IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON SECURITY AND PRIVACY IN EMERGING DECENTRALIZED COMMUNICATION ENVIRONMENTS

Received September 12, 2019, accepted September 30, 2019, date of publication October 8, 2019,
date of current version October 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946202

An Efficient Outsourced Privacy Preserving Machine Learning Scheme With Public Verifiability

ALZUBAIR HASSAN[1,2], RAFIK HAMZA[1,2], HONGYANG YAN[1,2], AND PING LI[3]

[1]School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China
[2]Peng Cheng Laboratory, Shenzhen 518055, China
[3]South China Normal University, Guangzhou 510631, China

Corresponding authors: Alzubair Hassan (alzubairuofk@gmail.com) and Hongyang Yan (hyang.yan@foxmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61902081, Grant 61802078, and Grant 61702126, and in part by the China Postdoctoral Science Foundation under Grant 204728.

ABSTRACT Cloud computing has been widely applied in numerous applications for storage and data analytics tasks. However, cloud servers engaged through a third party cannot be fully trusted by multiple data users. Thus, security and privacy concerns become the main obstructions to use machine learning services, especially with multiple data providers. Additionally, some recent outsourcing machine learning schemes have been proposed in order to preserve the privacy of data providers. Yet, these schemes cannot satisfy the property of public verifiability. In this paper, we present an efficient privacy-preserving machine learning scheme for multiple data providers. The proposed scheme allows all participants in the system model to publicly verify the correctness of the encrypted data. Furthermore, a unidirectional proxy re-encryption (UPRE) scheme is employed to reduce the high computational costs along with multiple data providers. The cloud server embeds noise in the encrypted data, allowing the analytics to apply machine learning techniques and preserve the privacy of data providers' information. The results and experiments tests demonstrate that the proposed scheme has the ability to reduce computational costs and communication overheads.

INDEX TERMS Cloud computing, machine learning, public verifiability, proxy re-encryption, differential privacy.

I. INTRODUCTION

Cloud computing, with its high data processing capabilities, is important to all applications that require high processing costs such as data processing machine learning [1]. Nonetheless, it is not appropriate to trust a third-party-based cloud system, especially with storing sensitive data. Cloud computing suffers from several security issues that represent highly debated topics. Cloud computing provides accessible computing services using on-demand, elastic, and easy-to-use techniques. Indeed, cloud computing provides many resources but also possesses crucial security issues. Third-party storage introduces different potential risks, especially concerning data security [2].

The data storage paradigm in the cloud brings several challenges and issues which have a huge influence on the security of the system [3], [4]. Data integrity verification at untrusted servers is one of the main issues with cloud systems. Mainly, the existing schemes falling into two categories: private verifiability and public verifiability. The private verifiability can deliver higher system efficiency, while the public verifiability allows anyone, not just the data providers, to challenge the cloud server for the data correctness and without holding private information. Cloud systems attempt to employ distinct security techniques. Yet, most of these systems cannot guarantee either users' privacy or data confidentiality without using multi-layer cryptography techniques [5]–[7]. In this case, encryption is the primary technique used to ensure data security, where data are encrypted and then stored in the cloud [8], [9]. Encrypted data exploitation is also extremely difficult amidst the high complexity.

Homomorphic encryption displays a promising role in cloud computing, developing the privacy of data providers. Homomorphic encryption gives a way to achieve several services on encrypted data and improves cloud users' privacy. Accordingly, the companies store encrypted data in a

The associate editor coordinating the review of this manuscript and approving it for publication was Zheli Liu.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

public cloud as well as perform analytic services of the cloud provider's on the encrypted data. Homomorphic encryption has properties that make these encrypted data useful for companies such as random self-reducibility, re-randomize encryption, and verifiable encryption [10]. Nevertheless, the homomorphic encryption still suffers from the high computational costs. Researchers presented several contributions such as fully homomorphic encryption [11], [12], Partially Homomorphic Encryption (PHE) [13], [14], and Somewhat Homomorphic Encryption (SWHE) [15], [16]. Herein, differential privacy is employed to guarantee the privacy of the users' data in the cloud. Differential privacy encourages the companies to collect and share aggregate information regarding the customers, at the same time they can maintain the privacy of their customers. Differential privacy displays with a probabilistic form, where the differential privacy algorithm outputs a distribution that changes little in the dataset and does not affect the privacy of an individual's data [17].

The foremost issue with cloud systems concerns the security and data privacy stored in the cloud hosting system. Most companies question their data security, and how they can trust storing their sensitive data through third-party services outside their off-line databases [18]. To this end, cryptography techniques have been proposed using various approaches that guarantee data security and users' privacy. At the user side, data encryption should be sufficient and considered as a standard form of defense that provides a high level of security in the cloud system [19].

To overcome the above issues, it is important to propose an effective privacy scheme based on machine learning given multiple data providers. Any proposed solution should reduce the cost of implementation and maintain the privacy of the participants [20]–[22]. Furthermore, it is important to address multiple data providers' problems. For example, Li *et al.* [23] proposed a privacy-preserving machine learning framework dealing with multiple data providers. However, this solution came with a high computational cost due to the dependence on integer factorization in their proposed framework. Additionally, none of the participating components in their proposed model can publicly verify the correctness of the outsourced data. This issue increases the overhead for all parties. Furthermore, the analyst should start the transaction with data providers through the cloud system and be online during communication.

This paper aims to overcome the mentioned above issues by proposing an efficient privacy-preserving machine learning scheme for multi-providers data in the cloud system. We propose a privacy-preserving framework using the additive homomorphic encryption scheme. First, the data providers encrypt their sensitive data using the unidirectional proxy re-encryption scheme and uploaded the ciphertexts to the proxy server cloud. Then, the cloud re-encrypt the received ciphertexts with a generated noise-data using partially homomorphic encryption of the Hashed-ElGamal scheme. Finally, the cloud will send the noisy-ciphertext to the analytic to perform the machine learning techniques for his predictive analytics. All the components of the proposed framework can publicly check the correctness of ciphertexts before performing any operation which ensures public verifiability. The proposed framework guarantees a high level of security and preserves the privacy of users.

The main contributions are highlighted as follows.

- This paper proposes an efficient privacy-preserving machine learning scheme with public verifiability.
- The unidirectional proxy re-encryption allows different data providers to delegate their data using the same public key.
- All the participating parties in our scheme can check the validity of the ciphertext before any operations are performed. This feature can reduce the time that checks for invalid ciphertext.
- This scheme can protect the privacy of the providers' data in the cloud and of the data analyst.
- This scheme uses ϵ-differential privacy (ϵ-DP), which improves the accuracy of applying machine learning techniques.

This paper is presented as follows. The related works are discussed in Section II. The preliminaries are given in Section III. The proposed scheme is explained in Section IV, while the result and discussion are introduced in Section V. The security analysis of our protocol is discussed in Section VI. The Finally, the conclusions are given in Section VII.

II. RELATED WORK

Most companies currently believe that machine learning will be a key customer expectation [24]. In this regard, machine learning performs an important role in technology generally, especially upon cloud computing. The extensive developments of the machine learning community have reduced the network overhead. However, these developments affect the computational cost, as most applications have high computational costs [1], [25]. Many machine learning techniques have been adopted to automatically employ complex mathematical computations, thereby suffering high computational costs.

Recently, machine learning over encrypted data has become an important topic in industry and academy. Various approaches have been introduced to overcome these challenges such as Partial traditional homomorphic encryption schemes. Unfortunately, these approaches are inefficient because they suffer from high computational costs, high network overhead and certain security issues. Several protocols have been proposed by researchers such as [19], [26]–[28]. For instance, Chen and Zhong [29] introduced a two-party distributed algorithm to preserve privacy for back-propagation neural networks (BPNNs). Their scheme allows the two parties to train their data while ensuring that the data are secure. They only considered training datasets that are vertically partitioned. To improve on previous work, Bansal *et al.* [30] proposed another algorithm that can be applied when the dataset is arbitrarily partitioned. Both Chen and Zhong [29] and Bansal*et al.* [30] used the

homomorphic properties to protect the privacy of the two parties.

The aforementioned works do not work properly in multi-party environments because of the high communication overhead. Samet and Miri [31] presented a protocol to protect the privacy of both the input data and the created learning model in a BPNN and extreme learning machine. Their protocol can be applied when the dataset is vertically or horizontally partitioned. Graepel *et al.* [32] proposed a scheme to ensure the confidentiality of the encrypted data during machine learning over these data in the training and test phases. Their algorithm can be applied to two types of classification algorithms: linear means and Fisher's linear discriminant [32]. Liu *et al.* [33] worked on preserving the users' privacy in social networks and keeping their sensitive information secure, considering the identity disclosure problem in weighted social graphs.

Nowadays, with the development of cloud computing and outsourcing, several techniques have proposed to guarantee users' privacy. For example, Wei *et al.* [34] presented practical outsourcing algorithms for exponentiation computation based on homomorphic mapping. Gao *et al.* [35] worked to compute the social contiguity between users to identify potential friends and keep their privacy based on a proxy re-encryption scheme with additive homomorphism.

The notation of differential privacy approach has been introduced to overcome privacy and accuracy issues [36]. Dwork and Roth [37] proposed the ϵ-DP approach, which considers how much data can be revealed. Consequently, researchers have introduced various approaches based on the formal definition of DP, for instance, computational differential privacy [36] as well as the differential privacy consensus algorithm [38], as discussed herein. Recently, Li *et al.* [23] proposed a privacy-preserving machine learning framework that is suitable to work with multiple data providers. They employed a double decryption public-key encryption scheme to ensure security and user privacy. However, their framework has high computational costs and communication overhead. Additionally, the framework consuming time on the invalid ciphertext.

III. PRELIMINARIES

In this part, we illustrate some related notations and concepts. First, the computational Diffie-Hellman assumption is described below. We also introduce the divisible computational Diffie-Hellman assumption, which concerns the difficulty of measuring the discrete logarithm in cyclic groups. Then, we present unidirectional proxy re-encryption, which is primary for the proposed protocol implementation. We illustrate the partial homomorphic hashed- ElGamal encryption scheme for predictive analytics. Finally, we present a brief introduction to Differential Privacy

A. COMPUTATIONAL DIFFIE-HELLMAN (CDH)

Let assume that we have $g, g^a, g^b \in \mathbb{G}$, where $\forall\{a, b\} \in \mathbb{Z}_q^*$ and $\mathbb{G}$ is a cyclic multiplicative group with prime order q, then $g^{ab} \in \mathbb{G}$ cannot be computed due to its difficulty.

B. DECISION DIFFIE-HELLMAN (DDH)

Let assume that we have two distributions $A = (g^x, g^y, g^{xy})$ and $B = (g^x, g^y, g^z)$ for randomly distributed $x, y, z \longleftarrow \mathbb{Z}_q$. Distinguish A from B [39].

C. DIVISIBLE COMPUTATIONAL DIFFIE-HELLMAN (DCDH)

In this part, we assume that we have $g, g^a, g^b \in \mathbb{G}$, where $\forall\{a, b\} \in \mathbb{Z}_q^*$, then $g^{b/a} \in \mathbb{G}$ cannot be computed due to its difficulty. The DCDH and CDH are equivalent in the same group [40].

D. UNIDIRECTIONAL PROXY RE-ENCRYPTION (UPRE) SCHEME

The definition of the unidirectional PRE scheme [41] is composed of six algorithms, described as follows:

- *Initialization*(k)**:** This algorithm uses a security parameter k as input. Then, the algorithm returns the public parameters *param*. Additionally, the message space M description is given in this algorithm.
- *KeyGen*()**:** This algorithm calculates the users' public key and corresponding private key pair (pk_{u_i}, sk_{u_i}).
- *ReKeyGen*(sk_{u_i}, pk_a)**:** This algorithm uses the private key of the delegator sk_{u_i} and the public key of the delegate pk_a. Then, the algorithm returns a re-encryption key $rk_{u_i \to a}$.
- *Enc*(pk_{u_i}, m)**:** This algorithm uses the public key pk_{u_i}, the delegator and the message $m \in \mathcal{M}$. Then, the algorithm returns a ciphertext $\mathcal{C}_i$ under pk_{u_i}.
- *ReEnc*$(rk_{u_i \to a}, \mathcal{C}_i, pk_{u_i}, pk_a)$: This algorithm uses $rk_{u_i \to a}$ as input to $\mathcal{C}_i$. Then, the algorithm returns a ciphertext $\mathcal{C}_a$ under the public key pk_a.
- *Dec*$(\mathbf{sk}, \mathcal{C})$**:** This algorithm uses sk and $\mathcal{C}$. Then, if the ciphertext is valid, the algorithm returns the plain message $m \in \mathcal{M}$; otherwise, the algorithm throws an error. This algorithm uses for the delegator *User Dec()* and the delegate *Analyst Dec()*.

E. PARTIAL HOMOMORPHIC HASHED- ELGAMAL ENCRYPTION SCHEME

Hash-ElGamal is partially homomorphic using only xor operator. Assume that the encryption of message m is known, then, anyone can compute the encryption of a message $m' = m \oplus K$ for any selected value $K \in \{0, 1\}$. Lets us assume that we have $C = (c_1, c_2)$ with $c_1 = g^r$, and $c_2 = m \oplus h(y^r)$. Then, we can have $C' = (c_1, c_2')$ with $c_2' = K \oplus c_2$. This is the hash-ElGamal encryption of m'. The partially homomorphic based on Hash-ElGamal encryption supports adding a mask on the ciphertext. Furthermore, this encryption is known to be one-way under the computational Diffie-Hellman assumption, and indistinguishability holds under the DDH assumption [39].

F. DIFFERENTIAL PRIVACY

Differential privacy is a probabilistic mechanism, where the algorithm outputs a distribution that changes little in the

dataset and does not affect the privacy of an individual's data. Assuming that d_1 and d_2 are two data sets; d_1, and d_2 can be neighboring if they are different in only one record [17]. Particularly, DP guarantees the security of the distributed data by adding measured perturbations to the ciphertext using the principal of homomorphic encryption [42].

Definition 1 (ϵ-DP): A randomized mechanism R is ϵ-differentially private if we have any pair of neighboring data sets d_1 and d_2, and K in Range(R). Then, the following equation holds:

$$\Pr[R(d_1) = K] \leqslant e^{\epsilon} \Pr[R(d_2) = K] \tag{1}$$

Note that if epsilon can get smaller, implying more stringent privacy. This randomized-based algorithm ensures differential privacy during the analytic process.

Definition 2 (Sensitivity): Let f be a function $f : d \to \mathbb{R}^d$ in the input space of the dataset. The sensitivity of f represents two neighboring datasets d_1 and d_2, given as follows:

$$\Delta f = \max_{d_1, d_2} ||f(d_1) - f(d_2)||_1 \tag{2}$$

In this equation, the maximum is over the pairs d_1 and d_2 in $\mathbb{R}^d$ differing in one element (at most) and $\|\cdot\|_1$ symbolize the ℓ_1 norm.

Definition 3 (The Laplace Mechanism): The Laplace mechanism adds noise from Laplace distribution by using the probability density function. Where, noise$(y) \propto \exp(-|y|/\lambda)$, with a mean equal to zero and standard deviation equal to $\sqrt{2}\lambda$. In our work, we use the Laplace mechanism which is given as follows:

$$L_P(x, f(.), \epsilon) = f(x) + (Q_1, \ldots Q_n) \tag{3}$$

where, we have a function $f : \mathbb{N}^{|x|} \to \mathbb{R}^k$. Laplace's mechanism relies on adding measured noises to the ciphertext that we want to compute.

G. SYSTEM MODEL

The considered system model consists of data providers, proxy servers and data analysts, as shown in Figure 1. The communication between these components is explained as follows:

1) Data providers $u_i \in \{u_1, u_2, u_3, \ldots, u_i\}$ provide the system with data from different sources. u_i needs to upload their sensitive data set d_i after encrypting it to the proxy server and delegate the encrypted sensitive data set $[d_i]$ to the analyst.
2) The proxy cloud server ($\mathcal{P}_S$) is responsible for redirecting the encrypted data set by the users to the analyst. $\mathcal{P}_S$ is a semi-honest cloud with high computational power.
3) Analyst $\mathcal{D}_A$ receives the encrypted data set and trains the machine learning model on it. The analyst can perform training on these ciphertexts without compromising the privacy of the data providers.

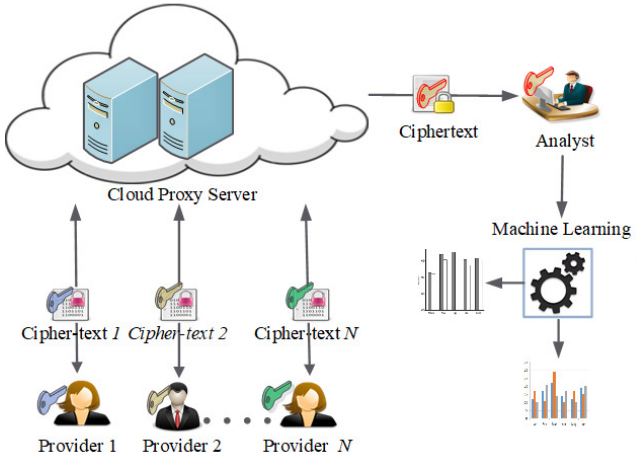

FIGURE 1. System Model of our proposed outsourced privacy-preserving machine learning scheme.

H. SECURITY MODEL

Suppose that the data providers u_i, $\mathcal{P}_S$, and $\mathcal{D}_A$ are semi-honest but untrusted. Furthermore, it is assumed that there is no collusion between the parties participating in the system model. Algorithm $\mathcal{B}$ answers the adversary queries according to our scheme. Adversary $\mathcal{A}$ has the following capabilities for attacking the plaintext of the users:

1) $\mathcal{A}$ can collude with u_i to obtain the plaintexts of all encrypted data downloaded from the cloud.
2) $\mathcal{A}$ can attack $\mathcal{P}_S$ to estimate the plaintexts of all ciphertext outsourced to the $\mathcal{P}_S$ by the users u_i and all data sent from $\mathcal{D}_A$.
3) $\mathcal{A}$ may corrupt some data from u_i to produce the plaintext from other users' ciphertexts.

IV. THE PROPOSED FRAMEWORK

The following subsections explain the proposed framework structure which contains an efficient privacy-preserving machine learning scheme with public verifiability.

A. OVERVIEW OF THE PROPOSED SCHEME

In the proposed framework, the data providers encrypt their data sets using the UPRE scheme and uploaded it to the proxy server cloud. In this step, the data providers could asset their data sets and decide which data are sensitive to be encrypted. Then, the cloud re-encrypt the ciphertexts received from the data providers under the public key of the analyst.

Accordingly, the server will add encrypted noised to the data provider ciphertexts using the partially homomorphic encryption of the Hashed-ElGamal scheme. After adding the encrypted noise, the cloud forwards the noise-ciphertext to the data analyst. In this step, the data analyst will decrypt the noise ciphertext to get the noise dataset. Then, the analyst performs machine learning algorithms for his predictive analytics. For instance, the analyst could use a k-nearest neighbor classifier, support vector machine classifier, naive Bayes classifier, and so on.

The data analyst will decrypt first the noise ciphertext to get the noise dataset using his private key. Then, the data analyst chooses and performs the mentioned-above classifiers on the noised data set using ϵ-DP without revealing the privacy of the individual users.

In our proposed framework, all the components of the model can publicly check the correctness of ciphertexts before performing any operation which ensures public verifiability. This part explains a significant contribution to this work because using this feature can reduce the waste time on invalid ciphertexts.

B. STRUCTURE OF THE PROPOSED SCHEME

Our protocol consists of the following phases:

- **Initialization Phase:**
  Here, the data providers prepare their public and private keys. Then, they compute the re-encryption key to use for redirecting their ciphertext to the data analyst. To do that, the data providers need to prepare the public parameters. p and q are selected as primes where $q|p-1$ and the bit-length of q is the security parameter k. The g uses as a generator of group $\mathbb{G}$, which is a subgroup of $\mathbb{Z}_q^*$ with order q. Four hash functions $H_1 : \{0,1\}^{e_0} \times \{0,1\}^{e_1} \rightarrow \mathbb{Z}_q^*$, $H_2 : \mathbb{G} \rightarrow \{0,1\}^{e_0+e_1}$, $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ and $H_4 : \mathbb{G} \rightarrow \mathbb{Z}_q^*$ are chosen. The public parameters are $(q, \mathbb{G}, g, H_1, H_2, H_3, H_4, e_0, e_1)$. The following algorithms represent this phase:
  1) *KeyGen*(): Pick $sk_{u_i} = (s_{u_i,1} \leftarrow \mathbb{Z}_q^*, s_{u_i,2} \leftarrow \mathbb{Z}_q^*)$ and set $pk_{u_i} = (pk_{u_i,1}, pk_{u_i,2}) = (g^{s_{u_i,1}}, g^{s_{u_i,2}})$.
  2) *ReKeyGen*(sk_{u_i}, pk_a): As input, with the private key of the users $sk_{u_i} = (sk_{u_i,1}, sk_{u_i,2})$ and the public key of the analyst $pk_a = (pk_{a,1}, pk_{a,2})$, this algorithm generates the re-encryption key $rk_{u_i \rightarrow a}$ as follows:
     a) Select $h \leftarrow \{0,1\}^{e_0}$, $\pi \leftarrow \{0,1\}^{e_1}$ and compute $\upsilon = H_1(h, \pi)$
     b) Compute $V = pk_{a,2}^{\upsilon}$ and $W = H_2(g^{\upsilon}) \oplus (h||\pi)$.
     c) Define $rk_{u_i \rightarrow a} = \frac{h}{s_{u_i,1} H_4(pk_{b,2}) + s_{u_i,2}}$.
     d) Return $(rk_{u_i \rightarrow a}^{\langle 1 \rangle}, V, W)$
- **Upload phase:**
  In this phase, the data providers will upload their encrypted data to cloud. The data provider's data set is represented by $d_i = \{(x^i, y^i) \subset X, Y\}$, where $x^i \in \mathbb{R}$ denoted the data vector and $y^i \in Y := \{0,1\}$ denotes the associated binary label. First, the data providers encrypt their sensitive data x^i using the $Enc(pk_{u_i}, x^i)$ algorithm with their public key $pk_{u_i} = (pk_{u_i,1}, pk_{u_i,2})$. Algorithm 1 details how the users encrypt $x^i \in \mathcal{M}$.
  The final result of the $Enc(pk_{u_i}, x^i)$ algorithm can also be represented as tuple of $[x^i] = (E_i, F_i)$. Therefore, the data set can be written as $Enc(pk_a, d_i) = [d_i] = ([x^i], [y^i]) = ((E_i, F_i, \varphi, \mu_i), [y^i])$, where φ, μ_i used as signature to confirm the ciphertext correctness. Second, the data providers determine the sensitivity level of their query function Δf_i and the privacy level ϵ_i for d_i. Finally, they send the encrypted data set $[d_i]$, Δf_i, ϵ_i and the re-encryption key $rk_{a \rightarrow b}$ to $\mathcal{P}_S$.
- **Download Phase:**
  This phase illustrates how the data providers can download their ciphertext from the cloud. The data providers use the $Dec(sk_i, \mathcal{C}_i)$ algorithm to download their outsourcing encrypted data from $\mathcal{P}_S$. The $Dec(sk_{u_i}, \mathcal{C}_i)$ algorithm takes $sk_{u_i} = (sk_{i,1}, sk_{i,2})$ and ciphertext $\mathcal{C}_i$ as input to get back the corresponding data set. Data providers can verify the correctness of $\mathcal{C}_i$ using (φ, μ_i) to receive the valid ciphertext. This step was used for the publicly verifiable experiments. Algorithm 2 describes how the data providers can decrypt their encrypted data and apply publicly verifiable data.

Algorithm 1 Encryption Algorithm

Input: leftmargin=4mm
- $pk_a = (pk_{u_i,1}, pk_{u_i,2})$: user's public key
- x^i: the message

1: Pick $\beta \leftarrow \mathbb{Z}_q^*$ and compute $\varphi = (pk_{u_i,1}^{H_4(pk_{u_i,2})} pk_{u_i,2})^{\beta}$.
2: Pick $w \leftarrow \mathbb{Z}_q^*$ and compute $r_i = H_1(x^i, w)$.
3: Compute $E_i = (pk_{u_i,1}^{H_4(pk_{u_i,2})} pk_{u_i,2})^{r_i}$ and $F_i = H_2(g^{r_i}) \oplus (x^i||w)$.
4: Compute $\mu_i = \beta + r_i.H_3(\varphi, E_i, F_i) mod q$.
5: Output the ciphertext $Enc(pk_{u_i}, x^i) = [x_i] = (E_i, F_i, \varphi, \mu_i)$

Output: $[x_i] = (E_i, F_i, \varphi, \mu_i)$: the ciphertext

Algorithm 2 User Decryption Algorithm

Input: leftmargin=4mm
- $sk_i = (sk_{u_i,1}, sk_{u_i,2})$: private key
- $\mathcal{C}_i = (E_i, F_i, \varphi, \mu_i)$: the ciphertext

if $(pk_{u_i,1}^{H_4(pk_{u_i,2})} pk_{u_i,2})^{\mu_i} = \varphi.E_i^{H_3(\varphi, E_i, F_i, \mu_i)}$ **then**
  compute $(x^i||\pi) = F_i \oplus H_2(E_i^{\frac{1}{s_{u_i,1} H_4(pk_{u_i,2}) + s_{u_i,2}}})$
  if $E = (pk_{u_i,1}^{H_4(pk_{u_i,2})} pk_{u_i,2})^{H_1(x^i, w)}$ **then**
    return $User\ Dec(sk_{u_i}, \mathcal{C}_i) = x^i$
  else
    return $\perp$.
  end if
else
  return $\perp$.
end if

Output: $x^i \in \mathcal{M}$: the message

- **Re-encryption phase:**
  Here, the cloud uses Re-encryption algorithm and transfers the generated ciphertext to the Analytic. The cloud receives $[d_i]$, Δf_i and ϵ_i from the data providers. Algorithm 3 details how $\mathcal{P}_S$ re-encrypt $\mathcal{C}_i$ with multiple public keys using the public key of the analyst. In this part, $\mathcal{P}_S$ checks the correctness of the ciphertext

Algorithm 3 Re-Encryption Algorithm

Input: leftmargin=4mm

- $rk_{u_i \to a} = (rk^{\langle 1 \rangle}_{u_i \to a}, V, W)$: Re-encryption key.
- $pk_{u_i} = (pk_{u_i,1}, pk_{u_i,2})$: Public key of the users.
- $pk_a = (pk_{a,1}, pk_{a,2})$: Public key of the analyst.
- $[x^i]_a = (E_i, F_i, \varphi, \mu_i)$: ciphertext under user i.

1: **if** $(pk_{u_i,1}^{H_4(pk_{u_i,2})} pk_{u_i,2})^{\mu_i} = \varphi . E_i^{H_3(\varphi, E_i, F_i)}$ **then**
2: compute $\gamma_i = E_i^{rk^{\langle 1 \rangle}_{u_i \to a}}$
3: Return $ReEnc(rk_{u_i \to a}, \mathcal{C}_i, pk_{u_i}, pk_a) = [x^i]_a(\gamma_i, F_i, V, W)$
4: **else**
5: Return $\perp$
6: **end if**

Output: $[x^i]_a = (\gamma_i, F_i, V, W)$: the ciphertext

using φ, μ_i. $\mathcal{P}_S$ uses the $ReEnc(rk_{u_i \to a}, \mathcal{C}_i, pk_{u_i}, pk_a)$ algorithm. Second, this algorithm uses $rk_{u_i \to a} = (rk^{\langle 1 \rangle}_{a \to b}, V, W)$ and (E_i, F_i) under the public key of the data providers $pk_{u_i} = (pk_{u_i,1}, pk_{u_i,2})$ as inputs. Then, $ReEnc(rk_{u_i \to a}, \mathcal{C}_i, pk_{u_i}, pk_a)$ returns a new ciphertext $[x^i]_a$ under the public key of the analyst $pk_a = (pk_{a,1}, pk_{a,2})$. Which means that the inputs of $\mathcal{C}_i$ with multiple public keys transfer to another ciphertext with the public key of analyst $pk_a = (pk_{a,1}, pk_{a,2})$.
Third, the cloud uses the Laplace distribution to generate a d-dimensional noise vector δ with parameter $\frac{\Delta f_i}{\epsilon_i}$. Here, the cloud computes $[x^i] \oplus [\delta^i]$. Then, the cloud system encrypts these noise vectors, transforming them into ciphertexts $[\delta]$. In this step, the cloud uses partial homomorphic hashed- ElGamal encryption [43]. Finally, the cloud sends $[d_i]'$ to the data analyst.

- **Learning phase**
Here, the analyst obtains the noisy ciphertexts from the cloud. Algorithm 4 describes this phase in detail. First, $\mathcal{D}_A$ checks the validation of $[d_i]'$ using V, W. Second, the analyst decrypts E_i' with the associated public keys of the data providers. Finally, the data analyst can obtain a noisy data set and apply his machine learning classification algorithms.

V. RESULT AND DISCUSSION

In this section, the implementation of the proposed scheme is presented with different perspectives in terms of results, performances, and discussions. The following subsections describe the implementation setup, the experimental results, and the discussions.

A. EXPERIMENT SETUP

To explain the theoretical analysis of computational complexity of our scheme, We denote by T_{exp} the computational cost of exponentiation in $\mathbb{G}$. Then, we have the Encryption time is 3 T_{exp}, re-Encryption time is 2.5 T_{exp}, the user Decryption time is 3.5 T_{exp}, and analyst decryption time is 4 T_{exp}.

Algorithm 4 Analyst Decryption Algorithm

Input: leftmargin=4mm

- $\mathcal{C}_a = (\gamma_i, F_i, V, W)$: public key

1: Compute $(h||\pi) = W \oplus H_2(V^{\frac{1}{s_{u_i,2}}})$
2: Compute $(x^i||w) = F \oplus H_2(\gamma_i^{\frac{1}{h}})$
3: **if** $V = pk_{u_i,2}^{H_2(h,\pi)}$ **then**
4: **if** $\gamma_i = g^{H_1(x^i,w).h}$ **then**
5: Return *Analyst Dec*$(sk_{a_i}, \mathcal{C}_a) = x^i$
6: **else**
7: Return $\perp$
8: **end if**
9: **end if**

Output: $x^i \in \mathcal{M}$: the message

TABLE 1. Security levels (Bits).

Security Level	Size of p	Size of q
80-bit	1024	160
112-bit	2048	224
128-bit	3072	256

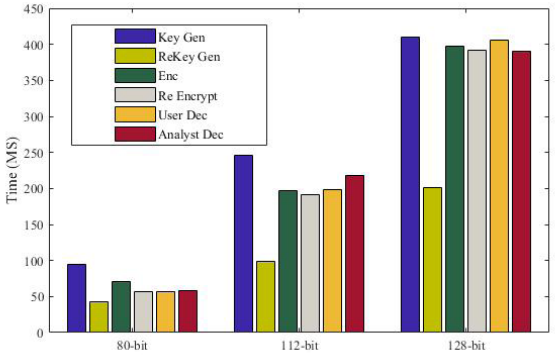

FIGURE 2. Computational costs.

To evaluate the computational costs of the proposed scheme, the experiment is conducted using Java pairing-based cryptography (JPBC) Library [44]. In this work, we employ a personal computer with CPU Intel Core i 7-3537U dual core (2.00 and 2.50) GHz and RAM 12 GB. Additionally, we use the curve $y^2 = x^3 + x$ over the field $\mathbb{F}_q$ to obtain type A pairings for $q = 3 \bmod 4$. To obtain security level, the experimental employ 80-bit, 112-bit, and 128-bit AES key size security level as shown in Table 1.

B. COMPUTATIONAL COST

This part illustrates the computational cost of our proposed framework. The processing costs of the proposed scheme are given according to the computational time. Figure 2 shows the computational times of our protocol phases using different security levels 80-bit, 112-bit, and 128-bit, respectively.

C. CIPHERTEXTS SIZE

The proposed scheme computes the ciphertexts with different security levels. For instance, we use the security level

TABLE 2. Comparative study with relevant schemes.

	No Pre-sharing	Non-Interactive	Homomorphic encryption	Differential privacy	Multiple data providers	Unidirectionaly	Collusion-resistance	Public verifiability
Our Scheme	✓	✓	AHE	✓	✓	✓	✓	✓
Li *et al.* [23]	✗	✗	AHE	✓	✓	✗	✓	✗
Gao *et al.* [35]	✗	✓	AHE	✗	✓	✗	✓	✗
Li *et al.* [45]	✗	✗	FHE	✗	✓	✗	✓	✗
Javier *et al.* [46]	✗	✗	AHE	✗	✓	✗	✓	✗

with 80-bit. Then, the elliptic curve with $q = 160/8$ bytes is employed. The size of $\mathbb{G}_1$ is given as 1024 bits. The size of $\mathbb{G}_1$ can be used with 65 bytes according to the related work [47]. As a result, the size of data providers' ciphertexts uploaded to the cloud is $3|\mathbb{G}| + |\mathbb{Z}_q| = 3 \times 65 + 160/8 = 215$ bytes. The re-encrypted ciphertexts size will be transformed to the analyst is $2|\mathbb{G}| + 2|\mathbb{Z}_q| = 2 \times 65 + 2 \times 160/8 = 170$ bytes.

D. PERFORMANCE OF THE ϵ-DP

The proposed scheme transforms the encrypted data with multiple public keys into noise-ciphertext. Accordingly, the proposed scheme improves the efficiency and accuracy of data processing.

The analyst performs the chosen machine learning algorithm on noise- data set with ϵ-DP and without revealing any information about the users. By using partial homomorphic hashed- ElGamal encryption scheme, the evaluation of the performances show excellent results in terms of accuracy and efficiency according to several related works [23], [43]. The analyst can use different machine learning such as naive Bayes classifier, support vector machine classifier, and a k-nearest neighbor classifier, etc.

E. PUBLIC VERIFIABILITY

The public verifiability is an important property, enabling a third party system to verify the integrity of the ciphertext stored in the cloud on behalf of data providers. Hence, the goal of our work is to guarantee data integrity with public verifiability and availability.

In the proposed protocol, all the components of the proposed model can publicly check the correctness or the validity of all the ciphertexts in public before doing any operation which is called public verifiability. Using this feature we can reduce the time that consuming for working on the invalid ciphertexts.

F. COMPARATIVE STUDY

In this part, we provide a comparative study between our proposed scheme and some relevant state-of-art schemes [23], [35], [45], [46]. Table 2 describes several points in our comparison including the variations and contributions of these schemes. The first column represents the state-of-art schemes [23], [35], [45], [46]. The second column shows whether a scheme needs to pre-sharing information or not to perform the computations between the data providers and the cloud. The third column illustrates whether all the actors in the systems should be online or not during the operations. The fourth column presents the type of homomorphic encryption used for each scheme. In this column, we denote "Fully Homomorphic Encryption" by "FHE" and "Additive Homomorphic Encryption" by "AHE". The fifth column illustrates the use of the differential privacy technique or not for each scheme. The sixth column compares the ability to work with multiple data providers or not. The seventh column explains whether the delegator in these schemes can re-direct the ciphertext to another actor in the system or not. The eighth column shows the resistance of the schemes against the collusion attack. Finally, the last line explains whether all the components in the system model of these schemes [23], [35], [45], [46] can check the correctness of the data providers' ciphertexts publicly or not.

VI. SECURITY ANALYSIS

This section introduces the security analysis with the ciphertext scenarios and the adversary relevant information. In detail, We adopt two security definitions: Adversary attacks the original ciphertext. Adversary attacks the re-encrypted ciphertext.

Theorem: The UPRE scheme is indistinguishability against chosen-ciphertext attacks (IND-PRE-CCA) secure in the random oracle model, where the CDH assumption is hard to solve in a group $\mathbb{G}$.

Assume that there is an adversary $\mathcal{A}$ to attack the IND-PRE-CCA security of the proposed scheme. Then, there exists an algorithm $\mathcal{B}$ to solve the CDH problem.

Here, we consider two types of adversaries: the first type of adversary attacks the original ciphertext of the users (encrypted data uploaded to the cloud). We denote this type by $\mathcal{A}_{or}$. The second type of adversary attacks the transformed ciphertext from the proxy cloud to the data analyst. We denote this type of adversary by $\mathcal{A}_{tr}$. Therefore, algorithm $\mathcal{B}$ answers the adversary queries.

Setup: Assume that $\mathcal{B}$ submits the public parameters $(q, \mathbb{G}, g, H_1, H_2, H_3, H_4, e_0, e_1)$ to $\mathcal{A}$. $\mathcal{B}$ simulates the random oracles of H_1, H_2 and H_3 with lists $\{L_{H_1}, L_{H_2}, L_{H_3}\}$, respectively, to avoid collision and ensure consistency. $\mathcal{B}$ also prepares two lists L_K for the public and private key and L_R for the re-encryption key. $\mathcal{B}$ starts generating the original keys and corrupted keys.

Lemma 1: Considering $\mathcal{A}_{or}$ for communicating with $\mathcal{B}$ according to the IND-PRE-CAA game

Phase 1: $\mathcal{A}_{or}$ issues a series of queries, and $\mathcal{B}$ answers $\mathcal{A}_{or}$ as in the proposed scheme.

Challenge: $\mathcal{A}_{or}$ challenges $\mathcal{B}$ and returns $(pk_{u_i^*,1}, pk_{u_i^*,2})$ as well as two messages of the same length $m_0, m_1 \in \{0, 1\}^{e_0}$. Then, $\mathcal{B}$ responds to $\mathcal{A}_{or}$ with the challenge ciphertext $\mathcal{C}^* = (E^*, F^*, \varphi^*, \mu^*)$ contains the instance element

of the DCDH problem in $H_1(m_\varsigma, w^*) = ab$ and $H_2(g^{ab}) = (m_\varsigma||w^*) \bigoplus F^*$ for $\varsigma \in \{0, 1\}$.

Phase 2: $\mathcal{A}$ continues to issue queries as in **Phase 1** with the restrictions described in the **IND-PRE-CCA game**. Algorithm $\mathcal{B}$ responds to these queries for $\mathcal{A}$ as in **Phase 1**.

Guess: Eventually, $\mathcal{A}_{or}$ responds with a guess $\varsigma^{'}$ and sends it to $\mathcal{B}$. Finally, $\mathcal{B}$ returns the solution of the DCDH instance.

Lemma 2: Considering $\mathcal{A}_{tr}$ communicating with $\mathcal{B}$ according to the IND-PRE-CAA game.

Phase 1: $\mathcal{A}_{tr}$ issues a series of queries, and $\mathcal{B}$ answers $\mathcal{A}_{tr}$ as in the proposed scheme.

Challenge: $\mathcal{A}_{tr}$ challenges $\mathcal{B}$ and returns $(pk_{u_i^*,1}, pk_{u_i^*,2})$ as well as two messages of the same length m_0, $m_1 \in \{0, 1\}^{e_0}$. Then, $\mathcal{B}$ responds to $\mathcal{A}_{tr}$ with the challenge ciphertext $\mathcal{C}^* = (E^*, F^*, V^*, W^*)$ contains the instance element of the DCDH problem in $H_1(m_\varsigma, w^*) = r^* = (b/a)(t/rk_{i' \to i^*}(x_{i^*,1}H_4(pk_{i',2}) + x_{i',2}))$ and $H_2(g^{a/b})^{t/rk_{i' \to i^*}(x_{i^*,1}H_4(pk_{i',2})+x_{i',2})} \bigoplus (m_\varsigma||w^*) = F^*$ for $\varsigma \in \{0, 1\}$.

Phase 2: $\mathcal{A}_{tr}$ continues to issue queries as in **Phase 1** with the restrictions described in the **IND-PRE-CCA game**. Algorithm $\mathcal{B}$ responds to these queries for $\mathcal{A}_{tr}$ as in **Phase 1**.

Guess: $\mathcal{A}_{or}$ responds with a guess $\varsigma^{'}$ and sends it to $\mathcal{B}$. Finally, $\mathcal{B}$ returns the solution of the DCDH instance. The UPRE scheme is IND-PRE-CCA secure in the random oracle model.

If A has corrupted $\mathcal{D}_A$ or $\mathcal{P}_S$ to get the outsourced data, then, $\mathcal{A}$ cannot get the plaintext due to the IND-PRE-CCA of our scheme. Additionally, if $\mathcal{A}$ obtains access to some data providers and if the re-encryption key cannot provide access to the plaintext of the data providers, our scheme achieves ϵ-DP. Our scheme is secure under the DCDH in the random oracle model.

VII. CONCLUSION

Cloud computing security is still considered as a major issue, especially with privacy-preserving of the data providers to the third party systems. This paper presents an efficient privacy-preserving machine learning scheme for multi-providers with the collaboration of a third party system. In this regard, the proposed protocol employed a unidirectional proxy re-encryption protocol to protect cloud data sets. All parties can publicly verify the encrypted data sets, which reduces the computational cost and network overhead. The proposed scheme is secure under the CDH assumption in the random oracle model. The proxy server adds noise to the ciphertext using ϵ-DP, rather than the data providers, to facilitate data analytics tasks. Our proposed protocol guarantees a secure multi-party computation and privacy-preserving classification based on partial homomorphic encryption.

For future works, we plan to study the parallel algorithm for secure multiparty computation on Blockchain techniques. We also aim to investigate the using of more complex computations such as partial differential equations over encrypted data.

REFERENCES

[1] Q. Liu, Y. Guo, J. Wu, and G. Wang, "Effective query grouping strategy in clouds," *J. Comput. Sci. Technol.*, vol. 32, no. 6, pp. 1231–1249, Nov. 2017.

[2] K. Popović and Ž. Hocenski, "Cloud computing security issues and challenges," in *Proc. 33rd Int. Conv. (MIPRO)*, 2010, pp. 344–349.

[3] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *Proc. Eur. Symp. Res. Comput. Secur.* Berlin, Germany: Springer, 2009, pp. 355–370.

[4] Y. Li, S. Yao, K. Yang, Y. Tan, and Q. Zhang, "A high-imperceptibility and histogram-shifting data hiding scheme for JPEG images," *IEEE Access*, vol. 7, pp. 73573–73582, 2019.

[5] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An id-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, vol. 6, pp. 20632–20640, 2018.

[6] A. D. Josep, R. Katz, A. Konwinski, L. Gunho, D. Patterson, and A. Rabkin, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[7] J. Xu, L. W. Wei, Y. Zhang, A. D. Wang, F. C. Zhou, and C.-Z. Gao, "Dynamic fully homomorphic encryption-based merkle tree for lightweight streaming authenticated data structures," *J. Netw. Comput. Appl.*, vol. 107, pp. 113–124, Apr. 2018.

[8] Z. Yu, C.-Z. Gao, Z. Jing, B. B. Gupta, and Q. Cai, "A practical public key encryption scheme based on learning parity with noise," *IEEE Access*, vol. 6, pp. 31918–31923, 2018.

[9] Q. Zhang, Y. Li, Q. Zhang, J. Yuan, R. Wang, Y. Gan, and Y. Tan, "A self-certified cross-cluster asymmetric group key agreement for wireless sensor networks," *Chin. J. Electron.*, vol. 28, no. 2, pp. 280–287, 2019.

[10] S. Halevi, "Homomorphic encryption," in *Tutorials on the Foundations of Cryptography*. Cham, Switzerland: Springer, 2017, pp. 219–276.

[11] C. Gentry and D. Boneh, *A Fully Homomorphic Encryption Scheme*, vol. 20, no. 9. Stanford, CA, USA: Stanford Univ. Stanford, 2009.

[12] D. Boneh, R. Gennaro, S. Goldfeder, A. Jain, S. Kim, P. M. R. Rasmussen, and A. Sahai, "Threshold cryptosystems from threshold fully homomorphic encryption," in *Proc. Annu. Int. Cryptol. Conf.* Cham, Switzerland: Springer, 2018, pp. 565–596.

[13] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1999, pp. 223–238.

[14] W. Ding, Z. Yan, and R. H. Deng, "Encrypted data processing with homomorphic re-encryption," *Inf. Sci.*, vol. 409, pp. 35–55, Oct. 2017.

[15] Y. Ishai and A. Paskin, "Evaluating branching programs on encrypted data," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2007, pp. 575–594.

[16] J. H. Cheon and J. Kim, "A hybrid scheme of public-key encryption and somewhat homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 5, pp. 1052–1063, May 2015.

[17] C. Dwork, "Differential privacy," in *Encyclopedia Cryptography Security*. Boston, MA, USA: Springer, 2011, pp. 338–340.

[18] Y. Xue, Y.-A. Tan, C. Liang, Y. Li, J. Zheng, and Q. Zhang, "RootAgency: A digital signature-based root privilege management agency for cloud terminal devices," *Inf. Sci.*, vol. 444, pp. 36–50, May 2018.

[19] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 201–210.

[20] Y. Tan, Y. Xue, C. Liang, J. Zheng, Q. Zhang, J. Zheng, and Y. Li, "A root privilege management scheme with revocable authorization for Android devices," *J. Netw. Comput. Appl.*, vol. 107, no. 4, pp. 69–82, Apr. 2018.

[21] R. Hamza, Z. Yan, K. Muhammad, P. Bellavista, and F. Titouna, "A privacy-preserving cryptosystem for IoT e-healthcare," *Inf. Sci.*, to be published.

[22] A. Hassan, N. Eltayieb, R. Elhabob, and F. Li, "An efficient certificateless user authentication and key exchange protocol for client-server environment," *J. Ambient Intell. Humanized Comput.*, vol. 9, no. 6, pp. 1713–1727, 2018.

[23] P. Li, T. Li, H. Ye, J. Li, X. Chen, and Y. Xiang, "Privacy-preserving machine learning with multiple data providers," *Future Gener. Comput. Syst.*, vol. 87, pp. 341–350, Oct. 2018.

[24] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed GraphLab: A framework for machine learning and data mining in the cloud," *Proc. VLDB Endowment*, vol. 5, no. 8, pp. 716–727, 2012.

[25] Q. Liu, G. Wang, X. Liu, T. Peng, and J. Wu, "Achieving reliable and secure services in cloud computing environments," *Comput., Elect. Eng.*, vol. 59, pp. 153–164, Apr. 2017.
[26] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Proc. IMA Int. Conf. Cryptogr. Coding*. Berlin, Germany: Springer, 2013, pp. 45–64.
[27] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: Deep neural networks over encrypted data," 2017, *arXiv:1711.05189*. [Online]. Available: https://arxiv.org/abs/1711.05189
[28] P. Li and J. Li, "Multi-key privacy-preserving deep learning in cloud computing," *Future Generat. Comput. Syst.*, vol. 74, pp. 76–85, Sep. 2017.
[29] T. Chen and S. Zhong, "Privacy-preserving backpropagation neural network learning," *IEEE Trans. Neural Netw.*, vol. 20, no. 10, pp. 1554–1564, Oct. 2009.
[30] A. Bansal, T. Chen, and S. Zhong, "Privacy preserving back-propagation neural network learning over arbitrarily partitioned data," *Neural Comput. Appl.*, vol. 20, no. 1, pp. 143–150, 2011.
[31] S. Samet and A. Miri, "Privacy-preserving back-propagation and extreme learning machine algorithms," *Data Knowl. Eng.*, vols. 79–80, pp. 40–61, Sep./Oct. 2012.
[32] T. Graepel, K. Lauter, and M. Naehrig, "Ml confidential: Machine learning on encrypted data," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Berlin, Germany: Springer, 2012, pp. 1–21.
[33] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 5, pp. 1417–1429, May 2017.
[34] Z. Wei, J. Li, X. Wang, and C.-Z. Gao, "A lightweight privacy-preserving protocol for vanets based on secure outsourcing computing," *IEEE Access*, vol. 7, pp. 62785–62793, 2019.
[35] C.-Z. Gao, Q. Cheng, X. Li, and S.-B. Xia, "Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network," *Cluster Comput.*, vol. 22, no. 1, pp. 1655–1663, 2018.
[36] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational differential privacy," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2009, pp. 126–142.
[37] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.
[38] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *Proc. ACM Workshop Privacy Electron. Soc.*, 2012, pp. 81–90.
[39] B. Chevallier-Mames, P. Paillier, and D. Pointcheval, "Encoding-free elgamal encryption without random oracles," in *Proc. Int. Workshop Public Key Cryptography*. Berlin, Germany: Springer, 2006, pp. 91–104.
[40] F. Bao, R. H. Deng, and H. Zhu, "Variations of Diffie-Hellman problem," in *Proc. Int. Conf. Inf. Commun. Secur.* Berlin, Germany: Springer, 2003, pp. 301–312.
[41] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proc. 14th ACM Conf. Comput. Commun. Secur.* 2007, pp. 185–194.
[42] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities," *IEEE Internet Things J.*, to be published.
[43] M. Joye and B. Libert, "Encoding-free ElGamal-type encryption schemes on elliptic curves," in *Proc. Cryptographers Track RSA Conf.* Springer, 2017, pp. 19–35.
[44] A. De Caro and V. Iovino, "JPBC: Java pairing based cryptography," in *Proc. 16th IEEE Symp. Comput. Commun. (ISCC)*, Corfu, Greece, Jun./Jul. 2011, pp. 850–855.
[45] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Comput.*, vol. 21, no. 1, pp. 277–286, Mar. 2018.
[46] F.-J. González-Serrano, A. Amor-Martín, and J. Casamayón-Antón, "Supervised machine learning using encrypted training data," *Int. J. Inf. Secur.*, vol. 17, no. 4, pp. 365–377, 2018.
[47] K.-A. Shim, Y.-R. Lee, and C.-M. Park, "EIBAS: An efficient identity-based broadcast authentication scheme in wireless sensor networks," *Ad Hoc Netw.*, vol. 11, no. 1, pp. 182–189, Jan. 2013.

ALZUBAIR HASSAN received the B.Sc. degree in computer science from the University of Kassala, in 2010, the M.Sc. degree in mathematical science from the University of Khartoum, in 2013, and the Ph.D. degree in computer science and technology from the University of Electronic Science and Technology of China, in 2018. He is currently a Postdoctoral Researcher with the School of Computer Science and Cyber Engineering, Guangzhou University. His current research interests include cryptography, network security, and privacy-preserving.

RAFIK HAMZA received the M.Sc. and Ph.D. degrees in computer science from the University of Batna 2, in 2014 and 2017, respectively. He was a Principal Engineer with R&D Sonatrach, Boumerdès, in 2018. He is currently a Researcher with Guangzhou University. He is involved in machine learning and cryptography techniques. He has published several articles between top international scientific journals and conferences. His research interests include machine learning, information security, access control, image and video processing, chaos theory, and lightweight cryptography applications. He is serving as a Reviewer for well-reputed international journals and conferences.

HONGYANG YAN received the M.S. degrees from the School of Mathematics and Information Science, Guangzhou University, in 2016. She is currently pursuing the Ph.D. degree with Nankai University. Her research interests include secure access control, such as attribute-based cryptography and identity-based cryptography, and the IoT secure.

PING LI received the M.S. and Ph.D. degrees in applied mathematics from Sun Yat-sen University, in 2011 and 2016, respectively. She held a postdoctoral position at Guangzhou University, from 2016 to 2018. She is currently a Researcher with the School of Computer Science, South China Normal University. Her current research interests include cryptography, privacy-preserving, and cloud computing.