr{C}$ computes $r_i = \Phi(ID_i)$, $R_i = (g_1)^{r_i}$ and $y_i = (g_1)^{\frac{s \cdot h_i}{h_i + r_i + s}}$. Otherwise, $\mathscr{C}$ sets $R_i = \prod_{i=1}^{q} (A_{i+1})^{\theta_i} = (g_1)^x$. As the user's identity is public known, so $P(y) = \prod_{j=0}^{q-1} (s + h_j + y)$. Let, $P_i(y)$ be polynomial for ID_i and

$$\begin{aligned} p_i(y) &= \frac{P(y)}{s + h_i + x} \\ &= \frac{\prod_{j=0}^{q-1} (s + h_j + y)}{s + h_i + x} \\ &= \prod_{j=0, j \neq i}^{q-1} (s + h_j + y) \\ &= \sum_{j=0}^{q-2} \mu_j y^j \end{aligned} \tag{7}$$

  for $(\mu_0, \mu_1, \cdots, \mu_{q-2}) \in_R (\mathbb{Z}_p^*)^{q-2}$. It sets $y_i = \{\prod_{i=0}^{q-2} (A_i)^{\mu_i}\}^{s \cdot h_i} = \{(g_1)^{\frac{1}{s + h_i + x}}\}^{s \cdot h_i}$. Finally, it stores $(ID_i, \perp, \perp, \perp, \perp, r_i, h_i, y_i, R_i)$ in $\mathscr{L}$. It is noted that PPK verification condition $e(g_1, Y_{KGC})^{h_i} = e(y_i, g_1^{h_i + x} \cdot Y_{KGC})$ holds.
- **Set-PPK**(ID_i): $\mathscr{A}_{II}$ asks the query on ID_i, $\mathscr{C}$ returns $D_i = (y_i, R_i)$ if ID_i is found in $\mathscr{L}$. Otherwise, $\mathscr{C}$ calls Create User and outputs $D_i = (y_i, R_i)$.
- **Set-Secret-Value**(ID_i): $\mathscr{A}_{II}$ asks the query on ID_i, $\mathscr{C}$ searches for $ID_i (\neq ID^*)$ in $\mathscr{L}$. $\mathscr{C}$ returns (x_i, c_i, R_i) if ID_i exists (other than $\perp$); otherwise, it chooses $(x_i', c_i') \in_R (\mathbb{Z}_p^*)^2$. Then, $\mathscr{C}$ updates only x_i as $x_i = x_i'$ and similarly for c_i with c_i' if an entry exists for $x_i = \perp$; otherwise, updates $x_i = x_i'$, $c_i = c_i'$ in $\mathscr{L}$ via calls Create-User.
- **Set-Public-Value**(ID_i): $\mathscr{A}_{II}$ asks the query on ID_i and $\mathscr{C}$ returns $Y_i = (Y_{i1}, Y_{i2})$ if ID_i is found in $\mathscr{L}$. Otherwise, $\mathscr{C}$ calls Create User and
  - Checks for ID_i where $x_i \neq \perp$ and $c_i \neq \perp$. If x_i and c_i do not exist, then it chooses $(x_i, c_i) \in_R (\mathbb{Z}_p^*)^2$. After that, it computes $Y_i = (Y_{i1}, Y_{i2})$, where $Y_{i1} = (y_i)^{\frac{1}{x_i}}$ and $Y_{i2} = (g_2)^{c_i}$.
  - After that, it replaces the tuple $(\perp, \perp, \perp, \perp)$ by $(Y_{i1}, Y_{i2}, x_i, c_i)$, i.e., updates $(ID_i, Y_{i1}, Y_{i2}, x_i, c_i, r_i, h_i, y_i, R_i)$ in $\mathscr{L}$.

  Finally, $\mathscr{C}$ sends public key $Y_i = (Y_{i1}, Y_{i2})$ to $\mathscr{A}_{II}$.
- **Sign**(ID_i, m): On receiving query $q_s = (ID_i, m)$, $\mathscr{C}$ proceeds as follows:
  - Collects the secret key pair (x_i, c_i) from list $\mathscr{L}$.
  - Computes $\sigma_1 = (g_2)^k$ where $k \in_R \mathbb{Z}_p^*$.
  - Computes $\sigma_2 = \{(g_1)^{h_i} \cdot R_i \cdot Y_{KGC}\}^{(\frac{c_i}{m} - t)x_i}$ for $h_i = H(ID_i)$ and returns $\sigma = (\sigma_1, \sigma_2)$ to $\mathscr{A}_{II}$.

Forgery phase: $\mathscr{A}_{II}$ stops asking queries and generate $\sigma' = (\sigma_1', \sigma_2')$ as a forged signature for a random chosen m' with ID' whose public key is $Y_{ID'} = (Y'_{ID'1}, Y'_{ID'2})$ where CLS-Verify($params, m', \sigma', ID', Y_{ID'}$) = VALID holds. Now, for $ID' \neq ID^*$, then $\mathscr{C}$ aborts simulation. Otherwise, it considers a polynomial $\psi(y) = \sum_{i=0}^{q-2} \tau_i y^i$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-2}) \in (\mathbb{Z}_p^*)^{q-2}$ and expands the polynomial $P(y)$ as $P(y) = \psi(y) \cdot [(s + h_i) + y] + \delta$ where δ is chosen selectively from $\mathbb{Z}_p^*$ so that the above equal holds successfully. Then, it finds $Y_{ID'1}$ from $\mathscr{L}$ where $Y_{ID'1} = (g_1)^{\frac{s \cdot h_{ID'}}{[(s + h_i) + x]} \cdot \frac{1}{x_{ID'}}}$. After that, $\mathscr{C}$ computes Υ as

$$\begin{aligned} \Upsilon &= [(Y_{ID'1})^{\frac{x_{ID'}}{s \cdot h_{ID'}}} \cdot \prod_{i=1}^{q-2} (A_{i+1})^{-\tau_i}]^{\frac{1}{\delta}} \\ &= [g^{\frac{P(x)}{[(s + h_i) + x]}} \cdot g^{-\Phi(x)}]^{\frac{1}{\delta}} \\ &= [g^{\Phi(x) + \frac{\delta}{[(s + h_i) + x]}} \cdot g^{-\Phi(x)}]^{\frac{1}{\delta}} \\ &= [g^{\frac{\delta}{[(s + h_i) + x]}}]^{\frac{1}{\delta}} \\ &= g^{\frac{1}{[(s + h_i) + x]}} \end{aligned} \tag{8}$$

$\mathscr{C}$ computes $Z = e(g, \Upsilon) = e(g, g)^{\frac{1}{s + h_i + x}}$. Now, if we view $\theta = s + h_i$, then $Z = e(g, g)^{\frac{1}{\theta + x}}$. Thus, it breaks q-BSDH assumption.

Next, the weaknesses in the security proofs of Karati-Islam-Karuppiah scheme will be pointed out.

B. THE WEAKNESSES IN THE PROOF OF THEOREM 1 (TYPE-I security)

1) WEAKNESS 1: THE PUBLIC KEY REPLACEMENT ATTACK WAS NOT CONSIDERED

In the Training Phase, $\mathscr{A}_I$ can issue Replace-Public-Key queries for any identities. For invoked query $(ID_i, Y_i' = (Y_{i1}', Y_{i2}'))$, the challenger $\mathscr{C}$ sets $Y_{i1} = Y_{i1}'$, $Y_{i2} = Y_{i2}'$,

$x_i = x_i'$ and $c_i = c_i'$ reflects in $\mathscr{L}$ if the corresponding tuple is present in $\mathscr{L}$. Finally, $\mathscr{C}$ inserts $(ID_i, x_i, c_i, Y_{i1}, Y_{i2})$ to list $\mathscr{L}$. Therefore, $\mathscr{A}_I$ can issue a Replace-Public-Key(ID^*) query.

In the Forgery Phase, $\mathscr{A}_1$ stops asking queries and generates $\sigma' = (\sigma_1', \sigma_2')$ with ID^*. $Y_{ID'1} = (g_1)^{\frac{xh_i}{[(r_i+h_i)+x]x_i}}$ are still used as the public key of ID^*. The authors clearly forgot that $\mathscr{A}_1$ may have already replaced this public key with another value. Therefore, in a sense, the public key replacement attack on the target identity was not considered in the proof of Theorem 1.

2) WEAKNESS 2: THE ABILITY OF $\mathscr{A}_1$ WAS NOT REDUCED TO COMPUTING A SOLUTION FOR THE Q-EBSDH PROBLEM

In the Forgery Phase, after $\mathscr{A}_I$ output $\sigma' = (\sigma_1', \sigma_2')$ with ID^*, $\mathscr{C}$ can continue the following process to solve the hard problem: It considers a polynomial $\psi(y) = \sum_{i=0}^{q-2} \tau_i y^{i+1}$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-1}) \in (\mathbb{Z}_p^*)^{q-1}$ and expands the polynomial $P(y)$ as $P(y) = y^{-1} \cdot \psi(y) \cdot [(r_i + h_i) + y] + c$ where c is chosen selectively from $\mathbb{Z}_p^*$. Then, it finds $Y_{ID'1}$ from $\mathscr{L}$ where $Y_{ID'1} = (g_1)^{\frac{xh_i}{[(r_i+h_i)+x]x_i}}$. After that, $\mathscr{C}$ computes $\Upsilon = [(Y_{ID'1})^{\frac{x_{ID'}}{s \cdot h_{ID'}}} \cdot \prod_{i=1}^{q-2}(A_{i+1})^{-\tau_i}]^{\frac{1}{\delta}}$ and views $Z = e(g, \Upsilon) = e(g, g)^{\frac{x}{r_i+h_i+x}}$ as a solution to the q-EBSDH problem.

In this process, $\mathscr{C}$ only used the public key $Y_{ID'} = (Y'_{ID'1}, Y'_{ID'2})$ associated with the target identity ID^*. The forged signature $\sigma' = (\sigma_1', \sigma_2')$ was not used at all. No matter what the signature is, $\mathscr{C}$ could continues the above process. Therefore, $\mathscr{C}$ solved the q-EBSDH problem without the help of $\mathscr{A}_I$. In another word, the capability of $\mathscr{A}_I$ was not reduced to solve the q-EBSDH problem in the proof of Theorem 1.

3) WEAKNESS 3: Q-EBSDH PROBLEM WAS NOT REALLY SOLVED IN THE PROOF

In the Training Phase, $\mathscr{C}$ reforms the polynomial $P(y)$ as $P(y) = \prod_{i=0}^{q-1}(y + r_i + h_i)$ to answer $\mathscr{A}_I$'s Create user oracle, where $r_i = \phi(ID_i)$ and $h_i = H(ID_i)$ for $i \in [0, q-1]$. In the Forgery Phase, $\mathscr{C}$ expands the same polynomial $P(y)$ as $P(y) = y^{-1} \cdot \psi(y) \cdot [(r_i + h_i) + y] + c$ where c is chosen selectively from $\mathbb{Z}_p^*$ and $\psi(y) = \sum_{i=0}^{q-2} \tau_i y^{i+1}$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-1}) \in (\mathbb{Z}_p^*)^{q-1}$. We have $y^{-1} \cdot \psi(y) \cdot [(r_i + h_i) + y] + c = \prod_{j=0}^{q-1}(y + r_j + h_j)$. We can derive that if and only if $c = 0$, the equation can be established.

Thus, at the end of game 1, the factor $\frac{1}{c}$ being used to compute Υ is meaningless. $\mathscr{C}$ cannot compute Υ via

$$\Upsilon = [(Y_{ID'1})^{\frac{x_i}{h_i}} \cdot \prod_{i=1}^{q-1}(A_{i+1})^{-\tau_i}]^{\frac{1}{c}}$$

q-EBSDH problem was not really solved in the proof.

C. THE WEAKNESSES IN THE PROOF OF THEOREM 2 (TYPE-II security)

1) WEAKNESS 1: THE ABILITY OF $\mathscr{A}_2$ WAS NOT REDUCED TO COMPUTING A SOLUTION FOR THE Q-BSDH PROBLEM

In the Forgery Phase, after $\mathscr{A}_{II}$ output a forged signature $\sigma' = (\sigma_1', \sigma_2')$ with the target identity ID^*. $\mathscr{C}$ could continues the process to solve the hard problem: it considers a polynomial $\psi(y) = \sum_{i=0}^{q-2} \tau_i y^i$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-2}) \in (\mathbb{Z}_p^*)^{q-2}$ and expands the polynomial $P(y)$ as $P(y) = \psi(y) \cdot [(s + h_i) + y] + \delta$ where δ is chosen selectively from $\mathbb{Z}_p^*$ so that the above equal holds successfully. Then, it finds $Y_{ID'1}$ from $\mathscr{L}$ where $Y_{ID'1} = (g_1)^{\frac{s \cdot h_{ID'}}{[(s+h_i)+x]} \cdot \frac{1}{x_{ID'}}}$. After that, $\mathscr{C}$ computes $\Upsilon = [(Y_{ID'1})^{\frac{x_{ID'}}{s \cdot h_{ID'}}} \cdot \prod_{i=1}^{q-2}(A_{i+1})^{-\tau_i}]^{\frac{1}{\delta}}$ as a solution.

In this process, $\mathscr{C}$ only used the public key $Y_{ID'} = (Y'_{ID'1}, Y'_{ID'2})$ associated with the target identity ID^*. $\sigma' = (\sigma_1', \sigma_2')$ generated by $\mathscr{A}_{II}$ was not used at all. No matter what the signature is, $\mathscr{C}$ could continues the above process. Therefore, in a sense, $\mathscr{C}$ solved the q-BSDH problem without the help of $\mathscr{A}_{II}$. In another word, the ability of $\mathscr{A}_{II}$ was not reduced to solving the hard problem in the proof of Theorem 2.

2) WEAKNESS 2: Q-BSDH PROBLEM WAS NOT REALLY SOLVED IN THE PROOF

The proof of Theorem 2 also has mistake in solving the hard problem. In the Training Phase, to answer the $\mathscr{A}_{II}$'s Create user oracle, $\mathscr{C}$ reforms the polynomial $P(y)$ as $P(y) = \prod_{j=0}^{q-1}(s + h_j + y)$, where $h_j = H(ID_j)$ for $j \in [0, q-1]$. In the Forgery Phase, $\mathscr{C}$ expands the same polynomial $P(y)$ as $P(y) = \psi(y) \cdot [(s + h_i) + y] + \delta$ where δ is some chosen integer and $\psi(y) = \sum_{i=1}^{q-2} \tau_i y^i$ for some $(\tau_1, \tau_2, \cdots, \tau_{q-1}) \in (\mathbb{Z}_p^*)^{q-2}$. We have $\prod_{j=0}^{q-1}(s + h_j + y) = \sum_{i=1}^{q-2} \tau_i y^i \cdot [(s + h_i) + y] + \delta$. We can derive that if and only if $c = 0$, the equation can be established.

Thus, at the end of game 2, the factor $\frac{1}{\delta}$ being used to compute Υ is meaningless. $\mathscr{C}$ cannot compute Υ via

$$\Upsilon = [(Y_{ID'1})^{\frac{x_i}{h_i}} \cdot \prod_{i=1}^{q-1}(A_{i+1})^{-\tau_i}]^{\frac{1}{\delta}}$$

q-BSDH problem was not really solved in the proof.

Therefore, due to the existence of these mistakes as mentioned above, Theorem 1 and Theorem 2 of Karati-Islam-Karuppiah scheme are invalid in terms of security.

V. CONCLUSIONS

Karati-Islam-Karuppiah scheme is quite efficient because no MTP hash function was used in the concrete construction. Karati *et al*. also gave formal security proofs without random oracles. However, as shown in this paper, Karati-Islam-Karuppiah scheme cannot defend the public key replacement attack and the known message attack. The adversary can generate forged signatures without knowing the victim's private key. In addition, the weaknesses in the proof process lead to a failure of provable security. In summary, Karati-Islam-Karuppiah scheme cannot be used in IIoT systems to provide authenticity since anybody can generate signatures on the data on behalf of the IIoT data owner without being detected. Furthermore, any communication protocol constructed based on this scheme is not secure and cannot resist various forms of attack, such as counterfeit attack, man-in-the-middle attack and so on [51].

REFERENCES

[1] R. Karati, S. H. Islam, and M. Karuppiah, "Provably secure and lightweight certificateless signature scheme for IIoT environments," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3701–3711, Aug. 2018, doi: 10.1109/TII.2018.2794991.

[2] V. Goyal, "Reducing trust in the PKG in identity based cryptosystems," in *Proc. CRYPTO*, CA, USA, 2007, pp. 430–447.

[3] A. Karati and G. P. Biswas, "Efficient and provably secure random oracle-free adaptive identity-based encryption with short-signature scheme," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 4060–4074, 2016.

[4] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. ASIACRYPT*, Taipei, China, 2003, pp. 452–474.

[5] Z. Zhang, D. Wong, J. Xu, and D. Feng, "Certificateless public-key signature: Security model and efficient construction," in *Proc. ACNS*, Singapore, 2006, pp. 293–308.

[6] B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng, "Key replacement attack against a generic construction of certificateless signature," in *Proc. ACISP*, Melbourne, VIC, Australia, 2006, pp. 235–246.

[7] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certficateless signature revisited," in *Proc. ACISP*, Townsville, QLD, Australia, 2007, pp. 308–322.

[8] M. Barbosa and P. Farshim, "Certificateless signcryption," in *Proc. ACM Symp. Inf., Comput. Commun. SecuRity (ASIACCS)*, 2008, pp. 369–372.

[9] D. Aranha, R. Castro, and J. Lopez, "Efficient certificateless signcryption," in *Proc. SBSEG*, 2008, pp. 257–258.

[10] C. Wu and Z. Chen, "A new efficient certificateless signcryption scheme," in *Proc. ISISE*, Beijing, China, 2008, pp. 661–664.

[11] W. Xie and Z. Zhang, "Efficient and provably secure certificateless signcryption from bilinear maps," in *Proc. IEEE Int. Conf. Wireless Commun., Netw. Inf. Secur.*, Jun. 2010, pp. 558–562.

[12] Z. Liu, Y. Hu, H. Ma, and X. Zhang, "Certificateless signcryption scheme in the standard model," *Inf. Sci.*, vol. 180, no. 3, pp. 452–464, 2010.

[13] L. Zhang, B. Qin, Q. Wu, and F. Zhang, "Efficient many-to-one authentication with certificateless aggregate signatures," *Comput. Netw.*, vol. 54, no. 14, pp. 2482–2491, 2010.

[14] Z. Gong, Y. Long, and H. Xiong, "Two certificateless aggregate signatures from bilinear maps," in *Proc. IEEE 8th ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw., Parallel/Distrib. Comput. (SNPD)*, vol. 3, Jul./Aug. 2007, pp. 188–193.

[15] L. Zhang and F. Zhang, "A new certificateless aggregate signature scheme," *Comput. Commun.*, vol. 32, no. 6, pp. 1079–1085, 2009.

[16] H. Xiong, Q. Wu, and Z. Chen, "An efficient provably secure certificateless aggregate signature applicable to mobile computation," *Control Cybern.*, vol. 41, no. 2, pp. 373–391, 2012.

[17] S.-J. Horng, S.-F. Tzeng, P.-H. Huang, X. Wang, T. Li, and M. K. Khan, "An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks," *Inf. Sci.*, vol. 317, pp. 48–66, Oct. 2015.

[18] L. Zhang, F. Zhang, and W. Wu, "A provably secure ring signature scheme in certificateless cryptography," in *Proc. 1st Int. Conf. Provable Secur. (ProvSec)*, 2007, pp. 103–121.

[19] S. Chang, D. S. Wong, Z. Zhang, and Y. Mu, "Certificateless threshold ring signature," *Inf. Sci.*, vol. 179, no. 20, pp. 3685–3696, 2009.

[20] L. Wang, Z. Cao, X. Li, and H. Qian, "Simulatability and security of certificateless threshold signatures," *Inf. Sci.*, vol. 177, no. 6, pp. 1382–1394, 2007.

[21] H. Yuan, F. Zhang, X. Huang, Y. Mu, W. Susilo, and L. Zhang, "Certificateless threshold signature scheme from bilinear maps," *Inf. Sci.*, vol. 180, no. 23, pp. 4714–4728, 2010.

[22] H. Xiong, F. Li, and Z. Qin, "Certificateless threshold signature secure in the standard model," *Inf. Sci.*, vol. 237, no. 10, pp. 73–81, 2013.

[23] D. H. Yum and P. J. Lee, "Generic construction of certificateless signature," in *Proc. ACISP*, 2004, pp. 200–211.

[24] M. C. Gorantla and A. Saxena, "An efficient certificateless signature scheme," in *Proc. CIS*, 2005, pp. 110–116.

[25] W.-S. Yap, S.-H. Heng, and B.-M. Goi, "An efficient certificateless signature scheme," in *Proc. EUC*, 2006, pp. 322–331.

[26] Z. F. Zhang and D. G. Feng, "Key replacement attack on a certificateless signature scheme," in *Proc. Cryptol. ePrint Arch.* Accessed: Nov. 30, 2006. [Online]. Available online http://eprint.iacr.org/2006/453

[27] X. Cao, K. G. Paterson, and W. Kou, "An attack on a certificateless signature scheme," in *Proc. Cryptol. ePrint Arch.* Accessed: Oct. 25, 2006. [Online]. Available online https://eprint.iacr.org/2006/367

[28] H. Du and Q. Wen, "Efficient and provably-secure certificateless short signature scheme from bilinear pairings," *Comput. Standards Inter.*, vol. 31, no. 2, pp. 390–394, 2009.

[29] D. He, J. Chen, and R. Zhang, "An efficient and provably-secure certificateless signature scheme without bilinear pairings," *Int. J. Commun. Syst.*, vol. 25, no. 11, pp. 1432–1442, 2012.

[30] K. Y. Choi, J. H. Park, and D. H. Lee, "A new provably secure certificateless short signature scheme," *Comput. Math. Appl.*, vol. 61, no. 7, pp. 1760–1768, 2011.

[31] M. Tian and L. Huang, "Cryptanalysis of a certificateless signature scheme without pairings," *Int. J. Commun. Syst.*, vol. 26, no. 11, pp. 1375–1381, 2013.

[32] J.-L. Tsai, N.-W. Lo, and T.-C. Wu, "Weaknesses and improvements of an efficient certificateless signature scheme without using bilinear pairings," *Int. J. Commun. Syst.*, vol. 27, no. 7, pp. 1083–1090, 2014.

[33] K.-H. Yeh, K.-Y. Tsai, R.-Z. Kuo, and T.-C. Wu, "Robust certificateless signature scheme without bilinear pairing," in *Proc. ICITCS*, 2013, pp. 1–4.

[34] D. He, Y. Chen, and J. Chen, "An efficient certificateless proxy signature scheme without pairing," *Math. Comput. Model.*, vol. 57, nos. 4–9, pp. 2510–2518, 2013.

[35] K.-H. Yeh, K.-Y. Tsai, and C.-Y. Fan, "An efficient certificateless signature scheme without bilinear pairings," *Multimedia Tools Appl.*, vol. 74, no. 16, pp. 6519–6530, 2015.

[36] P. Gong and P. Li, "Further improvement of a certificateless signature scheme without pairing," *Int. J. Commun. Syst.*, vol. 27, no. 10, pp. 2083–2091, 2015.

[37] L. Wang, K. Chen, Y. Long, H. Wang, and X. Mao, "A modified efficient certificateless signature scheme without bilinear pairings," in *Proc. Int. Conf. Intell. Netw. Collaborative Syst.*, 2015, pp. 82–85.

[38] K.-H. Yeh, C. Su, K.-K. R. Choo, and W. Chiu, "A novel certificateless signature scheme for smart objects in the Internet-of-Things," *Sensors*, vol. 17, no. 5, p. 1001, 2017.

[39] L. Wang, K. Chen, Y. Long, and H. Wang, "An efficient pairing-free certificateless signature scheme for resource-limited systems," *Sci. China Inf. Sci.*, vol. 60, no. 11, pp. 119102-1–119102-3, 2017.

[40] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proc. 1st ACM Conf. Comput. Commun. Secur.* New York, NY, USA: ACM Press, 1993, pp. 62–73.

[41] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, 2004.

[42] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model: Extended abstract," in *Proc. ACM Symp. Inf., Comput. Commun. Secur.*, 2007, pp. 273–283.

[43] Q. Huang and D. S. Wong, "Generic certificateless encryption in the standard model," in *Proc. IWSEC*, Nara, Japan, 2007, pp. 278–291.

[44] H. Xiong, Z. Qin, and F. Li, "An improved certificateless signature scheme secure in the standard model," *Fundam. Inform.*, vol. 88, nos. 1–4, pp. 193–206, 2008.

[45] Y. Yu, Y. Mu, G. Wang, Q. Xia, and B. Yang, "Improved certificateless signature scheme provably secure in the standard model," *IET Inf. Secur.*, vol. 6, no. 2, pp. 102–110, Jun. 2012.

[46] Y. Yuan and C. Wang, "Certificateless signature scheme with security enhanced in the standard model," *Inf. Process. Lett.*, vol. 114, no. 9, pp. 492–499, 2014.

[47] T.-T. Tsai, S.-S. Huang, and Y.-M. Tseng, "Secure certificateless signature with revocation in the standard model," *Math. Problems Eng.*, vol. 2014, Nov. 2014, Art. no. 728591.

[48] Y.-H. Hung, S.-S. Huang, Y.-M. Tseng, and T.-T. Tsai, "Certificateless signature with strong unforgeability in the standard model," *Informatica*, vol. 26, no. 4, pp. 663–684, 2015.

[49] L. Pang, Y. Hu, Y. Liu, K. Xu, and H. Li, "Efficient and secure certificateless signature scheme in the standard model," *Int. J. Commun. Syst.*, vol. 30, no. 5, p. e3041, 2017.

[50] B. Lynn. (2007). *PBC Library-the Pairing-Based Cryptography Library*. [Online]. Available: http://crypto.stanford.edu/pbc/

[51] A. Akhunzada *et al.*, "Man-At-The-End attacks: Analysis, taxonomy, human aspects, motivation and future directions," *J. Netw. Comput. Appl.* vol. 48, pp. 44–57, Feb. 2015.

BO ZHANG received the Ph.D. degree in computer application technology from Shandong University, China, in 2010. He is currently a Lecturer with the School of Information Science and Engineering, University of Jinan, China. He is focusing on certificateless signcryption, privacy-preserving data processing, and aggregation protocol in healthcare wireless sensor networks. His research interests include public key cryptography and information security.

TIANQING ZHU received the B.Eng. and M.Eng. degrees from Wuhan University, China, in 2000 and 2004, respectively, and the Ph.D. degree in computer science from Deakin University, Australia, in 2014. She was a Lecturer with the School of Information Technology, Deakin University, Australia. She is currently a Senior Lecturer with the School of Software, University of Technology Sydney, Australia. Her research interests include privacy preserving, data mining, and network security.

CHENGYU HU received the Ph.D. degree in computer application technology from Shandong University, China, in 2008. He is currently a Lecturer with the School of Software, Shandong University, China. He is focusing on the design of leakage-resilient cryptographic schemes and keyword search over encrypted data. His research interests include secure multi-party computing and zero-knowledge proof.

CHUAN ZHAO received the Ph.D. degree in computer science and technology from Shandong University, China, in 2016. He is currently a Lecturer with the School of Information Science and Engineering, University of Jinan, China. He is focusing on practical secure multi-party computation, privacy preserving technologies, and mobile security. His research interests include cryptography, information security, and privacy.