Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 2145829, 18 pages
https://doi.org/10.1155/2020/2145829

Research Article

A Lightweight Attribute-Based Security Scheme for Fog-Enabled Cyber Physical Systems

Aisha Kanwal Junejo and Nikos Komninos

Department of Computer Science, School of Mathematics, Computer Science, and Engineering, City University of London, EC1V 0HB London, UK

Correspondence should be addressed to Aisha Kanwal Junejo; aisha.junejo@city.ac.uk

Received 20 November 2019; Revised 9 July 2020; Accepted 24 July 2020; Published 21 September 2020

Academic Editor: Davide Mattera

In this paper, a lightweight attribute-based security scheme based on elliptic curve cryptography (ECC) is proposed for fog-enabled cyber physical systems (Fog-CPS). A novel aspect of the proposed scheme is that the communication between Fog-CPS entities is secure even when the certification authority (CA) is compromised. This is achieved by dividing the attributes into two sets, namely, secret and shared, and subsequently generating two key pairs, referred to as the partial and final key pairs, for each entity of the Fog-CPS system. Unlike existing attribute-based encryption (ABE) and identity-based encryption schemes, in the proposed scheme, each entity calculates the final public key of the communicating CPS devices without the need of generating and transmitting digital certificates. Moreover, the proposed security scheme considers an efficient and secure key pair update approach in which the calculation overhead is limited to one group element. To show the effectiveness of the proposed scheme, we have calculated and compared the memory and processing complexity with other bilinear and elliptic curve schemes. We have also implemented our scheme in a Raspberry Pi (3B+ model) for CPS simulations. The proposed scheme guarantees the confidentiality, integrity, privacy, and authenticity in Fog-CPS systems.

1. Introduction

Fog computing can improve the monitoring and management of next-generation cyber physical systems (CPS). A general fog-enabled cyber physical system (Fog-CPS) as shown in Figure 1 consists of three layers, namely, the *CPS device*, *fog*, and *cloud*. Fog-CPS systems are vulnerable to numerous security, privacy, and trust challenges. With regard to security, different attacks, namely, *interception*, *interruption*, *modification*, *fabrication*, *unauthorized authentication*, and *access*, can be carried out to disrupt the communication between Fog-CPS entities. The abovementioned challenges could be addressed by employing lightweight cryptographic techniques. However, the existing solutions have a number of limitations. *Firstly*, in public key encryption (PKE) schemes, the generation, verification, and distribution of certificates incur extra computation and communication overhead which is not desirable for resource-limited CPS devices. *Secondly*, in attribute-based encryption (ABE) schemes, there is a certification authority (CA) which generates the secret keys based on a set of attributes. However, the compromise of CA can endanger the secret keys and therefore the secrecy of encrypted messages. *Thirdly*, the existing ABE schemes which are based on bilinear pairing require large security parameters and therefore are not suitable for resource-constrained CPS devices.

Considering the limitations of existing solutions, in this paper, a lightweight security scheme is proposed. The proposed scheme guarantees that the communication between entities is secure in cases where the CA might be compromised.

1.1. Motivation. The motivation for designing such a solution comes from the identity-based encryption (IBE) and ABE schemes as it is believed that inherent attributes of CPS

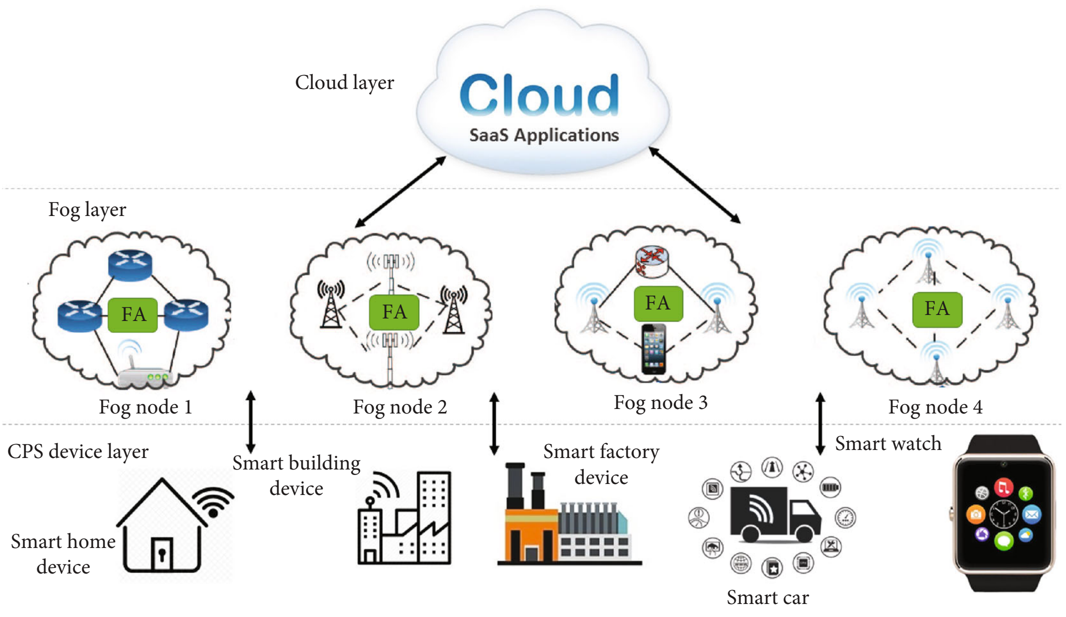

Figure 1: Fog-enabled cyber physical systems (Fog-CPS).

devices can be used for identification, authentication, and access control. From here onwards, the proposed scheme is called as Fog-CPS.

Moreover, in the proposed scheme, the CA (i.e., FA in this case) cannot decrypt the messages exchanged among CPS devices, thus maintaining their privacy. This is achieved by dividing the attribute set $\mathbb{A}$ into two subsets, a secret $\mathbb{A}_S$ and a shared $\mathbb{A}_K$ attribute subset. Precisely, each entity in a Fog-CPS system will have two key pairs, namely, a "partial key pair" and a "final key pair" generated from secret $\mathbb{A}_S$ and shared $\mathbb{A}_K$ attribute sets, respectively. The secret $\mathbb{A}_S$ attributes are only known to FA which registers the entities and generates the partial key pair. The registration of CPS entities with FA ensures that published public keys are authentic and do not require any further verification.

However, as the secret $\mathbb{A}_S$ attributes are only known to FA; the other collaborating CPS devices cannot verify them. To address this problem, a notion of "shared" $\mathbb{A}_K$ attributes is introduced. The shared $\mathbb{A}_K$ attributes are known to collaborating Fog-CPS entities which generate the final public and secret keys. The encryption and decryption would take place using final public and secret keys. Such an approach is advantageous for two reasons: (1) the secret $\mathbb{A}_S$ attributes are only shared with the FA and the leakage of partial secret keys would not risk the communication of collaborating entities and (2) the scheme is scalable because the final public keys are generated by the collaborating devices themselves without the aid of FA. Any device can generate the final public keys of other devices.

Furthermore, upon key update, the key regeneration process is lightweight, since the key generation process as found in Algorithms 1–3 is not repeated.

1.2. Contributions. The contributions of this paper are enumerated below:

(1) First, a lightweight attribute-based security scheme is proposed for Fog-CPS systems

(2) Second, the Fog-CPS scheme considers an efficient and secure key pair update approach in which the calculation overhead is limited to one group element

(3) Third, the proposed scheme is compared with other schemes based on bilinear pairing and elliptic curves by calculating its memory and processing overhead

(4) Fourth, the proposed scheme is implemented on a resource-limited Raspberry Pi (3B+ model) for CPS simulations.

The remaining parts of this paper are organized as follows: the related work is discussed in Section 2. The proposed scheme is presented in Section 3. Next, the experimental results are presented in Section 4. The conclusion and future work are discussed in Section 5.

2. Related Work

To secure the communication in CPS systems, the literature adopts a hybrid approach in which both symmetric (AES-based) and asymmetric (RSA-based and ABE) encryption techniques are used. AES (Advanced Encryption Standard) is used to encrypt the communication between sensor nodes and the gateway, whereas the asymmetric schemes are used to encrypt the communication between the gateway and the service provider.

```
1: **Input:** Security parameter $\lambda$, secret attribute set $\mathbb{A}_S$.
2: **Output:**$\mathrm{PK}_{\mathbb{A}_S}/\mathrm{SK}_{\mathbb{A}_S}$
    $\mathrm{PK}_{\mathbb{A}_S} = \{\mathbb{G}, P_i, H_1\}, \forall i = 0, 1, \cdots, t.$
    $\mathrm{SK}_{\mathbb{A}_S} = \{s\}.$
3: Choose elliptic curve group $\mathbb{G}$, where $P$ is a base point on the elliptic curve $E_p(a, b)$ defined over the finite field $Z_p$.
4: Choose a one-way collision resistance hash function, $H_1$ defined as:
        $H_1 : \{0, 1\}^* \rightarrow Z_p^*.$
5: Create a secret random number $r \in Z_p$.
6: Map $t < n$ attributes in secret set $\mathbb{A}_s$ to $Z_p$ using hash function $H_1$ and compute secret number $s$.
        $s_i = H_1(i) (\mathrm{mod\ p}), \quad \forall i \in \mathbb{A}_S, \quad (1)$
        $\acute{s} = \sum_{i=1}^{t} s_i,(2) s = r\acute{s}.(3)$
7: Next, compute public key components $P_i$ as:
        $P_i = s^i P, \forall i = 0, 1, .. \cdots, t. (4)$
```

Algorithm 1: Partial key pair generation.

```
1: **Input:** Public Key $\mathrm{PK}_{\mathbb{A}_S}$ and shared attribute set $\mathbb{A}_K$.
2: **Output:**$\mathrm{PK}_{\mathbb{A}_K} = \{U_i\}, \forall i = 0, 1, \cdots, t.$
3: Let $\mathbb{A}_K = a_1 a_2 \cdots .a_t$ be the device attribute string over shared attribute set.
4: Map $t < n$ attributes in shared set $\mathbb{A}_K$ to $Z_p$ using hash function $H_1$ and compute $k$.
        $k_i = H_1(i)(\mathrm{mod\ p}), \quad \forall i \in \mathbb{A}_K, \quad (5)$
        $k = \sum_{i=1}^{t} k_i. \quad (6)$
5: Next, compute final public key components $U_i$, $\forall i = 0, 1, \cdots, t$ as:
        $U_i = k^i P_i.(7)$
```

Algorithm 2: Final public key generation.

```
1: **Input:** Secret key $\mathrm{SK}_{\mathbb{A}_S}$ and shared attribute set $\mathbb{A}_K$.
2: **Output:**$\mathrm{SK}_{\mathbb{A}_K} = \{u_1\}.$
3: Compute secret number $\alpha$ as follows:
        $\alpha = sk,(8)$
   where $s$ is the secret key component from $\mathrm{SK}_{\mathbb{A}_S}$ and $k$ is computed similar to Equation (6) in Algorithm 2.
4: Next, compute Equation (9), $f(\alpha, \mathbb{A}_K)$ which is a $t$-degree at most polynomial in $Z_p[x]$.
        $f(\alpha, \mathbb{A}_K) = \prod_{i=1}^{t} (\alpha + H_1(i))^{1-a_i}.(9)$
5: Pick two random numbers $r_u, t_u \in Z_P$. Compute $s_u$ such that the following condition holds.
        $1/f(\alpha, \mathbb{A}_K) = s_u - r_u t_u \ (\mathrm{mod}\ p),$
        $s_u = (1/f(\alpha, \mathbb{A}_K)) + r_u t_u. \quad (10)$
6: Next, compute secret key component $u_1$
        $u_1 = s_u - r_u t_u \ (\mathrm{mod\ p}).(11)$
```

Algorithm 3: Final secret key generation.

Recently, Mahmood et al. [1] propose an authentication scheme for a smart grid. The scheme has a few limitations: (1) the scalability in case of large-scale CPS systems with hundreds of nodes and (2) the inconsideration of access control mechanism. In a smart grid system, there are several heterogeneous devices at different layers, so including access control mechanisms is crucial. Kocabas et al. [2] present a medical cloud-assisted CPS architecture consisting of acquisition, preprocessing, cloud, and action layers. The study proposes an AES (Advanced Encryption Standard) symmetric key encryption scheme for communication between the acquisition and preprocessing layers. The main disadvantage of [2] is the key management of symmetric keys in such a complex environment with hundreds of CPS devices. Shuo et al. [3] propose a distributed authentication framework for the multidomain M2M environment. The proposed framework employs a hybrid encryption scheme involving IBE and AES symmetric encryption.

Sravani et al. [4] present a signature-based authenticated key establishment scheme for IoT applications. Hu et al. [5]

Table 1: Notation table.

Notation	Description
λ	Security parameter
p	A sufficiently large prime number
$E_p(a, b)$	An elliptic curve $y^2 = x^3 + ax + b(\text{mod } p)$ defined over the finite field Z_p ; $Z_p = \{0, 1, \cdots, p-1\}$
P	A base point in $E_p(a, b)$
xP	Scalar multiplication, $P \epsilon E_p(a, b)$
$P + Q$	Elliptic curve point addition
$\mathbb{G}$	Elliptic curve group generated by P
$H_1()$	One-way collision resistance hash functions
$\mathbb{A}$	Device attribute set
$\mathbb{A}_S$	Secret attribute set
$\mathbb{A}_K$	Shared attribute set
$\mathbb{P}$	Access policy, $\mathbb{P} \subseteq \mathbb{A}_K$
P_i, U_i	Public key components
$\text{PK}_{\mathbb{A}_S}/\text{SK}_{\mathbb{A}_S}$	Public and secret key pair generated from $\mathbb{A}_S$
$\text{PK}_{\mathbb{A}_K}/\text{SK}_{\mathbb{A}_K}$	Public and secret keys generated from $\mathbb{A}_K$
u_1	Final secret key components
CT	Ciphertext
Z_p^*	Finite field over p

propose a communication architecture for Body Area Networks (BANs) and design a scheme to secure the data communication between wearable sensors and data consumers (doctors and nurses). They propose the CP-ABE and signature-based schemes to store the encrypted data at the data sink. Guo et al. [6] propose a CP-ABE scheme with constant-size decryption keys. Chen et al. [7] propose fully secure KP-ABE and CP-ABE with constant-size ciphertexts and a fully secure ABS with constant-size signatures. Odelu and Das [8] propose a lightweight and constant-size secret key ABE scheme based on ECC. However, key update/revocation and key generation are some of the limitations of the scheme in [8].

A major limitation of the existing ABE schemes [5, 7, 9, 10, 11, 12] is the complexity; these schemes require large security parameters (i.e., 1024- or 2048-bit size). Besides that, in the above cited ABE schemes, CA generates and distributes the secret keys. Nonetheless, sharing of private attributes with CA can risk the privacy, since the CA can also decrypt messages and retrieve CPS system data. Moreover, the compromise of CA can also risk the secrecy of communication between the sender and receiver. Additionally, some studies propose symmetric key schemes for resource-constrained devices. However, in large-scale Fog-CPS systems, the symmetric key management process becomes very complicated and complex. Symmetric schemes require a separate protocol for session key agreement and generation. Furthermore, when the short-size data is encrypted with a symmetric key, then the information which is revealed about the key may be critical for ciphertext-only attack.

Henceforth, it is believed a lightweight encryption scheme based on ECC wherein the CA only generates the partial key pair is the appropriate choice. ECC-based schemes require smaller security parameters (i.e., 128 or 256 bits) and therefore can be implemented in resource-limited CPS devices.

3. Proposed Fog-CPS Security Scheme

In this section, the proposed Fog-CPS scheme is presented. Table 1 lists the notations used throughout the paper.

3.1. Preliminaries.

In this section, the attributes, access structures, and the computational hard problems are discussed.

3.1.1. Secret and Shared Attribute Sets.

All CPS devices and fog nodes possess a set of attributes. Let $\mathbb{A} = \mathbb{A}_S \cup \mathbb{A}_K$ be the attribute set of each CPS device consisting of both secret $\mathbb{A}_S$ and shared $\mathbb{A}_K$ attributes. As an example, a few attributes for both sets are listed in Table 2.

3.1.2. Access Structure.

The attribute string of a device is presented with an n-bit string $a_1 a_2 \cdots a_n$. To further elaborate the attribute string, the example of shared attribute set $\mathbb{A}_K$ is considered. The attribute string is defined as follows: $a_i = 1$, if $A_i \in \mathbb{A}_K$ and $a_i = 0$, if $A_i \notin \mathbb{A}_K$. For example, if $n = 5$ and 4 attributes are considered for final key pair generation, then $\mathbb{A}_K = \{A_1, A_2, A_4, A_5\}$; the 5-bit string $\mathbb{A}_K$ becomes 11011. Likewise, an access policy is defined by $\mathbb{P}$ and specified with attributes in the shared attribute set $\mathbb{A}_K$. The access policy is also represented with n-bit string $b_1 b_2 \cdots b_n$ where $b_i = 1$,

Table 2: Attributes shared between CPS devices and fog assist node.

(a)

Secret attribute set shared between Fog-CPS entities and FA					
$\acute{A}_1$	$\acute{A}_2$	$\acute{A}_3$	$\cdots$	$\cdots$	$\mathbb{A}_S$
Device ID	IP address	Location	Malicious activities reported	Number of key updates	Trust

(b)

Shared attribute set shared between Fog-CPS entities					
A_1	A_2	A_3	$\cdots$	$\cdots$	$\mathbb{A}_K$
Entity ID	Entity type	Application ID	Application type	Data identifier	Trust

for $A_i \in \mathbb{P}$, and $b_i = 0$, for $A_i \notin \mathbb{P}$. If the access policy is defined on attributes $\{A_1, A_2, A_4\}$, then the policy string is $\mathbb{P} = 11010$. Additionally, the proposed scheme is based on the AND-gate access control structure. The attribute set $\mathbb{A}_K$ fulfills the access policy $\mathbb{P}$, if and only if $\mathbb{P} \subseteq \mathbb{A}$ and $a_i \geq b_i$, for all $i = 1, 2, \cdots, n$.

3.1.3. Computational Hard Problems. The security of the Fog-CPS scheme is based on the computational problems described below.

(1) q-Generalized Diffie-Hellman (q-GDH) Assumption [13]. Given $a_1P, a_2P, \cdots, a_qP$ in $\mathbb{G}$ and all the subset products $(\prod_{i\in S} a_i)P \in \mathbb{G}$ for any strict subset $S \subset \{1, \cdots, q\}$, it is hard to compute $(a_1 \cdots a_q)P \in \mathbb{G}$, where P is a base point in $E_p(a, b)$; $a_1, a_2, \cdots, a_q \in Z_p^*$. Since the number of subset products (elliptic curve scalar point multiplications) is exponential in q, access to all these subset products is provided through an oracle. For a vector $a = (a_1, \cdots, a_q) \in (Z_p)^q$, define $\mathcal{O}_{p,a}$ to be an oracle that for any strict subset $S \subset \{1, \cdots, q\}$ responds with $\mathcal{O}_{p,a}(S) = (\Pi_{i\in S} a_i) \in G$.

Definition 1 (q-GDH assumption). The $(t, q, \epsilon) - GDH$ assumption is satisfied in $\mathbb{G}$, if for all t-time algorithms A, the advantage $\text{Adv}_{A,q}^{GDH} = \Pr[\mathcal{A}^{\mathcal{O}_{p,a}} = (a_1 \cdots a_q)P] < \epsilon$, where $\mathbf{a} = (a_1, \cdots, a_q) \in (Z_p)^q$ and for any sufficiently small $\epsilon > 0$.*(2) q-Diffie-Hellman Inversion (q-DHI) Problem [13].* Given a $(q+1)$-tuple $(P, xP, x^2P, \cdots, x^qP) \in \mathbb{G}^{q+1}$, the problem is to compute $(1/x)P \in \mathbb{G}$ where $x \in Z_p^*$.

Definition 2 (q-DHI assumption). $\mathbb{G}$ satisfies the (t, q, ϵ) -DHI assumption, if for all t-time algorithms $\mathcal{A}$, the advantage becomes $\text{Adv}_{A,q}^{GDH} = \Pr[\mathcal{A}(P, xP, x^2P, \cdots, x^qP) = (1/x)P] < \epsilon$ for any sufficiently small $\epsilon > 0$, where the probability is over the random choice of x in Z_p^* and the random bits of $\mathcal{A}$.

It can be shown by a reduction that our computational problem is at least as hard as the discrete logarithm problem (DLP).

3.2. Assumptions. The following assumptions are made regarding the Fog-CPS entities.

(1) It is assumed that CPS devices, such as smart meter and home appliances, can be compromised and leak sensitive information

(2) The FA, fog nodes, and cloud provider are honest but curious and might try to gather as much information (from CPS devices, users, social media, and external resources) as possible that can later be used to generate the profile of CPS devices/users. Such information can be used for targeted advertisement and spamming

(3) All Fog-CPS entities will register with the FA using their secret attribute set and it would generate their partial key pair

(4) The CPS entities and fog nodes generate the final public keys of the collaborating devices

(5) Access policies based on shared attributes are shared with CPS devices, FA, fog nodes, and cloud

(6) It is assumed that the elliptic curve group parameters are preshared with entities

3.3. Fog-CPS Security Scheme Description. The proposed Fog-CPS security scheme has adopted the encryption and decryption algorithms of [8]. Furthermore, additional changes were made to the key generation algorithms of [8] as specified below:

(1) The attributes in the Fog-CPS scheme are divided into two sets, i.e., a secret $\mathbb{A}_S$ and a shared $\mathbb{A}_K$. Therefore, the key generation process is also distributed between the FA and Fog-CPS entities. Three algorithms, namely, *partial key pair generation*, *final public KeyGen*, and *final secret KeyGen*, are designed for the complete key generation process

(2) The Fog-CPS security scheme uses two elliptic curve (EC) points for each attribute instead of three as in [8]. The use of two points per attribute reduces the processing and memory overhead and makes the Fog-CPS scheme efficient but also secure.

```
1: **Input:** Secret $\mathbb{A}_S$ set.
2: **Output:** New partial key pair $PK_{\mathbb{A}_S}/SK_{\mathbb{A}_S}$.
3: Increment the counter $c$ for revoked keys.
4: If the attributes are the same, the previous computations over $t$ attributes are considered.
5: Else, perform calculations for attributes from $i = t$ to $t \pm 1$.
6: Map $t \pm 1$ attribute in $\mathbb{A}_S$ to $Z_p$ using Equation (1) and compute secret number $\acute{s}$.
        $\acute{s} = \acute{s} + s_{t\pm1}$.(13)
7: Next, compute $s$ using Equation (3).
8: Next, compute public key component $P_i$ for $i = t \pm 1$ attribute using Equation (4).
```

Algorithm 4: Partial key pair update.

```
1: **Input:** Shared $\mathbb{A}_K$ attribute set.
2: **Output:** New final public key $PK_{\mathbb{A}_K}$.
3: If the attributes are the same, the previous computations over $t$ attributes are considered.
4: Else, perform calculations for attributes from $i = t$ to $t \pm 1$.
5: Map $(t \pm 1)$ attribute in shared set $\mathbb{A}_K$ to $Z_p$. Apply hash function $H_1$ and compute $k_i$ using Equation (5). Then, compute $k$:
        $k = k + k_{t\pm1}$.(14)
6: Next, compute final public key component $U_i$, for $(t \pm 1)$ attribute using Equation (7).
```

Algorithm 5: Final public key update.

(3) Efficient key update algorithms with limited additional overhead are introduced

3.4. Fog-CPS Security Scheme Construction. The Fog-CPS security scheme consists of eight algorithms out of which seven are presented here. As previously mentioned, the encryption and decryption algorithms are adopted from [8]. However, we made a few changes to the encryption algorithm, so it is also presented, but the description of the decryption algorithm is omitted.

3.4.1. Partial Key Pair Generation $(\lambda, \mathbb{A}_S) \rightarrow PK_{\mathbb{A}_S}/SK_{\mathbb{A}_S}$. The partial key pair generation algorithm is executed by the FA which registers the Fog-CPS entities based on a secret attribute set $\mathbb{A}_S$. Algorithm 1 takes as input the security parameter λ and a set of secret attributes $\mathbb{A}_S$. λ consists of a long string of 1 s (chosen finite field) and defines the length of the secret keys and messages. It outputs the partial public/secret key pair $PK_{\mathbb{A}_S}/SK_{\mathbb{A}_S}$. Subsequently, the FA publishes the public key $PK_{\mathbb{A}_S}$ and sends the secret key $SK_{\mathbb{A}_S}$ to the CPS device. For initial communication, Fog-CPS entities can use the partial public keys. The partial public keys guarantee that Fog-CPS entities are legitimate and registered with FA.

3.4.2. Final Public KeyGen $(PK_{\mathbb{A}_S}, \mathbb{A}_K) \rightarrow PK_{\mathbb{A}_K}$. The second pair of keys, namely, the final public and secret keys, is generated from the shared attribute set. The final public keys can be generated by any CPS device or fog node that shares a set of attributes with some other entity. Algorithm 2 generates the final public key of a CPS device. It takes as input the partial public key $PK_{\mathbb{A}_S}$ generated over the secret attribute set $\mathbb{A}_S$ and the shared attribute set $\mathbb{A}_K$. It outputs the final public key $PK_{\mathbb{A}_K}$.

3.4.3. Final Secret KeyGen $(SK_{\mathbb{A}_S}, \mathbb{A}_K) \rightarrow SK_{\mathbb{A}_K}$. Algorithm 3 generates the final secret key of a CPS device. It takes as input the secret key generated over the secret attribute set $\mathbb{A}_S$ and the shared attribute set $\mathbb{A}_K$. It outputs the final secret key $SK_{\mathbb{A}_K}$.

3.4.4. Encrypt $(PK_{\mathbb{A}_K}, M, \mathbb{P}) \rightarrow CT$. The encryption algorithm takes as input the final public key $PK_{\mathbb{A}_K}$, access policy $\mathbb{P}$, and a message M. It outputs a ciphertext CT. Algorithm 7 7 presents the encryption procedure in detail.

Proposition 3. *From Equations (9) and (18), a new polynomial can be calculated as*

$$F(x, \mathbb{S}_K, \mathbb{S}_P) = \frac{f(x, \mathbb{S}_P)}{f(x, \mathbb{S}_K)} = \prod_{i=1}^{t} (x + H_4(i))^{a_i - b_i}. \quad (12)$$

It can easily be verified that $f(x, \mathbb{S}_P)/f(x, \mathbb{S}_K)$ is a polynomial function in x, if and only if $\mathbb{P} \subseteq \mathbb{A}_K$. The encryption algorithm and the secret key generation algorithms are designed in such a way that $f(x, \mathbb{S}_P)/f(x, \mathbb{S}_K)$ must be a polynomial for a successful decryption.

3.4.5. Decrypt $(CT, SK_{\mathbb{A}_K}) \rightarrow M$. The decryption algorithm takes as input the final secret key $SK_{\mathbb{A}_K}$ and ciphertext CT and outputs the plaintext message M. The construction of the decryption algorithm is the same as the one in [8].

3.4.6. Partial Key Pair Update $(\lambda, \acute{\mathbb{A}}_S) \rightarrow \acute{PK}_{\mathbb{A}_S}/\acute{SK}_{\mathbb{A}_S}$. If the secret attributes of a CPS entity are changed, then all the keys need to be updated. The key update procedure will start by regenerating the partial public/secret $PK_{\mathbb{A}_S}/SK_{\mathbb{A}_S}$ key pair. In the partial key pair update procedure, Algorithm 4 is executed. It takes the updated set of secret attributes $\mathbb{A}_S$ as input

```
1: Input: Shared $\mathbb{A}_K$ attribute set.
2: Output: New final secret key $\mathrm{SK}_{\mathbb{A}_K}$.
3: If the attributes are the same, the previous computations over $t$ attributes are considered.
4: Else, perform calculations for attributes from $i = t$ to $t \pm 1$.
5: Compute secret number $\alpha$ using Equation (8).
6: Next, compute $f(\alpha, \mathbb{A}_K)$:
       $f(\alpha, \mathbb{A}_K) = f(\alpha, \mathbb{A}_K) \cdot (\alpha + H_1(t \pm 1))^{1-a_{t\pm1}}$,(15)
   where polynomial $f(\alpha, \mathbb{A}_K)$ has been computed over $t$ attributes in Equation (9).
7: Pick two random numbers $r_u, t_u \in Z_P$ and then compute $s_u$ using Equation (10).
8: Next, compute secret key component $u_1$ using Equation (11).
```

Algorithm 6: Final secret key update.

```
1: Input:$\mathrm{PK}_{\mathbb{A}_K}$, $M$, and$\mathbb{P}$, $\mathbb{P} \subseteq \mathbb{A}_K$ .
2: Output:$\mathrm{CT} = \{\mathbb{P}, U_{m,i}, C_1, C_r, C_m\}$.
3: Create a random number $c \in \{0,1\}^{l_r}$ and compute
       $r_m = H_1(\mathbb{P}, M, c)$,   (16)
       $k_m = \mathrm{KDF}(r_m P)$,   (17)
   where KDF is a key derivation function which takes the new elliptic curve point $r_m P$ and generates a secret key $k_m$.
4: Let $\mathbb{S}_P = b_1 b_2 \cdots .b_t$ be the access policy string. Compute the corresponding $(t-1)$ degree at most polynomial function $f(x, \mathbb{S}_P)$ in $Z_p[x]$ as
       $f(x, \mathbb{S}_P) = \prod_{i=1}^{t} (x + H_1(i))^{1-b_i}$.(18)
   Let $c_i$ denote the coefficient of $x^i$ in the polynomial $f(x, \mathbb{S}_P)$.
5: Choose two one-way collision resistance hash functions, $H_2$ and $H_3$, defined as:
       $H_2{:}\{0,1\}^* \rightarrow \{0,1\}^{l_r}$,
       $H_3{:}\{0,1\}^* \rightarrow \{0,1\}^{l_m}$,
   where $l_r$ is the length of a random string, $l_m$ is the length of message $M$, $\{0,1\}^*$ is a binary string of arbitrary length, and $\{0,1\}^l$ is a binary string of length $l$. The length of the hash value is the same as the length of a random string $r$, and similarly, the hash value will be the same size as the message $M$.
6: Next, the CT which consists of three components $C_1$, $C_r$, and $C_m$ is computed. $C_1$ is a point on the elliptic curve which is computed from the polynomial $f(x, \mathbb{S}_P)$ and $U_i$ components in $\mathrm{PK}_{\mathbb{A}_K}$ corresponding to attributes in $\mathbb{P}$. $C_1$ is computed as follows:
       $U_{m,i} = r_m U_i, \quad i = 1, 2, \cdots, t - |\mathbb{P}|$,   (19)
       $C_1 = r_m \sum_{i=0}^{t} c_i U_i = r_m f(\alpha, \mathbb{P}) P$.(20)
   Next, $C_r$ is computed by a XOR operation on $k_m$ and $c$ computed in Step 3.
       $C_r = H_2(km) \oplus c$.(21)
   Lastly, $C_m$ is computed by a XOR operation on $c$ and message $M$.
       $C_m = H_3(c) \oplus M$.(22)
```

Algorithm 7: Encryption.

and generates a new $\acute{\mathrm{PK}}_{\mathbb{A}_K}/\acute{\mathrm{SK}}_{\mathbb{A}_K}$ partial key pair for the CPS device. Subsequently, the final public and secret keys are also regenerated.

3.4.7. Final Key Update $(\acute{PK}_{\mathbb{A}_S}/\acute{\mathbb{A}}_K) \rightarrow \acute{PK}_{\mathbb{A}_K}/\acute{SK}_{\mathbb{A}_K}$. If the shared attributes of a CPS device are updated, then the final public and secret keys need to be regenerated. In this case, Algorithms 5 and 6 are executed to generate the new final public and secret keys of the Fog-CPS entities. These algorithms take the updated set of shared attributes $\acute{\mathbb{A}}_S$ as input and generate the new final public and secret keys P$\acute{\mathrm{K}}_{\mathbb{A}_K}/\acute{\mathrm{SK}}_{\mathbb{A}_K}$ of the CPS device.

3.4.8. KeyRevoke. Similar to existing ABE schemes, the keys are revoked by the CA but in our application scenario FA. However, the keys can also be revoked due to the malicious behaviour of CPS devices. Three cases for key revocation have been identified:

(1) *Legitimate revoke*: in the first case, both key pairs can be revoked due to a system update, expiration date, and scheduled maintenance of the Fog-CPS system.

(2) *Malicious activity*: in the second case, the key revocation may take place due to the malicious behaviour which might be observed and/or reported by FA,

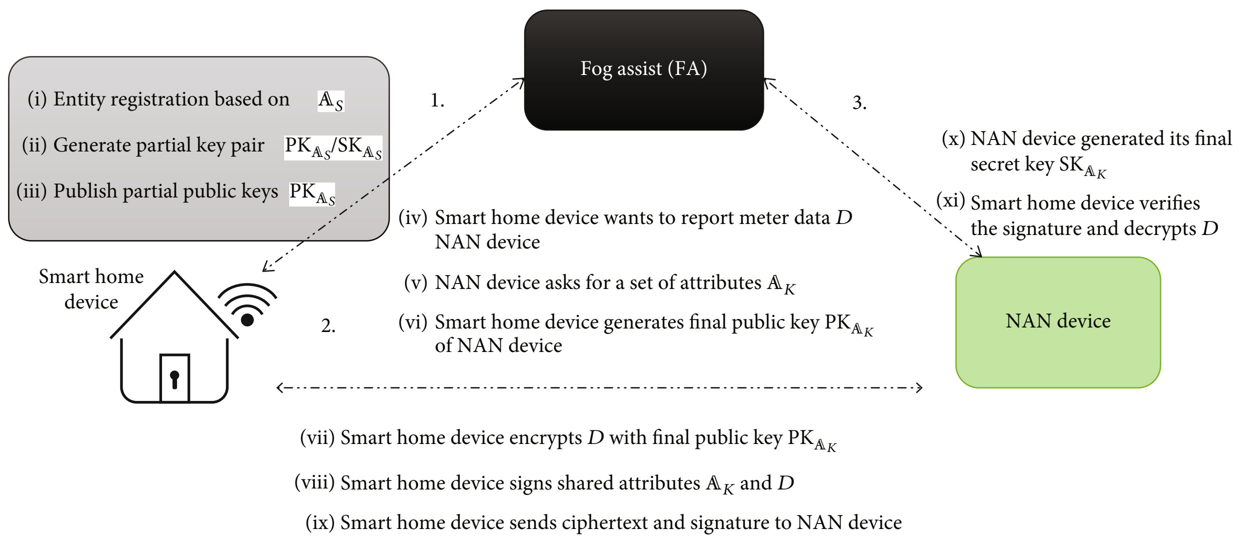
(i) Entity registration based on $\mathbb{A}_S$
(ii) Generate partial key pair $PK_{\mathbb{A}_S}/SK_{\mathbb{A}_S}$
(iii) Publish partial public keys $PK_{\mathbb{A}_S}$
Fog assist (FA)
1.
3.
(x) NAN device generated its final secret key $SK_{\mathbb{A}_K}$
(xi) Smart home device verifies the signature and decrypts D
(iv) Smart home device wants to report meter data D NAN device
(v) NAN device asks for a set of attributes $\mathbb{A}_K$
(vi) Smart home device generates final public key $PK_{\mathbb{A}_K}$ of NAN device
Smart home device
2.
NAN device
(vii) Smart home device encrypts D with final public key $PK_{\mathbb{A}_K}$
(viii) Smart home device signs shared attributes $\mathbb{A}_K$ and D
(ix) Smart home device sends ciphertext and signature to NAN device

FIGURE 2: Fog-CPS scheme application.

fog nodes, and CPS devices. Two cases of malicious activity are considered below:

(a) *Malicious FA*: the compromise of partial key pair would not trigger key revocation.

(b) *Malicious CPS devices*: the compromise of a CPS device would trigger revocation of the partial key pair. For example, if the secret key is compromised or leaked, then again keys should be revoked and regenerated.

(3) *Attribute update*: in the third case, the change in the attribute set can trigger a key revocation.

In key revocation, FA revokes the existing partial key pair and generates new keys in the first two cases, i.e., legitimate revoke and malicious activity. In the third case, the keys are regenerated as discussed in Sections 3.4.6 and 3.4.7. As the generation of final public and secret key is dependent upon the partial key pair generated over secret attributes, so the revocation of partial key pair requires the revocation of final public and secret keys. As a result, Algorithms 4–6 are designed. The proposed key update algorithms are lightweight as each revocation only incurs the overhead of one extra key component. In each subsequent key update, the t attribute counter is incremented by one.

3.4.9. Correctness. The correctness of the Fog-CPS security scheme is based on the following property. For a given pair of final keys $(PK_{\mathbb{A}_K}, SK_{\mathbb{A}_K})$ and CT generated from *Encrypt* $(PK_{\mathbb{A}_K}, M, \mathbb{P})$, the decryption algorithm *Decrypt* $(CT, \mathbb{P}, SK_{\mathbb{A}_K}, \mathbb{A}_K)$ will output the correct M, if $\mathbb{P} \subseteq \mathbb{A}_K$; otherwise, the decryption will fail.

3.5. Fog-CPS Security Scheme Application Scenario. The proposed scheme can be applied to any Fog-CPS scenario. To demonstrate it, a case in a fog-enabled smart grid power control (Fog-SGC) system is considered wherein a smart home device reports meter data D to a neighbouring area network (NAN) device. Figure 2 illustrates the communication between smart home device, NAN device, and FA. The interaction between all other entities in a Fog-SGC system would be similar as between smart home and NAN devices. Initially, all entities, namely, smart meters and fog nodes, register with FA based on their secret attribute sets $\mathbb{A}_S$. The FA executes Algorithm 1 to generate their partial key pair. Subsequently, the FA publishes the public key $PK_{\mathbb{A}_S}$ and securely transmits the secret key $SK_{\mathbb{A}_S}$ to the device.

After registration with FA, the smart home device sends a data store request to the NAN device. Upon receiving the request, the NAN device asks for a set of shared attributes $\mathbb{A}_K$. Subsequently, the smart home device generates the final public key of the NAN device by executing Algorithm 2. Next, the NAN device executes Algorithm 3 to generate the final secret key $SK_{\mathbb{A}_K}$ corresponding to the shared attribute set. Then, the smart home device encrypts D using the final public key $PK_{\mathbb{A}_K}$ of the NAN device and signs the shared attributes $\mathbb{A}_K$ using its secret key $SK_{\mathbb{A}_K}$. Subsequently, the smart home device sends CT and signature σ to the NAN device. Following this, the NAN device verifies the σ and decrypts the CT. Upon successful decryption, it gets an assurance that the smart home device possesses the required attributes and stores D.

4. Theoretical Security Analysis and Evaluation

As mentioned in the previous sections, in the proposed Fog-CPS security scheme, each entity possesses two key pairs, namely, partial and secret. So keeping that in view, the security of the proposed scheme is carefully analyzed to ensure security against the following attacks:

(i) Computing the final secret key $SK_{\mathbb{A}_K}$ from the partial secret key $SK_{\mathbb{A}_S}$

(ii) Computing the partial/final secret keys from the partial/final public keys

(iii) Computing the final secret key $\mathrm{SK}_{\mathbb{A}_K}$ from multiple ciphertexts (i.e., chosen ciphertext attack)

The Fog-CPS scheme is secure against the abovementioned attacks due to the q-Diffie-Hellman Inversion (q-DHI) problem [13], elliptic curve discrete logarithm problem (ECDLP), and the robustness of the hash functions.

Additionally, the robustness of the proposed scheme is based on two fundamental security notions of encryption schemes, namely, indistinguishability of messages and the collision resistance against secret keys. Message indistinguishability is an important security property of many encryption schemes. Given the ciphertext and the encryption key, the adversary cannot tell apart two same-length but different messages encrypted under the scheme, even if he chose the messages himself. With collision resistance, the attackers cannot pool their secret key components corresponding to a set of attributes to generate a new key which otherwise cannot be generated from their own attributes. Before presenting the security analysis against the abovementioned attacks, the notion of collision resistance as presumed in this scheme is discussed.

4.1. Collision Resistance against Secret Keys. The proposed scheme does not follow the same conventional attribute sharing as the existing ABE schemes. Attributes are only shared between two CPS devices and the FA node. So, the collision attack as presumed in existing schemes does not apply in this case. In other words, the pooling of attributes and secret key components (i.e., the collision attack) from several adversaries who do not share the attributes would not benefit in generating the secret keys. Precisely, for the security of the proposed scheme, the definition of collision resistance is modified.

In this case, it is essential to prevent a device from generating the final secret key $\mathrm{SK}_{\mathbb{A}_K}$ of another device.

Theorem 1. *It can be shown by a reduction that the computational problem in the proposed scheme is at least as hard as the discrete logarithm problem (DLP).*

Proof. Assume $\mathscr{A}$ is an algorithm that efficiently solves our problem. We can use $\mathscr{A}$ to compute the discrete logarithm of an element h to the base g as follows:

(i) Invoke $\mathscr{A}$ on input $m = g$ and $\acute{m} = h$

(ii) $\mathscr{A}$ will return will return r, s, and t such that $r^s \equiv g$ and $r^t \equiv h$ modulo p

(iii) Now we can compute a number x such that $x \cdot s \equiv 1(\mathrm{mod}\ G)$ (this can be done efficiently using the extended Euclidean algorithm because p is prime and therefore s and p are coprime)

(iv) Then, $g^{(x.t)} \equiv (r^s)x \cdot t \equiv (r^{(x.s)}t \equiv r^t \equiv h(\mathrm{mod}\ G))$

(v) Hence, $x \cdot t$ is the discrete logarithm of h to the base g

(vi) Note that the discrete logarithm problem is at least as hard as our problem since if you can compute discrete logarithms to some base r, you can s and t for given m and $\acute{m}$ such that $r^s \equiv m$ and $r^t \equiv m$ modulo G. Hence, the two problems are equally hard under the assumption that r is of order p.

4.2. Key Generation Analysis. In this section, the difficulty of deriving the partial/final secret keys from their respective public keys and final secret key derivation from multiple ciphertexts and partial secret key is analyzed. Additionally, the computational difficulty of guessing the attributes and subsequently generating the secret keys is also discussed.

4.2.1. Partial/Final Secret Key Guessing. The partial and final secret keys in Algorithms 1 and 3 are generated based on the secret and shared attributes which are mapped to Z_p. The success probability of guessing an attribute is equivalent to the complexity of hashing algorithm H_1, i.e., $2^{n/2}$ (birthday paradox). For the partial secret key $\mathrm{SK}_{\mathbb{A}_S}$, the adversary should guess all attributes in set $\mathbb{A}_S$ and the secret random number r. The secret numbers s_i which are used in partial key pair generation cannot be derived by collision attack due to its complexity. To be precise, the computational complexity is of the order of number of attributes for hash function and random guessing. This also applies to final public and secret key generation algorithms whereby the shared attributes are hashed and subsequently used in key generation. Additionally, the assumption that each entity possesses a unique set of secret and shared attributes with no overlap with the attribute set of other entities makes attribute guessing more difficult.

4.2.2. Partial/Final Secret Key Generation. The partial secret key $\mathrm{SK}_{\mathbb{A}_S}$ of a CPS device cannot be guessed due to the difficulty of deriving the secret key components $s_i \in \mathbb{A}_S$ and $r \in Z_p$ in Algorithm 1. So, in order to generate/guess the final secret key, the adversary needs to know the secret key $\mathrm{SK}_{\mathbb{A}_S}$, shared attribute set $\mathbb{A}_K$, and three secret numbers α, r_u, t_u. α is computed from the secret components s and k in algorithm 3.4.2 whereas r_u, t_u are random numbers. The secret component s can only be computed and/or known if both the FA node and the CPS device are compromised. The compromised device can leak the shared attribute set and the final secret key $\mathrm{SK}_{\mathbb{A}_K}$.

Theorem 2. *The proposed scheme is secure against an adversary $\mathscr{A}$ with knowledge of the shared attribute set $\mathbb{A}_K$ for deriving the final $\mathrm{SK}_{\mathbb{A}_K}$ secret key by collision attack.*

Proof. Having the knowledge of $\mathbb{A}_K$ is not enough for generating the $\mathrm{SK}_{\mathbb{A}_K} = u_1$, where

$$u_1 = s_u - r_u t_u\ (\mathrm{mod}\ p). \tag{23}$$

From Equation (10) in Algorithm 3,

$$\frac{1}{f(\alpha, \mathbb{A}_K)} = s_u - r_u t_u \ (\mathrm{mod}\ p), \tag{24}$$

where r_u and t_u are random numbers. The condition in Equation (24) only holds if s_u and α are known, and subsequently, the values of $r_u t_u$ can be computed. All these values can then be used to solve Equation (23). Another solution to Equation (23) is to correctly guess the random numbers r_u, t_u and compute α. The difficulty of computing α is already explained in preceding paragraphs. Hence, generating $\mathrm{SK}_{\mathbb{A}_K}$ without knowing secret components (r_u, t_u), s_u, and α is computationally infeasible for an adversary.

4.2.3. Computing the Secret Keys from Public Keys. It is underlined that the secret keys, either partial or final, cannot be computed from their respective public keys due to the intractability of the elliptic curve discrete logarithm problem (ECDLP). Given two points $P, Q \in E(F_q)$, the ECDLP problem is to find an integer x, if it exists, such that $Q = xP$. Following the same notion, the problem is to compute partial/final secret $\mathrm{SK}_{\mathbb{A}_S}/\mathrm{SK}_{\mathbb{A}_K}$ keys from public keys $\mathrm{PK}_{\mathbb{A}_S}/\mathrm{PK}_{\mathbb{A}_K}$. Like in case of $\mathrm{SK}_{\mathbb{A}_S}$, given the $\mathrm{PK}_{\mathbb{A}_S} = \{P_i = s^i P\}$ for all attributes in $\mathbb{A}_S$, the problem is to compute s^i from its corresponding public key P_i component. The ECDLP problem has to be solved for all attributes in a given attribute set. The same applies to the final secret key $\mathrm{SK}_{\mathbb{A}_K}$ generation from $\mathrm{PK}_{\mathbb{A}_K}$. For the $\mathrm{SK}_{\mathbb{A}_K}$, ECDLP is to compute k_i from given U_i and P_i. To be precise, due to the intractability of ECDLP, it is not feasible to compute secret keys from public keys.

4.2.4. Computing the Decryption Key from Ciphertext. Additionally, the proposed scheme is secure against an adversary for deriving the decryption key $r_m P$ from the ciphertext $\mathrm{CT} = \{\mathbb{P}, U_{m,i}, K_{1m}, C_{\sigma m}, C_m\}$.

Theorem 3. *Given the ciphertext $CT = \{\mathbb{P}, U_{m,i}, K_{1m}, C_1, C_r, C_m\}$, it is hard to compute decryption key $r_m P$.*

Proof. A ciphertext CT corresponding to the access policy $\mathbb{P}$ consists of the following parameters:

$$\begin{aligned} U_{m,i} &= r_m P_i, \quad i = 1, 2, \cdots, n - |\mathbb{P}|, \\ C_1 &= r_m \sum_{i=0}^{n} f_i U_i = r_m f(\alpha, \mathbb{P}) P, \\ C_r &= H_2(k_m) \oplus c, \\ C_m &= H_3(c) \oplus M. \end{aligned} \tag{25}$$

Since $\sum_{i=1}^{n-|\mathbb{P}|} U_{m,i} = rm(f(\alpha, \mathbb{P}) - f_0)P$, it is hard to compute $r_m P$ using C_1 due to the difficulty of solving the elliptic curve discrete logarithm problem. Given $U_{m,i} = rmUi = r_m k P_i$, $i = 1, 2, \cdots, q = n - |\mathbb{P}|$, this problem can be reduced to the $(q-1)$-DHI problem as follows. Let $Q = \alpha r_m P$. The parameters are then rewritten $U_{m,i} = r_m U_i = \alpha^i r_m P$ as $Q_i = U_{m,i} = \alpha^{i-1} Q$, $i = 1, 2, \cdots, q$. This implies that if an adversary $\mathscr{A}$ has the ability to solve the $(q-1)$-DHI problem, he/she can compute the key $r_m P = (1/\alpha) Q_1 = (1/\alpha) Q$ and then successfully decrypt the ciphertext CT. The following theorem proves that solving the $(q-1)$-DHI problem is as hard as the q-GDH problem.

Remark 1. From the above discussion, the proposed scheme is collision resistant against secret keys. As a result, computing the key $k_m = r_m P$ from a ciphertext CT corresponding to the access policy $\mathbb{P}$ without a valid user secret key $\mathrm{SK}_{\mathbb{A}_K}$ is as hard as the $q-\mathrm{GDH}$ problem. This implies that given $\{U_{m,1}, U_{m,2}, \cdots, U_{m,q}, C_1\}$, where $q = n - |\mathbb{P}|$, $T \in \mathbb{G}$, the $q-$ *GDH* problem reduces to the $(q-1)-\mathrm{DHI}$ problem and then decides whether T is equal to $r_m P$ or a random element in $\mathbb{G}$. But as the $q-\mathrm{GDH}$ problem is hard to solve, so would be $(q-1)-\mathrm{DHI}$. Hence, an adversary cannot derive $r_m P$ from C_1.

4.3. Network Devices Compromise Analysis. Having discussed the difficulty of generating and/or guessing the secret keys, the impact of the compromise of Fog-CPS entities on the proposed security scheme is discussed.

4.3.1. Compromise of FA. The compromise of FA can have drastic impact on the security of the Fog-CPS system. A compromised FA can reveal the partial secret keys $\mathrm{SK}_{\mathbb{A}_S}$ of Fog-CPS entities. An adversary in possession of a partial secret key $\mathrm{SK}_{\mathbb{A}_S}$ and the shared attribute set $\mathbb{A}_K$ can generate the corresponding final secret key $\mathrm{SK}_{\mathbb{A}_K}$. After having generated the final secret key, the adversary can also change the attributes agreed with FA and subsequently further compromise the network. However, if the adversary is not aware of the shared attribute set, then it cannot generate the final secret key. Moreover, the actual encryption and decryption is performed using final public and secret keys $\mathrm{PK}_{\mathbb{A}_K}/\mathrm{SK}_{\mathbb{A}_K}$ meaning that the communication between the CPS devices is still secure.

4.3.2. Compromise of CPS Device and Fog Nodes (Leakage of Final Secret Key). The compromise of CPS devices and fog nodes will only leak their own secret keys. The compromise of one set of secret keys does not risk the messages encrypted under different shared attributes therefore keys. Henceforth, legitimate CPS devices can still communicate securely.

5. Experimental Evaluation

To evaluate the performance of the proposed scheme, its algorithmic efficiency in terms of processing time and memory complexity is measured.

5.1. System Configurations. For benchmarking the time complexity, two sets of experiments are conducted to demonstrate the effectiveness of the proposed scheme on both resource-limited CPS devices and resourceful fog nodes. In the *first experiment*, the scheme is evaluated on a Raspberry Pi 3B+ model (CPS devices). It has a Quad Core 1.2 GHz,

Table 3: Processing times (sec) ($\mathbb{U} = 10$).

Schemes	Setup+KeyGen (final KeyGen)	Encrypt (1 kB)	Decrypt (1 kB)	Encrypt (1 MB)	Decrypt (1 MB)
		First experiment			
Guo et al. [6]	0.27	0.15	0.59	175.09	696.16
Odelu and Das [8]	0.18	0.30	0.08	326.7	91.9
Cheng et al. [12]	0.36	0.18	0.31	212.3	367.5
Yamada et al. [14]	0.63	1.54	1.83	1547.5	1838.3
Zhou et al. [9]	0.13	0.12	0.94	153.2	1118.5
Fog-CPS scheme	0.02	0.20	0.035	252.2	43.11
		Second experiment			
Guo et al. [6]	0.034	0.02	0.05	24.21	62.65
Odelu and Das [8]	0.03	0.09	0.02	96.71	27.28
Cheng et al. [12]	0.05	0.03	0.02	34.89	32.58
Yamada et al. [14]	0.23	0.56	0.26	564.66	288.43
Zhou et al. [9]	0.05	0.20	0.12	20.21	119.75
Fog-CPS scheme	0.008	0.06	0.01	65.29	11.14

Table 4: Processing times (sec) ($\mathbb{U} = 20$).

Schemes	Setup+KeyGen (final KeyGen)	Encrypt (1 kB)	Decrypt (1 kB)	Encrypt (1 MB)	Decrypt (1 MB)
		First experiment			
Guo et al. [6]	0.32	0.15	0.62	184.29	740.01
Odelu and Das [8]	0.36	0.31	0.08	377.4	105.8
Cheng et al. [12]	0.68	0.24	0.37	275.4	416.5
Yamada et al. [14]	1.16	1.55	1.84	1595.2	1822.3
Zhou et al. [9]	0.19	0.12	0.94	153.9	1123.7
Fog-CPS scheme	0.06	0.24	0.04	253.2	43.28
		Second experiment			
Guo et al. [6]	0.034	0.02	0.05	24.49	63.09
Odelu and Das [8]	0.05	0.09	0.02	99.21	27.87
Cheng et al. [12]	0.09	0.04	0.03	36.85	34.88
Yamada et al. [14]	0.41	0.88	0.38	911.76	383.99
Zhou et al. [9]	0.08	0.02	0.11	20.18	119.57
Fog-CPS scheme	0.017	0.06	0.011	65.47	11.26

64-bit CPU, 1 GB of RAM, a wireless LAN and Bluetooth Low Energy (BLE) on board, 100 Base Ethernet, 40-pin extended GPIO, 4 USB ports, HDMI, and micro SD port. In the *second experiment*, it is executed on a virtual machine running Ubuntu R16.04 with Python 3.6.4. on Intel (R) Core(TM) i5-4310U CPU@2.000 GHz with 8.0 GB RAM (fog nodes).

5.2. Implementation and Evaluation. The Fog-CPS scheme is compared with five other ABE schemes in Guo et al. [6], Odelu and Das [8], Cheng et al. [12], Yamada et al. [14], and Zhou et al. [9] using the Charm crypto library [15]. All security schemes, including this, are based on a selective security model. It is noted that the proposed scheme is not based on bilinear elliptic curves and can be implemented on any elliptic curve. However, in order to compare the scheme with existing ABE schemes which are based on bilinear maps, it is implemented on bilinear curves, i.e., MNT159 and SS512. On other curves, namely, prime192v1 and secp224r1, the memory overhead would be less. The proposed scheme and two others in Guo et al. [6] and Odelu and Das [8] are tested on a non-super-singular asymmetric bilinear curve (i.e., MNT159), whilst three of the schemes [9, 12, 14] have been tested on the super-singular SS512 curve. Both SS512 and MNT159 curves provide 80-bit security.

5.3. Timing Results. The execution times of all algorithms are benchmarked to compare the efficiency of different schemes. In the existing schemes in Guo et al. [6], Odelu and Das [8], Cheng et al. [12], Yamada et al. [14], and Zhou et al. [9], the *Setup* and *KeyGen* algorithms are separate. But, since there is no *Setup* in the Fog-CPS scheme, the execution time of both

Table 5: Processing times (sec) ($\mathbb{U} = 30$).

Schemes	Setup+KeyGen (final KeyGen)	Encrypt (1 kB)	Decrypt (1 kB)	Encrypt (1 MB)	Decrypt (1 MB)
		First experiment			
Guo et al. [6]	0.37	0.16	0.68	186.78	740.01
Odelu and Das [8]	0.52	0.32	0.09	375.0	105.61
Cheng et al. [12]	1.0	0.31	0.42	361.21	515.24
Yamada et al. [14]	1.69	1.55	1.84	1533.4	1838.8
Zhou et al. [9]	0.31	0.15	1.10	154.2	1128.2
Fog-CPS scheme	0.10	0.24	0.04	253.7	43.28
		Second experiment			
Guo et al. [6]	0.045	0.02	0.05	24.49	63.09
Odelu and Das [8]	0.07	0.08	0.02	97.64	27.97
Cheng et al. [12]	0.11	0.04	0.03	39.84	36.10
Yamada et al. [14]	0.62	1.38	0.59	1330.16	580.92
Zhou et al. [9]	0.10	0.02	0.11	20.18	119.57
Fog-CPS scheme	0.027	0.06	0.01	65.76	12.92

these algorithms is added and compared with the timing of the final public and secret key generation. To be precise, the final key pair generation timing of this scheme is the sum of the execution times of Algorithms 2 and 3.

For *Setup* and *KeyGen*, three different sizes of attribute universe $\mathbb{U}$ and user attribute sets $\mathbf{A}$ are considered. To be precise, an attribute universe $\mathbb{U}$ of 10, 20, and 30 attributes has been implemented for measuring the timing of the *Setup* algorithm. Likewise, for the secret key generation, a user attribute set $\mathbb{A}$ of 5, 15, and 25 attributes is taken into consideration.

Additionally, the encryption and decryption algorithms are also implemented to demonstrate that they are more efficient than the ones in Odelu and Das [8] because this scheme uses lesser elliptic curve group elements. Another reason to implement the encryption and decryption algorithms was to measure their execution timings on the Raspberry Pi, i.e., CPS devices. Henceforth, two types of benchmarks are set for measuring the times for encryption and decryption: (1) 1 kilobyte (1 kB) and (2) 1 megabyte (1 MB). These two low size messages are used because CPS devices and cloud requests are usually transmitted in low size messages. Furthermore, in the encryption algorithm, an access policy $\mathbb{P}$ of constant size, i.e., 5 attributes, is considered. For the key update, there are two cases with an increment and decrement of one attribute, i.e., $t \pm 1$. However, in our experimental evaluation, the execution time for key update is recorded in case of $t + 1$ attributes only.

Tables 3–5 list the processing times of all algorithms for the *first* and *second* experiments for attribute $\mathbb{U}$ of 10, 20, and 30, respectively. Additionally, graphs (see Figures 3–10) are also plotted for the results of the *first* experiment.

5.3.1. First Experiment. In this experiment, our implementations are carried out on the Raspberry Pi 3B+ model. Timing results are shown in Figures 3–10. From Figure 3, it can be observed that the Fog-CPS scheme takes 0.02 sec for the final key pair generation over 10 attributes, 0.06 sec over 20 attributes, and 0.10 sec over 30 attributes. Additionally, Figure 4 shows the partial key pair generation timings and Figures 5 and 6 show the final and partial key pair update timings of the Fog-CPS scheme. It can be observed that key pair update timings of this scheme are negligible due to the lightweight and efficient process. Figures 7 and 8 show the timings of encryption; this scheme takes 0.20 and 252.3 sec to encrypt a message of 1 kB and 1 MB, respectively. Similarly, for decrypting a ciphertext of 1 kB and 1 MB, it takes 0.03 sec and 43.1 sec as shown in Figures 9 and 10, respectively.

5.3.2. Second Experiment. In the second experiment, the implementations are executed on a desktop computer (configurations are mentioned in Section 5.1). Tables 3–5 list the timing results of the final key pair generation, encryption, decryption, and key pair updates. Overall, it is observed that benchmarks recorded on the Raspberry Pi 3B+ model are slower than on the desktop computer. Comparing the timings of both experiments in Table 3, it is noted that the proposed scheme is slower on the Raspberry Pi. But it is still the fastest compared to the rest of the schemes as it only takes 0.008, 0.017, and 0.02 seconds for an attribute universe of 10, 20, and 30 attributes, respectively.

Analyzing the processing times for encryption of 1 kB and 1 MB messages, it can be observed that the encryption timing of the schemes by Zhou et al. [9] and Yamada et al. [14] is almost equal and they are faster than rest of the schemes, followed by the scheme of Cheng et al. [12].

The Fog-CPS scheme is three times slower whereas the scheme by Odelu and Das [8] is four times slower than the schemes by Zhou et al. [9] and Guo et al. [6]. Likewise, the Fog-CPS security scheme is 10 times faster than the scheme by Yamada et al. [14] which is the slowest of all schemes. Comparing the timings of encryption and decryption, it is observed that in the case of encryption, the proposed scheme is a bit slower than three of the other schemes. However, in decryption, this scheme is the fastest of all; it takes only

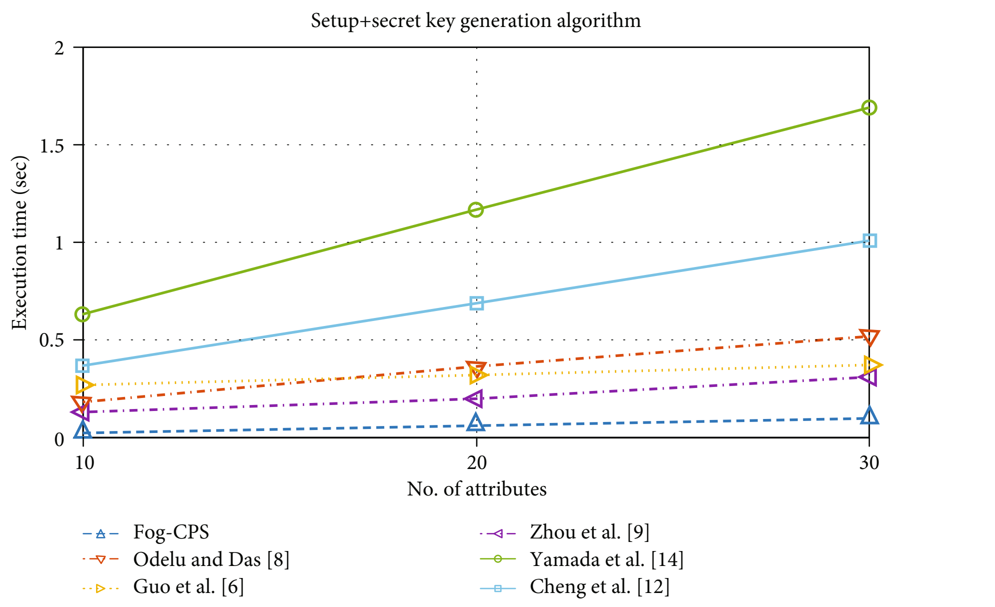

Figure 3: Final key pair generation Fog-CPS scheme (Algorithms 2 and 3) and Setup+KeyGen (other schemes).

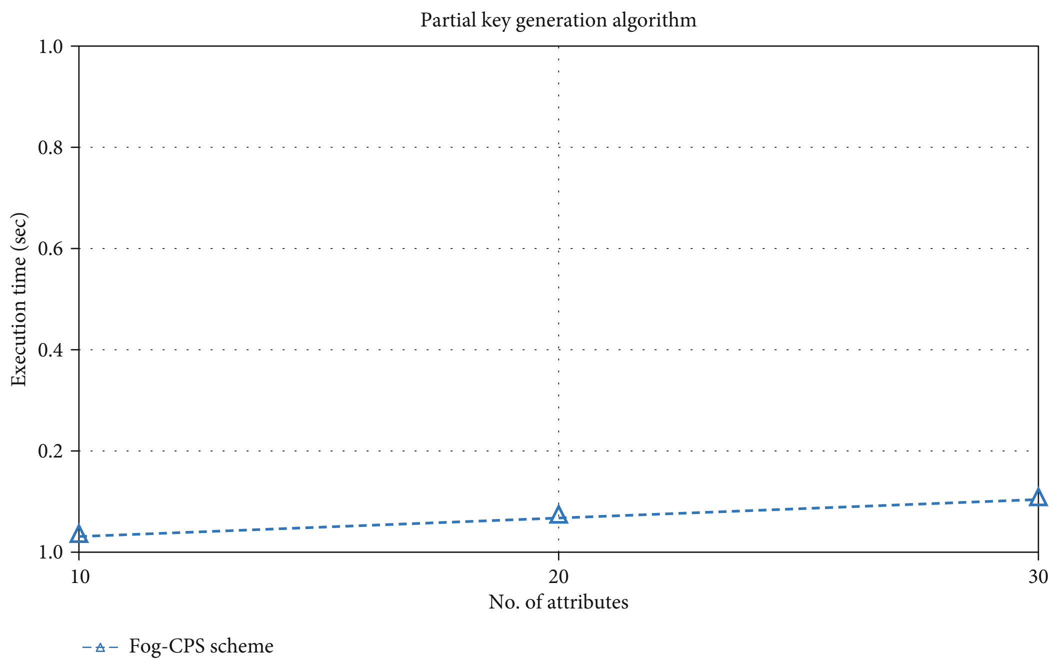

Figure 4: Partial key generation (Algorithm 2).

0.01 seconds for encryption of the 1 kB message. Following the same trend as in encryption, the scheme by Yamada et al. [14] is the slowest of all the other schemes in decryption as well. Furthermore, it can be observed that for decryption of the 1 kB message, the execution time of the scheme by Odelu and Das [8] is almost equal to that by Cheng et al. [12].

5.4. Memory Overhead. Table 6 shows the calculation of the memory overhead of each scheme. In the MNT159 curve, one group element in $\mathbb{G}$ and $\mathbb{G}_1$ takes $2_{\times} 160_{=} 320$ bits whereas one group element in $\mathbb{G}_T$ takes $2 \times 512 = 1024$ bits. Likewise, in the SS512 curve, one group element in $\mathbb{G}_1$ takes $2_{\times} 512_{=1}024$ bits whereas one group element in $\mathbb{G}_T$ takes $2 \times 1024 = 2048$ bits.

The column *Bytes* represents the total number of bytes required in all algorithms (i.e., Setup, KeyGen, Encrypt, Decrypt, and KeyRevoke) for an attribute universe $\mathbb{U}$ of 10 and an access policy $\mathbb{P}$ of 5 attributes. In the case of the proposed scheme, the memory overhead of one partial and final key pair generation and one key update in both cases is also considered when calculating the number of bytes required by each algorithm.

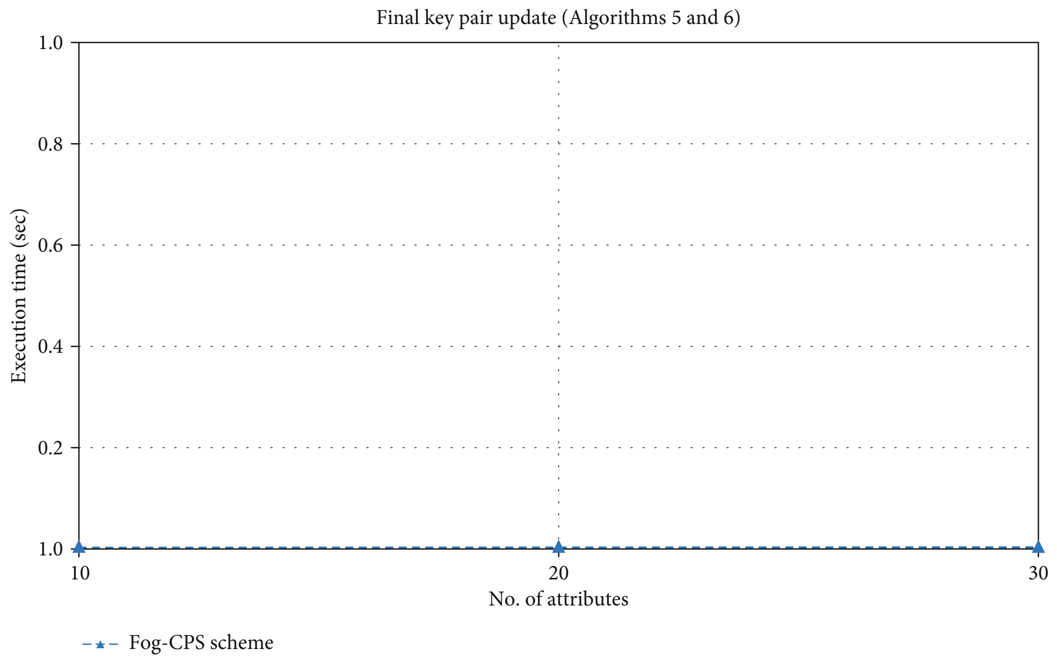

Figure 5: Final key update (Algorithms 5 and 6).

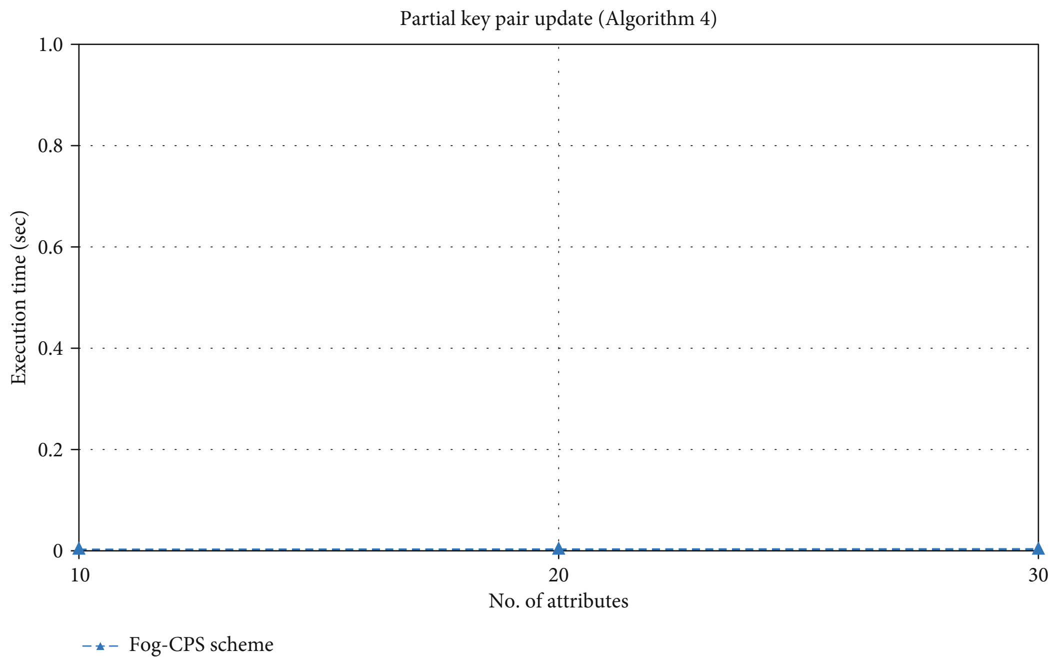

Figure 6: Partial key update (Algorithm 4).

The proposed Fog-CPS scheme is lightweight than the Odelu and Das [8] scheme as it requires fewer elliptic curve group elements. Our scheme requires $2(n+1)$ elements in $\mathbb{G}$ for generation of both partial and final public keys whereas the scheme by Odelu and Das [8] requires $3(n+1)$ elements for the public key. Likewise, for partial and final secret keys, our scheme requires only two secret elements in the finite field Z_p, whereas the scheme by Odelu and Das [8] requires three. The length of CT in our scheme is $(n-|\mathbb{P}|+2)$ group $\mathbb{G}$ elements whereas $(n-|\mathbb{P}|+3)$ in the scheme by Odelu and Das [8].

As can be seen in Table 6, the Fog-CPS scheme has the lowest memory overhead, i.e., 1340 bytes followed by the Guo et al. [6] and Odelu and Das [8] schemes which take 1636 and 1760 bytes, respectively. The Zhou et al. [9] scheme incurs the highest overhead of 9876 bytes. It is noted that for

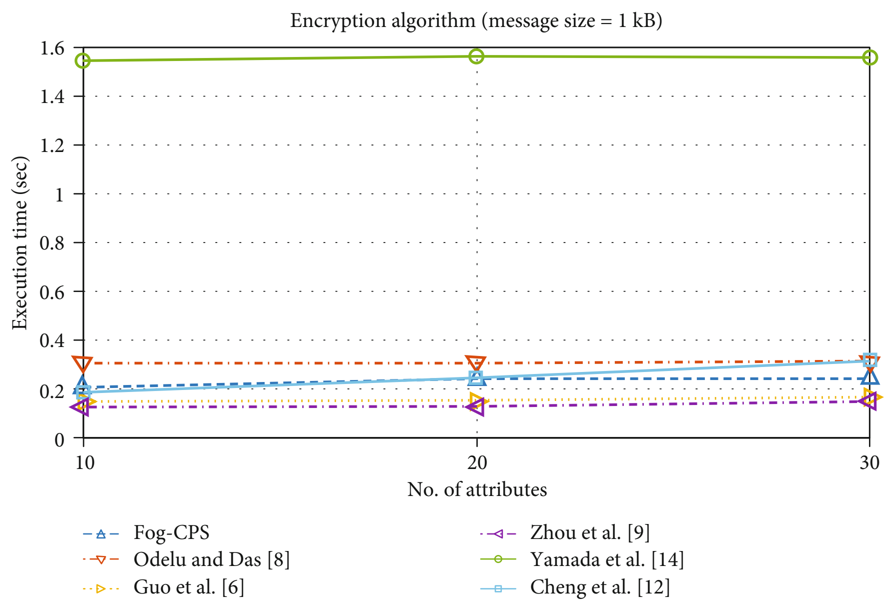

Figure 7: Processing time (sec) for encryption algorithm (1 kB).

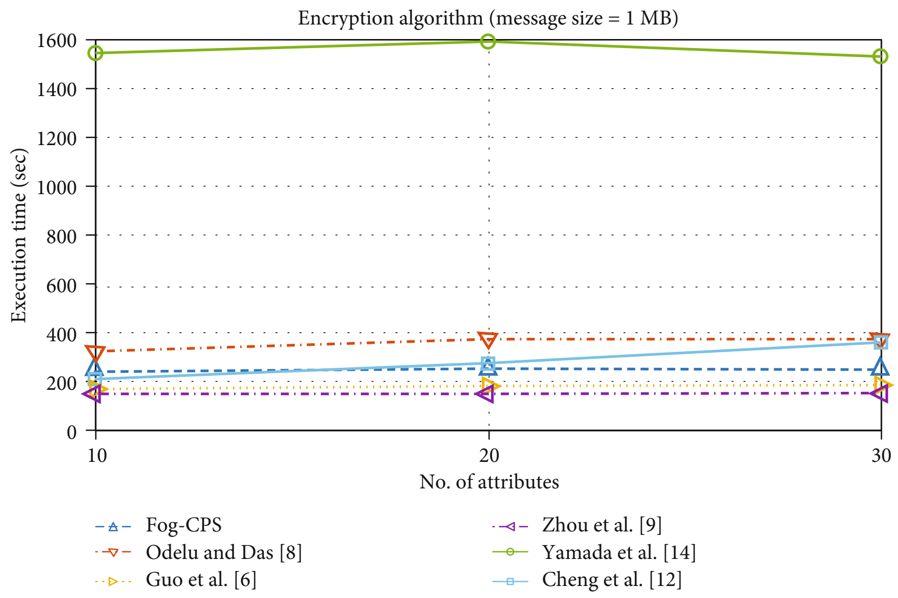

Figure 8: Processing time (sec) for encryption algorithm (1 MB).

the proposed scheme, the memory overhead of one final public and secret key over a shared attribute set ($|\mathbb{A}_k| = 10$) has also been considered. Moreover, for the *Setup* and *KeyGen* algorithms, i.e., partial key pair generation in this case, the Fog-CPS scheme has the lowest overhead compared to all other schemes. The final secret key in the proposed scheme only requires one element of the order of base point on the elliptic curve $\mathbb{G}$.

In the Guo et al. [6] scheme, there are $(2n+1)$ elements in $\mathbb{G}_1$ and one in G_T for PK, two elements in $\mathbb{G}_1$ for SK, and $(n - |\mathbb{P}| + 2)$ elements in $\mathbb{G}_1$ for CT. In the Zhou et al. [9] scheme, there are $(6n+1)\mathbb{G}_1$ and $(2|\mathbb{A}|+1)$ group elements in PK and SK, respectively, whereas CT has 2 group elements in $\mathbb{G}_1$ and one in $\mathbb{G}_T$. The PK in the Cheng et al. [12] scheme has $2n$ elements in $\mathbb{G}_1$ and two in $\mathbb{G}_T$; the SK has $|\mathbb{A}|+1$ elements in $\mathbb{G}_1$, whereas the CT has 2 elements in $\mathbb{G}_1$ and one in $\mathbb{G}_T$. In the Yamada et al. [14] scheme, PK contains 6 elements in $\mathbb{G}_1$ and one in G_T, SK contains $(4|\mathbb{A}| + 2)$ elements in $\mathbb{G}_1$, and CT contains $3(|\mathbb{P}|+1)$ in $\mathbb{G}_1$. The Zhou et al. [9] scheme has the highest memory overhead followed by the Cheng et al. [12] and Yamada et al. [14] schemes. Both the Zhou et al. [9] and Cheng et al. [12] schemes have constant size ciphertexts which only require 2 group elements in $\mathbb{G}_1$ and one element in $\mathbb{G}_T$. The CT in the Fog-CPS scheme and the Guo et al. [6] scheme have almost the same number of elements. In the key update

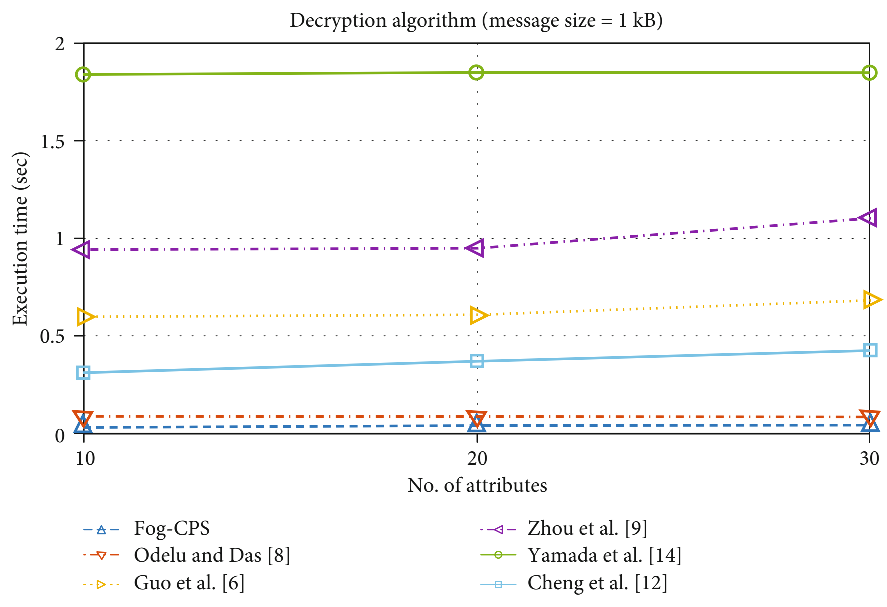

Figure 9: Processing time (sec) for decryption algorithm (1 kB).

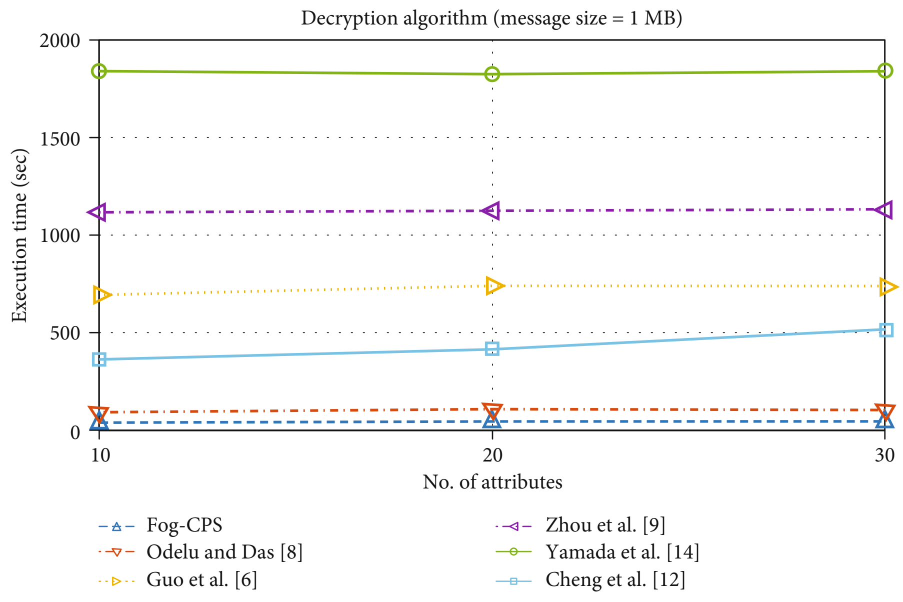

Figure 10: Processing time (sec) for decryption algorithm (1 MB).

Algorithms 4–6, this scheme only requires 2 elements in $\mathbb{G}$ and 2 elements of the order of base point on the elliptic curve $\mathbb{G}$.

5.5. Computational Overhead. The computational overhead of all schemes is summarized in Table 7. The column *Total operations* represents the total number of operations by considering all algorithms, i.e., Setup, KeyGen, Encrypt, and Decrypt. In the proposed scheme, the computational overhead of one partial and final key pair generation and update are also considered. From Table 7, it is observed that the Fog-CPS scheme introduces the lowest computational overhead than all other schemes which are based on bilinear maps and elliptic curves.

For partial and final public key generation, the Fog-CPS scheme requires $(2n+2)$ scalar multiplications in the elliptic curve group $\mathbb{G}$. Likewise, for *Encrypt* and *Decrypt*, $(n-\mathbb{P}+1)$ and $(n-\mathbb{P}+2)$ scalar point multiplications are required, respectively. Moreover, in the key update algorithms, i.e., 4, 5, and 6, this scheme only requires two scalar multiplications.

For the *Setup* and *KeyGen* algorithms, the Guo et al. [6] scheme requires $(2n+2)$ exponentiations and 1 pairing operation, respectively. In addition, the *Encrypt* algorithm requires $(n-|\mathbb{P}|+3)$ exponentiations and a single pairing whereas *Decrypt* requires $(2|n-\mathbb{P}|+4)$ exponentiations and 4 pairing operations. The *Setup* and *KeyGen* algorithms in the Zhou et al. [9] scheme require $6n+1$ and $(2|\mathbb{A}|+1)$

Table 6: Memory overhead.

Schemes	Setup	KeyGen	Encrypt	Decrypt	Key update	Bytes
Fog-CPS scheme	$PK_{\mathbb{A}_s} = (n+1)\mathbb{G}$	$SK_{\mathbb{A}_s} = 1 \times O(P)$	$(n - \lvert\mathbb{P}\rvert + 2)\mathbb{G}$	M	$2 \times O(P) + 2\mathbb{G}$	1340
Odelu and Das [8]	$PK = (3n+1)\mathbb{G}$	$SK = 2 \times O(P)$	$(n\lvert\mathbb{P}\rvert + 3)\mathbb{G} + L$	M	N/A	1760
Guo et al. [6]	$PK = (2n+1)\mathbb{G}_1 + \mathbb{G}_T$ $MSK = \mathbb{G}_1$	$SK = 2\mathbb{G}_1$	$(n - \lvert\mathbb{P}\rvert + 2)\mathbb{G}_1 + \mathbb{G}_T$	M	N/A	1636
Zhou et al. [9]	$PK = (6n+1)\mathbb{G}_1$ $MSK = 2Z_p$	$SK = (2\lvert\mathbb{A}\rvert + 1)\mathbb{G}_1$	$2\mathbb{G}_1 + \mathbb{G}_T$	M	N/A	9876
Yamada et al. [14]	$PK = 6\mathbb{G}_1 + \mathbb{G}_T$ $MSK = 2Z_p$	$SK = (4\lvert\mathbb{A}\rvert + 2)\mathbb{G}_1$	$3(\lvert\mathbb{P}\rvert + 1)\mathbb{G}_1$	M	N/A	6292
Cheng et al. [12]	$PK = 2n\mathbb{G}_1 + 2\mathbb{G}_T$ $MSK = 2n\mathbb{G}_1$	$SK = (\lvert\mathbb{A}\rvert + 1)\mathbb{G}_1$	$2\mathbb{G}_1 + \mathbb{G}_T$	$\mathbb{G}_T$	N/A	6932

Note: $\mathbb{G}_1$ and $\mathbb{G}_T$: prime order pairing groups (160 bits); exp: exponent operation; $\ddot{p}$: pairing operation; $\lvert\mathbb{A}\rvert$ and $\lvert\mathbb{P}\rvert$: no. of attributes in secret key and access policy; $\mathbb{G}$: elliptic curve group defined over finite field Z_p; $O(P)$: order of the base point (160 bits in Z_p); L: length of plaintext message M.

Table 7: Computational overhead.

Schemes	Setup	KeyGen	Encrypt	Decrypt	KeyRevoke	Total operations
Fog-CPS scheme	$(n+1)$ecm	N/A	$(n - \mathbb{P} + 1)$ecm	$(n - \mathbb{P} + 2)$ecm	2ecm	37ecm
Odelu and Das [8]	$3(n+1)$ecm	N/A	$(n - \mathbb{P} + 2)$ecm	$(n - \mathbb{P} + 3)$ecm	N/A	48ecm
Guo et al. [6]	$(2n)$ exp + $\ddot{p}$	2 exp	$(n - \lvert\mathbb{P}\rvert + 3)$ exp + $\ddot{p}$	$(2\lvert n - \mathbb{P}\rvert + 4)$ exp + $4\ddot{p}$	N/A	44 exp + $6\ddot{p}$
Zhou et al. [9]	$(6n+1)$ exp	$(2\lvert\mathbb{A}\rvert + 1)$ exp	3 exp + $\ddot{p}$	$(2\lvert\mathbb{P}\rvert + 1)$ exp + $(2\lvert\mathbb{P}\rvert + 1)\ddot{p}$	N/A	96 exp + $12\ddot{p}$
Yamada et al. [14]	1 exp	$(4\lvert\mathbb{A}\rvert + 2)$ exp	$(3\lvert\mathbb{P}\rvert + 1)$ exp	$2\lvert\mathbb{P}\rvert$ exp + $2\lvert\mathbb{P}\rvert\ddot{p}$	N/A	69 exp + $10\ddot{p}$
Cheng et al. [12]	n exp + $n\ddot{p}$	$\lvert\mathbb{A}\rvert$ exp	3 exp	$2\ddot{p}$	N/A	23 exp + $12\ddot{p}$

Note: exp: exponentiation operation; $\ddot{p}$: pairing operation; ecm: scalar point multiplication in the elliptic curve group $\mathbb{G}$.

exponentiation operations, respectively. Similarly, for *Encrypt*, it requires 3 exponentiations and 1 pairing, whereas for *Decrypt*, $(2\lvert\mathbb{P}\rvert + 1)$ exponentiation and pairing operations are required. Moreover, for the *Setup* and *KeyGen* algorithms, the Cheng et al. [12] scheme requires $(n + \lvert\mathbb{A}\rvert)$ exponentiation and n pairing operations. On the contrary, it requires 3 exponentiation and 2 pairing operations for the *Encrypt* and *Decrypt* algorithms, respectively. In the Yamada et al. [14] scheme, $(4\lvert\mathbb{A}\rvert + 3)$ exponentiations are required for both the *Setup* and *KeyGen* algorithms. However, the *Encrypt* algorithm requires $(3\lvert\mathbb{P}\rvert + 1)$ exponentiations and *Decrypt* requires $2\lvert\mathbb{P}\rvert$ exponentiation and pairing operations.

6. Conclusion

The security and privacy challenges posed by Fog-CPS systems can affect both individuals and systems. The heterogeneous nature of CPS devices with varying degrees of computation and storage capacity also make it a challenging task to devise the security solutions. It is therefore essential to find lightweight solutions which eliminate the need for applying different security mechanisms at different layers of Fog-CPS. Moreover, the devised schemes must also enable the users/CPS devices to have control over their data without risking their privacy of information (e.g., identities, location, data, and type of service used) with other entities, such as the fog nodes and/or cloud provider. The existing PKE and ABE schemes are computationally expensive and, therefore, not suitable for resource-constrained devices. Moreover, the sharing of attributes with the FA risks the privacy of CPS devices as it can also decrypt the messages. Considering the limitations of existing schemes, in this paper, a lightweight encryption scheme is proposed. In the Fog-CPS scheme, the CPS devices generate the keys without relying on a FA. In our proposed security scheme, the above problems are addressed by dividing the attribute set into secret and shared attributes. The secret attributes are used for entity registration with FA, whereas the shared attributes are only shared with the collaborating CPS devices and employed to generate the final public and secret keys. The compromise of FA would not endanger the secrecy of messages among CPS devices, as they can still securely communicate. Furthermore, the key update process of the proposed scheme is very lightweight, since the key generation process as found in Algorithms 1–3 is not repeated.

Another novel aspect of the Fog-CPS scheme is that it is based on elliptic curve cryptography which supports smaller key sizes and is highly suitable for resource-constrained CPS devices. The experimental evaluation shows that the Fog-CPS security scheme outperforms other ABE schemes based on bilinear pairing and elliptic curves. In the future, the authors will investigate how the proposed scheme can be further improved to support more expressive access structures and an adaptive security model.

Data Availability

This study does not involve any personal data and other datasets.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

References

[1] K. Mahmood, S. A. Chaudhry, H. Naqvi, S. Kumari, X. Li, and A. K. Sangaiah, “An elliptic curve cryptography based lightweight authentication scheme for smart grid communication,” *Future Generation Computer Systems*, vol. 81, no. 1, pp. 557–565, 2018.

[2] O. Kocabas, T. Soyata, and M. K. Aktas, “Emerging security mechanisms for medical cyber physical systems,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 13, no. 3, pp. 401–416, 2016.

[3] S. Chen, M. Ma, and Z. Luo, “An authentication framework for multi-domain machine-to-machine communication in cyber-physical systems,” in *2015 IEEE Globecom Workshops (GC Wkshps)*, 2015.

[4] S. Challa, M. Wazid, A. K. Das et al., “Secure signature-based authenticated key establishment scheme for future IoT applications,” *IEEE Access*, vol. 5, pp. 3028–3043, 2017.

[5] C. Hu, H. Li, Y. Huo, T. Xiang, and X. Liao, “Secure and efficient data communication protocol for wireless body area networks,” *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, no. 2, pp. 94–107, 2016.

[6] F. Guo, Y. Mu, W. Susilo, D. S. Wong, and V. Varadharajan, “CP-ABE with constant-size keys for lightweight devices,” *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 5, pp. 763–771, 2014.

[7] C. Chen, J. Chen, H. W. Lim et al., “Fully secure attribute-based systems with short ciphertexts/signatures and threshold access structures,” in *Proceedings of the 13th International Conference on Topics in Cryptology*, pp. 50–67, San Francisco, CA, USA, 2013, Springer-Verlag.

[8] V. Odelu and A. K. Das, “Design of a new CP-ABE with constant-size secret keys for lightweight devices using elliptic curve cryptography,” *Security and Communication Networks*, vol. 9, no. 17, 4059 pages, 2016.

[9] Z. Zhou, D. Huang, and Z. Wang, *Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption*, vol. 64, no. 1, 2015IEEE Transactions on Computers, 2015.

[10] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in *2007 IEEE Symposium on Security and Privacy (SP '07)*, Berkeley, CA, USA, 2007.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proceedings of the 13th ACM conference on Computer and communications security - CCS '06*, pp. 89–98, 2006.

[12] C. Chen, Z. Zhang, and D. Feng, “Efficient ciphertext policy attribute-based encryption with constant-size ciphertext and constant computation-cost,” in *Provable Security: 5th International Conference, ProvSec Proceedings*, X. Boyen and X. Chen, Eds., pp. 84–101, Springer, Berlin, Heidelberg, 2011.

[13] X. Boyen and D. Boneh, “Efficient selective-ID secure identity-based encryption without random oracles,” in *Advances in Cryptology - EUROCRYPT 2004sss*, Springer, 2004.

[14] S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro, “A framework and compact constructions for non-monotonic attribute-based encryption,” in *Public-Key Cryptography – PKC 2014*, vol. 8383, Springer, Berlin, Heidelberg, 2014.

[15] J. A. Akinyele, C. Garman, I. Miers et al., “Charm: a framework for rapidly prototyping cryptosystems,” *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.