A Robust and Lightweight Privacy-Preserving Data Aggregation Scheme for Smart Grid

Liqiang Wu, Shaojing Fu, Yuchuan Luo, Hongyang Yan, Heyuan Shi, and Ming Xu

***Abstract*—Privacy-preserving data aggregation (PPDA) enables data availability and privacy preservation simultaneously in smart grid. However, existing methods, such as masking and homomorphic encryption, cannot simultaneously offer strong privacy preservation, fault tolerance for both smart meters and aggregators, verifiable aggregation, and lightweight encryption. To tackle these challenges, we design HTV-PRE, a homomorphic threshold proxy re-encryption scheme with re-encryption verifiability. HTV-PRE involves only linear operations and resists quantum attacks after being instanced by ideal lattices. By leveraging HTV-PRE, we propose a robust and lightweight data aggregation scheme with strong privacy preservation for smart grid. Robustness ensures fault tolerance and error detection. Even if some smart meters or aggregators are faulty, data aggregation can still work without imposing expensive computation on other smart meters or requiring additional trust assumptions. Additionally, to detect aggregators' errors, a proof for the aggregated result is presented so that anyone can verify whether the result has been correctly computed or not. The verifiable aggregation adds no computation/communication overhead on the user side. The performance evaluations demonstrate that our PPDA scheme significantly offloads computation overhead from smart meters and control center to the edge, and its user encryption is up to 4x faster than existing approaches.**

***Index Terms*—Data aggregation, fault tolerance, privacy preservation, smart grid, verifiable aggregation.**

I. Introduction

As the energy industry integrating with emerging information technologies, such as cloud/edge computing, artificial intelligence, and Internet of Things, smart grid [1] makes the traditional power grid decarbonized, digitalized, and intelligent. The benefits provided by smart grid include 1) real-time load monitoring and balancing, 2) automatic troubleshooting and maintenance, 3) price optimization by demand-response mechanism, 4) enhanced security and privacy, and 5) increasing integration of large-scale renewable energy systems. Therefore, smart grid represents an unprecedented opportunity to promote economic boom and environmental health.

As a core component of smart grid, advanced meter infrastructure[1] enables high-speed and two-way communications between Smart Meter (SM) and Control Center (CC). SM periodically collects real-time power consumption data (such as 15 minutes) and forwards them to CC by a gateway or fog node. Real-time total power consumption is crucial to decision-making, such as dynamic scheduling, load balancing, and power failure reporting. Unfortunately, because fine-grained power consumptions are sensitive, improper handling of these sensitive data will seriously threaten user privacy. For example, disclosed fine-grained electricity data could reconstruct many intimate details of a user's daily life[2], including when householders leave home or come back, what electrical appliances are launched, and their duration time. Therefore, fine-grained user data should be protected from unauthorized access.

Privacy-Preserving Data Aggregation (PPDA) is the most promising approach to simultaneously enable data availability and privacy preservation in smart grid. Apart from protecting user privacy, an ideal and applausive PPDA must be *lightweight* and *robust*. Since smart meters are resource-constrained, data aggregation has to impose as little computation/communication overhead as possible on smart meters. In addition, many intractable problems are inevitable during data aggregation, such as some meters' breakdowns, a small number of aggregators' unavailability, and even forged or wrong computations. Despite this, a robust PPDA scheme has to work and finally provides correct aggregated results as expected.

A. Motivation

There are several methods to realize PPDA, as listed in Table I. (1) Differential Privacy (DP). SMs inject some noises into fine-grained readings before data aggregation[3], [4], [5]. DP is easy to achieve spatial aggregation and temporal aggregation. Besides, DP is fault-tolerant, where some SMs' breakdowns do not affect normal data aggregation. However, aggregated results are inaccurate due to noises. (2) Masking. Each SM masks its reading with a blinding factor. All the blinding factors disappear while the sum of readings is reserved after aggregation. The blinding factors are generated through Trusted Authority (TA)[6], [7], [8], or collaboration of some fixed SMs[9][10].

Manuscript received 2 December 2021; revised 25 December 2022; accepted 27 February 2023. Date of publication 6 March 2023; date of current version 12 January 2024. This work was supported in part by the National Nature Science Foundation of China under Grants 62072466, 61872372, 52177067, 62102429, 62102422, 62172436, and 62102452, in part by Graduate Research and Innovation projects in HuNan province under Grant CX20210008, in part by the Natural Science Foundation of Shaanxi Province, China under Grant 2021JM-252, and in part by the NUDT under Grants ZK19-38. *(Corresponding authors: Shaojing Fu and Yuchuan Luo.)*

Liqiang Wu, Shaojing Fu, Yuchuan Luo, and Ming Xu are with the College of Computer, National University of Defense Technology, Changsha 410073, China (e-mail: wuliqiang@nudt.edu.cn; fushaojing@nudt.edu.cn; luoyuchuan09@nudt.edu.cn; xuming@nudt.edu.cn).

Hongyang Yan is with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangdong, China, and also with the Department of Electronic and Information Engineering, the Hong Kong Polytechnic University, Hong Kong (e-mail: hyang_yan@163.com).

Heyuan Shi is with the Big Data Institute, Central South University, Changsha 410073, China (e-mail: hey.shi@foxmail.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TDSC.2023.3252593, provided by the authors.

Digital Object Identifier 10.1109/TDSC.2023.3252593

TABLE I
COMPARISON OF EXISTING PPDA METHODS

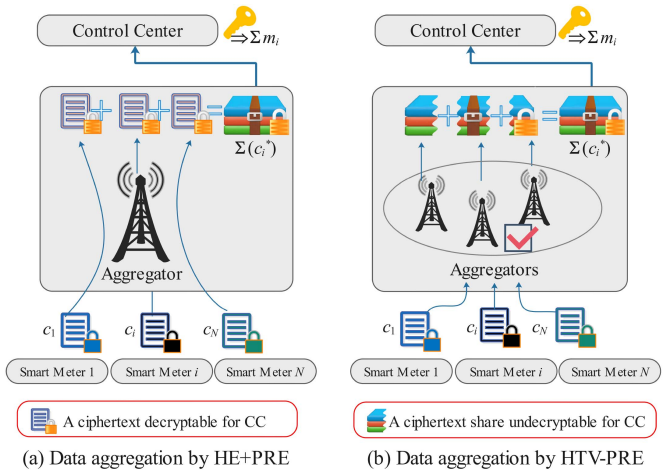
Methods	Lightweight			Robustness					
	F1	F2	F3	F4	F5	F6	F7	F8	F9
DP [3] [4]	√	√	√	×	√	√	×|√	√	×
Masking [9] [10]	√	×	○	√	√	○	×|○	√	×
HE [11] [12]	×	√	√	√	√	√	×|√	×	√
HE+Masking [17] [7]	×	○	○	√	√	○	×|○	√	×
HE+PRE [15] [16]	×	√	√	√	√	√	×|√	×	×
HTV-PRE (Ours)	√	√	√	√	√	√	√|√	√	√

(1) F1: Lightweight encryption. F2: Simple key management. F3: User dynamics. F4: Accurate aggregation. F5: Spatial aggregation. F6: Temporal aggregation. F7: Fault tolerance (a small number of aggregators' breakdowns | some SMs' breakdowns). F8: Strong privacy preservation. F9: Verifiable aggregation.
(2) √: The property is satisfied easily. ×: The property is unsatisfied. ○: The property can be achieved at the expense of relying on stronger assumptions or increasing computation/communication overhead on the user side.

Masking is efficient for SM's encryption, but complicated in blinding factor management because blinding factors must be random and fresh. (3) Homomorphic Encryption (HE). Additive HE supports data aggregation in an encrypted form. Lu et al. [11] encrypted multi-dimensional data by Paillier. Ni et al. [12] employed bilinear paring to achieve verifiable aggregation. Unfortunately, existing HE-based PPDA schemes usually introduce heavy computation burdens to SMs. Moreover, they encrypt all the user readings under CC's public key so that CC can access any individual meter reading [11], [13]. Therefore, HE often combines with other methods, such as HE+Masking and HE+PRE (Proxy Re-Encryption). HE+Masking turns key management efficient but loses fault tolerance. To be fault-tolerant, some HE+Masking schemes assume that TA is real-time online to deal with SM's dropouts[7], or some SMs cooperatively calculate an equivalent ciphertext[14]. HE+PRE encrypts meter readings under different public keys of SMs[15][16] but suffers collusion attack, where a single aggregator controlling the proxy key will arbitrarily transform SM's ciphertexts without their consent or even awareness.

In summary, existing PPDA methods can satisfy only part, but not all, of the favorable properties required in smart grid. Certain properties are fulfilled at the expense of relying on stronger assumptions or increasing computation/communication overhead on the user side. Worse still, it is still unable to design a lightweight and robust PPDA scheme by a simple combination of existing methods and techniques because different types of cryptographic primitives are unable to be well coordinated.

A natural question from the current PPDA area is *how to design a robust and lightweight PPDA in smart grid, in particular, supporting strong privacy preservation, resisting faults from both smart meters and aggregators, achieving verifiable aggregation, and enjoying lightweight encryption on the user side simultaneously.*



(a) Data aggregation by HE+PRE (b) Data aggregation by HTV-PRE

Fig. 1. Data aggregation by HE+PRE and HTV-PRE.

B. Technical Overview

Motivated by this question, we propose a robust and lightweight privacy-preserving data aggregation scheme for smart grid. The trick behind our PPDA scheme is the innovation of HTV-PRE, a homomorphic threshold proxy re-encryption with re-encryption verifiability. HTV means it simultaneously satisfies Homomorphism over ciphertext, Thresholding for aggregators, and Verifiability on re-encryption operations.

HTV-PRE is a new cryptographic mechanism dedicated to PPDA scenarios after observing the vulnerabilities and deficiencies of HE+PRE. The basic idea is demonstrated in Fig. 1. First, the failure of aggregators is fatal to normal aggregation but often neglected in previous schemes. Besides, a single aggregator with unsupervised power easily leads to collusion attacks. In contrast, our HTV-PRE allows multiple aggregators to work independently. Each aggregator generates a share of aggregated ciphertext undecryptable for CC. Therefore, HTV-PRE can still work even if a small number of aggregators are faulty.

Second, aggregators do not always aggregate ciphertexts honestly, such as discarding some user ciphertexts, or providing random values to save computing resources. To address these challenges, we require aggregators to present a proof, by which the correctness of re-encryption and aggregation can be verified publicly. Compared to existing work[12], [16], our HTV-PRE realizes verifiable aggregation without adding any computation/communication overhead on the user side.

Third, HTV-PRE selects a lightweight HE scheme over Ring-LWE [18] because of its simple linear operations and high parallelization. The Ring-LWE-based soft implementation has proved ideal for resource-constrained devices[19]. As far as we know, Ring-LWE is the only tool competent for our HTV-PRE that simultaneously supports thresholding and re-encryption without losing additive homomorphism on ciphertexts.

C. Contributions

Our contributions can be summarized as follows.

- We design a homomorphic threshold proxy re-encryption with re-encryption verifiability over Ring-LWE, namely

HTV-PRE. HTV-PRE allows each user to encrypt its message under its respective public key. It also decentralizes the proxy power by allowing multiple proxies to re-encrypt and aggregate ciphertexts independently. Meanwhile, each proxy provides a proof to certify its correctness. HTV-PRE is provably secure against chosen-plaintext attacks.

- By leveraging HTV-PRE, we propose a robust and lightweight PPDA scheme for smart grid. It is fault-tolerant for both malfunctioning smart meters and unavailable aggregators. Besides, it achieves verifiable aggregation while adding no computation or communication overhead on the user side.
- We implement the proposed PPDA and evaluate it in experiments. The result shows the majority of computation costs are significantly offloaded from SMs and CC into the edge. Our PPDA achieves a speedup for data encryption up to 4x faster than existing competing approaches at the same security level.

The rest of this paper is organized as follows. The related work is introduced in Section II. The network model, threat model, design goals, and related cryptographic primitives are given in Sections III and IV, respectively. Detailed design and security analysis of our mechanism is presented in Sections V and VI. Performance evaluation is performed in Section VII. The last section summarizes our work.

II. Related Work

To tackle privacy concerns of the fine-grained user data, many PPDA schemes have been proposed with good performance and rich properties.

(1) Fault-tolerant PPDA

Shi et al. [8] proposed a diverse grouping-based aggregation protocol with error detection. It aggregated user data in groups and dropped the group containing malfunctioning SMs. Bao and Lu [3] proposed a secure data aggregation scheme with differential privacy and fault tolerance. They added Laplace noises to the accurate reading before encryption. If a SM failed, an auxiliary ciphertext was used to decrypt the aggregated ciphertext. Xue et al. [9] constructed a masking-based PPDA scheme without TA. Each user selected its blinding factor by cooperating with multiple peering users randomly. If a SM failed, its private key was recovered by secret sharing. Knirsch et al. [20] proposed a masking-based PPDA. If SM_i failed, its blinding factor and masked value would be both excluded from aggregated results. Lyu et al. [21] applied masking to deal with noisy individual measurements. When a SM failed, the connected fog node's blind factor was updated as the negative sum of the remaining connected nodes to ensure correct decryption. Wang et al. [22] proposed a fault-tolerant and multi-subset data aggregation scheme. If a SM was faulty, its cooperators had to remove some parts associated with the malfunctioning SM from their blind factors. Subsequently, all users uploaded the ciphertexts again. Ni et al. [23] proposed DiPrism, which simultaneously supported differential privacy, fault tolerance, and range-based filtering. Ahsan et al. [6] proposed a fog-enabled scheme FESDA by Paillier, and Xue et al. [7] presented PPSO by making Paillier decryptable in two ways. Both schemes[6], [7] concealed the original data from CC by centralized management of blinding factors. To be fault-tolerant, those schemes [6], [7], [14], [17], [23] allowed aggregators or TA to construct an equivalent ciphertext for correct decryption. Dimitriou and Karame[24] introduced multiple aggregators to reduce the dependence on a single aggregator but brought heavy communication overhead.

(2) Verifiable PPDA

Considering different security requirements, we usually impose verification on data or computation.

(i) Verification on data. Lu et al. [11] proposed a multi-dimensional PPDA by a super-increasing sequence. Besides, it used a batch verification technique to reduce authentication cost. Fan et al. [25] first formatted the security requirement against internal attackers. He et al. [26] provided an improved PPDA scheme against internal attackers, which satisfied both integrity and privacy. After that, many PPDA schemes are proposed with batch verification on signatures. For example, Ding et al. [27] constructed an efficient identity-based PPDA supporting batch verification. Gope and Sikda [10] proposed a lightweight and privacy-friendly data aggregation scheme through masking. Liu et al. [28] proposed a lightweight and verifiable PPDA scheme by online/offline signature. Liu et al. [29] proposed privacy-preserving data aggregation and some function queries on encrypted data. Liu et al. [30] proposed an efficient and multi-dimensional PPDA without TA. Furthermore, those two schemes [29], [30] provided an approach to recognize invalid signatures if batch verification failed. EPIC[31] enabled the utility to verify the end-to-end integrity of aggregated results and identify the attackers in a privacy-preserving way.

(ii) Verification on computation. Emura et al. [32] achieved public verifiability on computation from a variant of the Computational Diffie-Hellman assumption. Ni et al. [33] proposed a security-enhanced PPDA through trapdoor hash functions and homomorphic authenticators. Dimitrioua and Awad [34] constructed a publicly verifiable data aggregation scheme against malicious smart meters, not aggregators. Ni et al. [12] and David et al. [16] respectively achieved verifiable aggregation against misbehaving aggregators. Their primary tools are proxy re-signature [12] and proxy re-authentication[16], respectively. Zhang et al. [35] proposed a multi-type PPDA scheme where CC could guarantee data integrity and aggregation correctness.

(3) PRE-based PPDA.

David et al. [16] first introduced homomorphic proxy re-encryption and re-authentication into multi-user data aggregation. In addition to input and output privacy, it achieved end-to-end authenticity and verifiable aggregation. Huang et al. [15] proposed a reliable and privacy-preserving selective data aggregation scheme for fog-enabled IoT systems based on HE+PRE. However, those two schemes[15], [16] suffered from collusion attacks and dishonest re-encryption. Ni et al. [12] proposed P ^{2} SM to re-encrypt the bills generated by the collectors so that the users could decrypt daily bills. Unfortunately, their scheme treated the operation center as a trusted entity. Therefore, P ^{2} SM failed to meet strong privacy preservation.

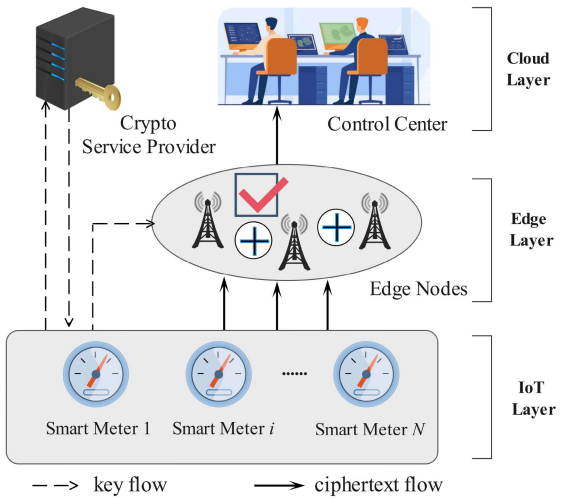


Fig. 2. System model.

Existing PRE-based PPDA schemes are constructed on traditional number-theoretic assumptions, such as discrete logarithms and bilinear pairings. They can be problematic if quantum computers are available. The LWE-based PRE scheme is a promising alternative due to its potential for post-quantum security and efficient linear operations. Our work explores those advantages.

III. Problem Formulation

A. System Model

Our system is based on a three-layer network framework from advanced meter infrastructure, as shown in Fig. 2. It mainly involves the following four entities.

- *Control Center (CC)*: CC is responsible for collecting, analyzing real-time power consumption data, and performing related operations.
- *Edge Node (EN)*: Edge nodes extend the computation and storage capabilities from the cloud to the edge of terminal devices. Multiple edge nodes work independently, and they belong to different edge service providers.
- *Smart Meter (SM)*: Each user is equipped with a smart meter. SM encrypts the collected power consumption in a certain period and then reports it to edge nodes.
- *Crypto Service Provider (CSP)*. CSP provides digital signature services for SMs. For a legitimate message, CSP generates the corresponding signature to SM. Anyone can verify its validity with CSP's verification key. CSP will be offline after system initialization.

B. Workflow

Our PPDA scheme works as follows.

Step 1. System initialization.

CC initializes the whole system and generates public parameters. Then CC and SMs generate their respective public/private key pairs. Every SM generates proxy key shares and sends them to CSP for signature. At last, SMs distribute proxy key shares and their signatures to ENs.

Step 2. Data reporting.

A SM encrypts the power consumption data under its public key, then forwards the ciphertext to ENs.

Step 3. Data aggregation and proof generation.

After collecting all the SM's ciphertexts, each edge node independently re-encrypts them from different SMs to CC. Simultaneously, the same transformation is performed on the corresponding signatures. Finally, it aggregates those transformed ciphertexts as ciphertext share and aggregates their signatures as a proof.

Step 4. Proof verification and data combination.

A public auditor is designated randomly among edge nodes to verify each edge node's ciphertext share by its proof. Then it combines all the shares into a whole ciphertext decryptable for CC if sufficient ciphertext shares are provided. All the work performed by the auditor is public, so anyone can check its correctness.

Step 5. Decryption.

CC decrypts the aggregated ciphertext and gets the total consumption.

C. Threat Model

In our work, CC is regarded as honest-but-curious, which follows the protocol, but is eager to disclose user privacy. To be closer to the smart grid scenario, we consider edge nodes as misbehaving aggregators. On the one hand, misbehaving edge nodes will not actively expose their proxy key shares, which are easily captured and punished severely. Moreover, we allow at most $\bar{k}-1$ edge nodes colluding with CC, where $\bar{k}$ is a predefined threshold. This assumption is reasonable because nodes run independently and belong to different edge service providers. On the other hand, misbehaving edge nodes are allowed to deviate from normal protocol execution, such as discarding some user ciphertexts, or providing random values to save time or computing resources. The CSP, fully controlled by a trusted authority or government, is completely trusted. SMs are physically protected and honest to provide actual power consumption data to edge nodes.

Our threat model is also considered by [12], [33], which are security-enhanced PPDA schemes by modeling aggregators as misbehaving attackers. By contrast, most of the existing PPDA regarded aggregators as honest-but-curious only. Besides, some attacks during data transmission, such as tampering, replying, and false data injection, can be prevented by authentication and signature techniques or already-made STL protocols. It is straightforward, so we omit those threats in our work for simplicity.

D. Design Goals

Our design goals include the following aspects:

- *Data confidentiality and privacy preservation*. The underlying message is unable to be recovered from the ciphertext. None is allowed to extract user privacy from real-time power consumption data.
- *Cheap fault tolerance*. CC can still obtain the total consumption from normal users, even if some smart meters

fail to submit their reports or a small number of edge nodes fail to perform data aggregation. Strategies for dealing with error tolerance never rely on stronger assumptions or increase computation/communication overhead on the user side.

- *Smart meter dynamics*. The proposed scheme should provide a flexible user enrollment and revocation mechanism, which allows a SM to join in or quit from deployed systems dynamically.
- *Verifiable aggregation*. An aggregator presents a proof such that anyone can verify whether the aggregated result has been correctly computed or not. Therefore, anyone can detect aggregators' errors immediately.
- *Efficiency*. Smart meters usually have low computing and storage capacities, so encryption algorithms and key management should be lightweight enough for resource-constrained devices.

IV. Preliminaries

In this section, we introduce some underlying building blocks of HTV-PRE.

A. Notation

We denote $R_q = (\mathbb{Z}_q[x]/ < f(x) >, +, *)$ as a modular polynomial ring over base field $\mathbb{Z}_q$. Addition and multiplication over this ring are defined over modulo $(f(x), q)$. A polynomial vector is presented as $\boldsymbol{x} = (x_1, x_2, \cdots, x_m)$ with length m, where $x_i \in R_q (1 \leq i \leq m)$. When we want to access the i-th coefficient of a polynomial y, we write $y[i]$. We define two types of multiplication operations.

$$\boldsymbol{x} \cdot y = (x_1 \cdot y, x_2 \cdot y, \cdots, x_m \cdot y) \in R_q^m, \boldsymbol{x} \in R_q^m, y \in R_q$$

$$\boldsymbol{x} \otimes \boldsymbol{y} = \sum_{i=1}^{m} (x_i \cdot y_i) \in R_q, \boldsymbol{x} \in R_q^m, \boldsymbol{y} \in R_q^m \quad (1)$$

Assume $m = \lceil \log_2 q/r \rceil$, where $r \geq 1$ is a key switching window. Let a polynomial be $x \in R_q$, and we decompose x into m polynomials with 2^r-base $x_i \in R_{2^r} (i = 1, 2, ..., m)$ satisfying $x = x_1 + (2^r)x_2 + (2^r)^2 x_3 + ... + (2^r)^{m-1} x_m \in R_q$. We definite this decomposition as $\text{Bits}(x) = (x_1, x_2, ..., x_m) \in R_{2^r}^m$. Let $y \in R_q$, $\text{Power}(y) = (y, (2^r)y \mod q, ..., (2^r)^{m-1} y \mod q) \in R_q^m$. It can be verified that $\text{Bits}(x) \otimes \text{Power}(y) = x \cdot y$ easily.

We denote $e \in \chi_\epsilon$ as a noise polynomial, where $e[i]$ are independently sampled from a narrow discrete Gaussian error distribution [18] with a parameter ϵ. We introduce $\mathcal{U}_q$ as a discrete uniform distribution over R_q.

B. Proxy Re-Encryption Over Ideal Lattice

Many lattice-based cryptographic constructions are built on learning with errors over rings (Ring-LWE) [18].

The computational Ring-LWE problem is to find a uniform secret $s \in R_q$, given many independent samples with the form $(a_i, b_i = a_i \cdot s + e_i) \in R_q \times R_q (1 \leq i \leq m)$, where $a_i \in R_q$ is uniformly random and each $e_i \in \chi_\epsilon$ follows noise distribution. The decisional Ring-LWE problem is to decide that given (a_i, b_i) are honestly generated as above or randomly sampled over $(\mathcal{U}_q \times \mathcal{U}_q)$ with a non-negligible probability.

Polyakov et al. [36] proposed a proxy re-encryption scheme based on Ring-LWE, namely BV-PRE. Given a ciphertext under user A's public key, BV-PRE can securely transform the ciphertext into another one decryptable by user B's secret key, without full decryption or unallowed access to sensitive data in the middle stages. BV-PRE[36] includes six algorithms (*ParamsGen*, *KeyGen*, *ReKeyGen*, *Enc*, *ReEnc*, *Dec*). It has the lowest time and space complexity among existing lattice-based PRE schemes.

C. Homomorphic Signature

Under a verifiable computing model, a client outsources complex computing tasks to a server. The server runs heavy calculations and returns the result associated with a proof. The public can verify its correctness with the proof at a low computation cost. Recent homomorphic signature scheme [37] offers an efficient approach with non-interactivity and public verifiability.

A homomorphic signature (HS) allows a client to sign a message x using its signing key. Then the client distributes x and its signature to an untrusted server. The server can perform arbitrary computations $y = F(x)$ over this data and homomorphically derive a signature $\sigma_{F,y}$, which certifies that y is the correct output of the computation F over the client's data x. HS contains the following four algorithms:

- *HS.KeyGen*(λ, d, N): Input a security parameter λ, a circuit depth d and a message length N, output private signing key SK_s and public verification key VK_s.
- *HS.Sign*(SK_s, x) : Input signing key SK_s and a message x to be signed, output a signature σ.
- *HS.SignEval*(F, σ): Input a signature σ, and an evaluation circuit $F : \{0,1\}^N \to \{0,1\}^N$, the evaluation algorithm outputs a homomorphically computed signature $\sigma_{F,y}$.
- *HS.Verify*$(VK_s, F, y, \sigma_{F,y})$: Input a verification key VK_s, a circuit F, a value y, and a signature $\sigma_{F,y}$, output 1 if the signature $\sigma_{F,y}$ is valid for data y, output 0 otherwise.

D. Shamir's Secret Sharing

Shamir's Secret Sharing (SSS) [38] includes two algorithms.

- *SSS.Split*: Let $S \in \mathbb{Z}_q$ be a secret to be shared, where q is a prime integer. A dealer randomly chooses $\bar{k} - 1$ coefficients $a_1, a_2, ..., a_{\bar{k}-1} \in \mathbb{Z}_q$, and let $a_0 = S$. Then, the dealer constructs a polynomial of degree $\bar{k} - 1$ as

$$f(x) = \sum_{j=0}^{\bar{k}-1} \left(a_j x^j\right) \bmod q \quad (2)$$

  The dealer computes shares $y_i = f(x_i)$ for $1 \leq i \leq \bar{n}$, where x_i is a participant P_i's number.
- *SSS.Comb*: A collector with $\bar{k}$ or more shares (x_i, y_i) $(1 \leq i \leq \bar{k})$ can reconstruct the secret S by computing

$$S = f(0) = \sum_{i=1}^{\bar{k}} (\lambda_i y_i) \bmod q \quad (3)$$

  where $\lambda_i = \prod_{j=1, j \neq i}^{\bar{k}} \frac{-x_j}{x_i - x_j} \bmod q$.

V. Homomorphic Threshold Proxy Re-Encryption With Re-Encryption Verifiability

A. HTV-PRE: Construction

HTV-PRE takes a ring dimension n, a ciphertext modulus q satisfies $2n|(q-1)$, $f(x)=x^n+1$, $m=\lceil \log_2 q/r \rceil$. All operations are defined over ring R_q. The plaintext modulus is $p \geq 2$. A pseudo-random function $F_\delta : R_q^2 \to R_z^2$ is selected with its range's coefficients from $[-z, z]$, where $z \in \mathbb{Z}$, and δ is a key. In Shamir's secret sharing, the total number is $\bar{n}$, and the threshold value is $\bar{k}$, set $\eta=(\bar{n}!)^2$. To be universal, we temporarily use Π_{HS}=(HS.KeyGen, HS.Sign, HS.SignEval, HS.Verify) as HS without restricting it to a specific scheme. Our HTV-PRE can re-encrypt many ciphertexts from multi-users $U_i (1 \leq i \leq N)$ and then aggregate the results into a single ciphertext decryptable for a receiver U_r, with the help of multiple proxies $P_j (1 \leq j \leq \bar{n})$.

- *HTV-PRE.ParamsGen(n, q)*
  On input n and q, the system generates public parameter pp. A signature server invokes HS.KeyGen$\to$ (VK_s, SK_s). It publishes PP=(pp, VK_s).
- *HTV-PRE.KeyGen(PP)*
  User U_i chooses polynomials $a_i \leftarrow \mathcal{U}_q$ and $s_i, e_i \leftarrow \chi_\epsilon$ randomly, computes $b_i = a_i \cdot s_i + pe_i \in R_q$. The key pair for U_i is $(PK_i=(a_i, b_i), SK_i=(s_i))$. Similarly, a receiver U_r selects its private key $SK_r=(s_r)$. To enable re-encryption, for $k=1, 2, \ldots, m$, U_r chooses polynomials $\beta_k \leftarrow \mathcal{U}_q$ and $e_k \leftarrow \chi_\epsilon$, then computes $(\boldsymbol{\beta}, \boldsymbol{\theta})=(\beta_k, \theta_k = \beta_k \cdot s_r + pe_k)(1 \leq k \leq m)$. U_r publishes $PK_r=(\boldsymbol{\beta}, \boldsymbol{\theta})$.
- *HTV-PRE.ReKeyGen*$(SK_i=(s_i), PK_r)$
  User U_i $(1 \leq j \leq \bar{n})$ calculates proxy key shares $\{\overline{RK}_{i,j}\}$ from $U_i (1 \leq i \leq N)$ to U_r as follows.
  Step 1: U_i computes $\boldsymbol{\theta}' = (\theta_1 - pe_1, \theta_2 - pe_2, ..., \theta_m - pe_m)$, where $e_k \leftarrow \chi_\epsilon (1 \leq k \leq m)$.
  Step 2: U_i computes $\boldsymbol{\gamma}_i = \boldsymbol{\theta}' - \mathrm{Power}(s_i) \in R_q^m$, and gets $RK_i = (\boldsymbol{\beta}_i = \boldsymbol{\beta}, \boldsymbol{\gamma}_i)$. Next, U_i takes $RK_i \in R_q^{2m}$ as two matrices with size $n \times m$, and shares each element one by one using SSS.Split. For $1 \leq j \leq \bar{n}$, P_j's proxy key share is $\bar{\boldsymbol{u}}_{i,j} = (\bar{\boldsymbol{\beta}}_{i,j}, \bar{\boldsymbol{\gamma}}_{i,j})$.
  Step 3: U_i selects $\bar{n}$ independent keys $\delta_{i,1}, \delta_{i,2}, \cdots, \delta_{i,\bar{n}}$. U_i sends $\bar{\boldsymbol{x}}_{i,j} = (\bar{\boldsymbol{u}}_{i,j}, \delta_{i,j})(1 \leq j \leq \bar{n})$ to the signature server. It returns $\bar{\sigma}_{i,j} = \mathrm{HS.Sign}(SK_s, \bar{\boldsymbol{x}}_{i,j})$ back to U_i.
  Step 4: The share $\{\overline{RK}_{i,j}\} = \{\bar{\boldsymbol{x}}_{i,j}, \bar{\sigma}_{i,j}\}(1 \leq j \leq \bar{n})$ is sent to the jth proxy P_j through a secure channel.
- *HTV-PRE.Enc*$(PK_i, \bar{m}_i \in \mathcal{M})$
  U_i selects some polynomials $v, e_0, e_1 \in \chi_\epsilon$ randomly, then encrypts its message $\bar{m}_i$

$$g_i = b_i \cdot v + pe_0 + \bar{m}_i \in R_q,$$
$$h_i = a_i \cdot v + pe_1 \in R_q \tag{4}$$

  U_i's ciphertext is $C_i=(g_i, h_i) \in R_q^2$.
- *HTV-PRE.ReEnc*$(\{\overline{RK}_{i,j}\}, C_i)$
  $P_j (1 \leq j \leq \bar{n})$ performs re-encryption on C_i with the proxy key share $\{\overline{RK}_{i,j}\} = \{(\bar{\boldsymbol{\beta}}_{i,j}, \bar{\boldsymbol{\gamma}}_{i,j}), \delta_{i,j}, \bar{\sigma}_{i,j}\}$.
  Step 1: P_j computes

$$g'_{i,j} = g_i + \mathrm{Bits}(h_i) \otimes \bar{\boldsymbol{\gamma}}_{i,j} \in R_q,$$
$$h_{i,j}{}' = \mathrm{Bits}(h_i) \otimes \bar{\boldsymbol{\beta}}_{i,j} \in R_q \tag{5}$$

  and $(e_0', e_1') = F_{\delta_{i,j}}(g_i, h_i) \in R_z^2$.
  Step 2: P_j obtains $\overline{C}_{i,j}$=$(\bar{g}_{i,j}, \bar{h}_{i,j}) \in R_q^2$ where $\bar{g}_{i,j} = g_{i,j}{}' + \eta pe_0'$, $\bar{h}_{i,j} = h_{i,j}{}' + \eta pe_1'$.
  Step 3: The evaluation circuit for homomorphic signature is defined as

$$\begin{aligned} &F_{C_i}\left(\bar{\boldsymbol{\beta}}_{i,j}, \bar{\boldsymbol{\gamma}}_{i,j}, \delta_{i,j}\right) \\ &\quad = \left(g_i + \mathrm{Bits}(h_i) \otimes \bar{\boldsymbol{\gamma}}_{i,j}, \mathrm{Bits}(h_i) \otimes \bar{\boldsymbol{\beta}}_{i,j}\right) \\ &\qquad + \eta p F_{\delta_{i,j}}(g_i, h_i) \end{aligned} \tag{6}$$

  P_j computes HS.SignEval$(F_{C_i}, \bar{\sigma}_{i,j}) = \bar{\sigma}'_{i,j}$, and outputs $\{\overline{C}_{i,j}, \bar{\sigma}'_{i,j}\}$.
- *HTV-PRE.Agg*$(\{\overline{C}_{i,j}, \bar{\sigma}'_{i,j}\})$
  P_j aggregates all the transformed ciphertext shares

$$\overline{C}_j = \left(\bar{g}_j = \sum\nolimits_{i=1}^{N} \bar{g}_{i,j}, \bar{h}_j = \sum\nolimits_{i=1}^{N} \bar{h}_{i,j}\right). \tag{7}$$

  Meanwhile, it aggregates all the signatures

$$\bar{\sigma}_j = \sum\nolimits_{i=1}^{N} \bar{\sigma}'_{i,j}. \tag{8}$$

  At last, P_j outputs $\{\overline{TC}_j\} = \{\overline{C}_j, \bar{\sigma}_j\}$.
- *HTV-PRE.Verify*$(PP, \{\overline{TC}_j\})$
  It is required to check HS.Verify$(VK_s, G, \overline{C}_j, \bar{\sigma}_j) = 1$ or not, where $G = \sum_{i=1}^{N} F_{C_i}$.
- *HTV-PRE.Comb*$(\{\overline{TC}_j\}, \{C_i\})$
  Given more than $\bar{k}$ ciphertext shares provided by k proxies (their numbers form a new subset S), a collector obtains a combined ciphertext by SSS.Comb.

$$C_r = (g_r, h_r) = \sum_{j \in S} \lambda_j \left(\bar{g}_j, \bar{h}_j\right) + \sum_{j \in S} g_s(1 - \lambda_j, 0) \tag{9}$$

  where $g_s = \sum_{i=1}^{N} g_i$, λ_j is computed as (3).
- *HTV-PRE.Dec*(SK, C)
  On input $SK=(s)$ and $C=(g, h)$, U_r calculates $t = g - s \cdot h \in R_q$ and $m' = t \bmod p$. It outputs a message m'.

B. HTV-PRE: Correctness

Theorem 1. (Correctness). Let B_e be the upper bound of discrete Gaussian distribution χ_ϵ, and B_z be the upper bound of uniform pseudo-random function F_δ, set $q > 2pN[3\sqrt{n}B_e^2 + 2^{r+1}mnB_e + \bar{k}(\bar{n}!)^3 B_z(B_e+1)]$. Both original ciphertexts and aggregated ciphertexts can be decrypted correctly.

Proof. (1) U_i can successfully decrypt original ciphertext $C_i=(g_i, h_i)$ with its private key s_i.

$$\begin{aligned} g_i - s_i \cdot h_i &= b_i \cdot v + pe_0 + \bar{m}_i - s_i \cdot (a_i \cdot v + pe_1) \\ &= (a_i \cdot s + pe) \cdot v + pe_0 + \bar{m}_i - s_i \cdot (a_i \cdot v + pe_1) \\ &= \bar{m}_i + \underbrace{p(e \cdot v + e_0 - s_i \cdot e_1)}_{noise} \end{aligned} \tag{10}$$

As long as the noise $p(e \cdot v + e_0 - s_i \cdot e_1)$ does not exceed $q/2$, it can still recover m after mod p. Indeed, noises v, s_i, e_0, e_1, e

are not more than B_e, their sum $p(e \cdot v + e_0 - s_i \cdot e_1) \leq 3\sqrt{n}pB_e^2$. Therefore, $q \geq 6\sqrt{n}pB_e{}^2$ is sufficient.

(2) CC can successfully decrypt aggregated ciphertexts. We present its proof in Appendix A *(available online)*.

C. HTV-PRE: Security

Theorem 2. (CPA security). If the Ring-LWE assumption holds, HTV-PRE can realize CPA security.

Proof. Indeed, our HTV-PRE is modified from BV-PRE [36], which has been proven CPA (Indistinguishability under Chosen Plaintext Attack) secure. Therefore, we only need to confirm that our modifications to BV-PRE will not affect its security.

Specifically, there are four differences between BV-PRE [36] and HTV-PRE. (1) BV-PRE transforms a single ciphertext from one sender, while HTV-PRE has to aggregate multiple transformed ciphertexts from multi-users into a single ciphertext, and then decrypts it in one slot. The "aggregation" on ciphertexts should not affect security. (2) Regarding proxy key queries between honest users, BV-PRE returns a whole proxy key, while HTV-PRE returns $\bar{n}$ key shares after SSS.Split. In addition, regarding proxy key queries from honest users to corrupted users, BV-PRE is unallowed, while HTV-PRE can return $\bar{k}-1$ key shares owned by $\bar{k}-1$ corrupted proxies. The "threshold" on key generation should not affect security. (3) It is consistent between re-encryption queries and proxy key queries in BV-PRE because an attacker can perform re-encryption with the corresponding proxy key. The "threshold" should not break the consistency. (4) The introduced HS does not break the security of BV-PRE.

Therefore, we take a provable security method. The basic idea is to construct a series of games. The first game is an actual attack on HTV-PRE, and the last attack is on BV-PRE. The intermediate games match these four differences listed above. We only need to prove that the probability of an attacker winning neighboring games keeps the same in polynomial time. We distinguish two terms. A whole ciphertext is a combination of multiple ciphertext shares by HTV-PRE.Comb. An aggregated ciphertext is a summation of N ciphertexts by HTV-PRE.Agg.

Game 0: A real CPA attack game for HTV-PRE scheme.

Game 1: In Game 0, all the ciphertext shares have been aggregated using HTV-PRE.Agg by P_j. While in Game 1, P_j provides all the ciphertext shares without aggregation. The collector first performs HTV-PRE.Comb on each user's ciphertext shares. Then it aggregates *N* combined ciphertexts together. Other settings are identical to Game 0.

Game 1 is equivalent to Game 0. Whether "HTV-PRE.Agg and HTV-PRE.Comb" or "HTV-PRE.Comb and HTV-PRE.Agg", are computationally equivalent because linear operations are exchangeable. Therefore, batch decryption on multiple ciphertexts under the same receiver's public key will not affect CPA security. HTV-PRE can be considered as various BV-PRE instances running parallelly.

Game 2: We make the following improvements to Game 1. When proxy key queries are performed, a challenger $\mathcal{C}$ first calculates a whole proxy key RK_i, then forwards RK_i, $\bar{n}$ and $\bar{k}$ to its assistant $\mathcal{S}$. The rest of the work is done by $\mathcal{S}$. At last, $\mathcal{S}$ sends all $\bar{n}$ key shares $\{\overline{RK}_{i,j}\}(1 \leq j \leq \bar{n})$ back to attacker $\mathcal{A}$ if queries occurred between honest users, or only first $\bar{k}$-1 shares $\{\overline{RK}_{i,j}\}(1 \leq j \leq \bar{k}-1)$ if queries occurred from honest users to corrupted users.

Game 2 is equivalent to Game 1. The proxy key generation in Game 1 is now accomplished by the cooperation between assistant $\mathcal{S}$ and challenger $\mathcal{C}$ in Game 2. Because SSS.Split and SSS.Comb can complete in polynomial time. Providing a whole proxy key is equivalent to delivering $\bar{n}$ shares in information theory.

Game 3: We make the following improvements. Challenger $\mathcal{C}$ no longer provides homomorphic signatures by HS.SignEval honestly, but obtains $\bar{\sigma}'_{i,j}$ by directly interacting with the signature server.

Game 3 is equivalent to Game 2. Because Π_{HS} is content-hiding, neither original messages nor their signatures will be leaked from the evaluated result in the attacker's view. Moreover, the signature can be verified by VK_s successfully.

Game 4: When obtaining $\bar{C}_{i,j}$, challenger $\mathcal{C}$ no longer computes $(e_0{}', e_1{}') = F_{\delta_{i,j}}(g_i, h_i) \in R_z^2$ honestly, but selects two polynomials $(e_0{}', e_1{}') \in R_z^2$ randomly.

Game 4 is equivalent to Game 3. Whether $(e_0{}', e_1{}')$ are selected randomly from uniform distribution or calculated honestly by pseudo-random function, the final result is uniform within $\{-z, -(z-1), ..., -1, 0, 1, ..., z-1, z\}^{2n}$, which is indistinguishable in attacker $\mathcal{A}$'s view.

Game 5: In the previous games, challenger $\mathcal{C}$ transforms ciphertexts honestly with a real proxy key. In Game 5, if proxy key queries are from honest users to corrupted users, challenger $\mathcal{C}$ randomly selects a uniform $(\boldsymbol{\beta}_i, \boldsymbol{\gamma}_i) \in R_q^{2m}$ as RK_i and then sends $\bar{k}-1$ key shares $(\bar{\boldsymbol{\beta}}_{i,j}, \bar{\boldsymbol{\gamma}}_{i,j})(1 \leq j \leq \bar{k}-1)$ to attacker $\mathcal{A}$ (the corrupted proxies' numbers constitute a set S^*, $|S^*| = \bar{k}-1$). After that, a whole transformed ciphertext can be obtained as follows.

Step 1: Calculate $\lambda_w^{\tilde{S}^*}(w \in \tilde{S}^*)$, where $\tilde{S}^* = S^* \cup \{0\}$.

Step 2: Select $e'_0, e'_1 \in R_z^2$ randomly.

Step 3: Calculate

$$\begin{aligned}(\bar{g}'_i, \bar{h}'_i) = {} & \lambda_0^{\tilde{S}^*}(g_i + \boldsymbol{\gamma}_i \otimes \mathrm{Bits}(h_i), \boldsymbol{\beta}_i \otimes \mathrm{Bits}(h_i)) \\ & + \sum_{j \in S^*} \lambda_j^{S^*}\left(g_i + \bar{\boldsymbol{\gamma}}_{i,j} \otimes \mathrm{Bits}(h_i), \bar{\boldsymbol{\beta}}_{i,j} \otimes \mathrm{Bits}(h_i)\right) \\ & + \eta p\,(e'_0, e'_1) \in R_q^2 \end{aligned} \tag{11}$$

Game 5 is equivalent to Game 4. In (11), the first part $\lambda_0^{\tilde{S}^*}(g_i + \boldsymbol{\gamma}_i \otimes \mathrm{Bits}(h_i), \boldsymbol{\beta}_i \otimes \mathrm{Bits}(h_i))$ can be regarded as one ciphertext share provided by a proxy with number 0 and proxy key share $(\boldsymbol{\beta}_i, \boldsymbol{\gamma}_i)$. The second part can be regarded as $\bar{k}-1$ ciphertext shares provided by $\bar{k}-1$ proxies from set S^*. The sum of two parts exactly reaches the threshold $\bar{k}$. Therefore, the consistency between proxy key shares and ciphertext shares is still maintained.

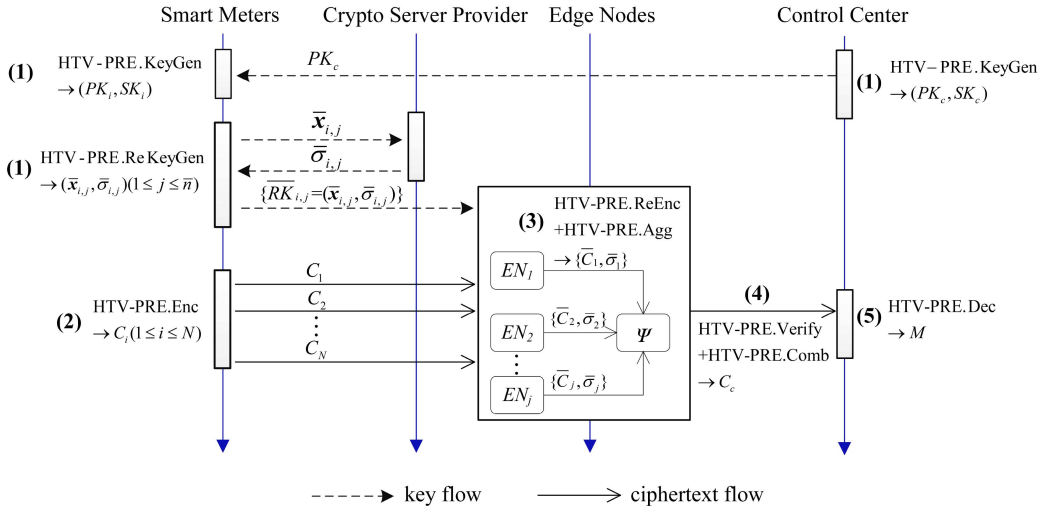


Fig. 3. Workflow of the proposed PPDA scheme.

So far, the attack on HTV-PRE can be completely transformed into the corresponding attack on BV-PRE in polynomial time. Our HTV-PRE is CPA secure.

Theorem 3. (Verifiability). The HTV-PRE is computationally verifiable, provided that homomorphic signature Π_{HS} is unforgeable.

Proof. Intuitively, the verification will fail if a forged circuit is different from the right one, i.e., $G = \sum_{i=1}^{N} F_{C_i}$. Therefore, proxies are forced to perform honest transformations. Formally, the verifiability of HTV-PRE can be reduced to the unforgeability of HS. If an adversary $\mathcal{A}$ can break the verifiability of HTV-PRE, we can construct a simulator $\mathcal{S}$ to break the unforgeability of HS by interacting with $\mathcal{A}$ as follows:

$\mathcal{S}$ first obtains verification key VK_s^*, and $\mathcal{A}$ chooses the target proxy P_j^* intended to attack. When $\mathcal{A}$ submits a forged transformed ciphertext share $\{\overline{TC}_j^*\} = \{\overline{C}_j{}^*, \bar{\sigma}_j^*\}$ to $\mathcal{S}$, $\mathcal{S}$ reshapes it as $(VK_s^*, G = \sum F_{i=1}^{N} C_i, \overline{C}_j{}^*, \bar{\sigma}_j^*)$ and forwards it to a homomorphic signature oracle.

If adversary $\mathcal{A}'s$ ciphertext and its proof can pass verifiability, that is, $\{\overline{TC}_j^*\}$ is not computed honestly, but $\bar{\sigma}_j^*$ is a valid signature for $\bar{C}_j^*$ under VK_s^*. So $\mathcal{A}$ has successfully forged an illegal signature. Homomorphic signature oracle will successfully verify the received signature by HS.Verify$(VK_s^*, G, \bar{C}_j^*, \bar{\sigma}_j^*) = 1$. It means the unforgeability of Π_{HS} is breached. However, this is contrary to our assumption.

Therefore, if the unforgeability of Π_{HS} holds, HTV-PRE is computationally verifiable.

VI. Robust and Lightweight Privacy-Preserving Data Aggregation Scheme for Smart Grid

This section will apply HTV-PRE to smart grid to get a robust and lightweight PPDA scheme.

A. Construction of the PPDA Scheme for Smart Grid

Data aggregation for smart grid requires transforming multiple encrypted readings under different public keys into a single ciphertext under CC's public key. HTV-PRE is entirely compatible with smart grid scenarios if we take SM_i as U_i, EN_j as P_j, and CC as the final receiver U_r, respectively. There are N smart meters in an aggregation area. Some symbols are defined in Table II, and the workflow is described in Fig. 3. The key flow launches only once on system initialization. In contrast, the ciphertext flow starts up every data aggregation.

TABLE II
Table of Symbols

Notation	Description
U_i, SM_i	A user i or smart meter i, we use both without distinguishing
EN_j	Edge node j
(PK_i, SK_i)	The public key and private key for SM_i
(PK_c, SK_c)	The public key and private key for CC
(SK_s, VK_s)	The signing key and verification key for CSP
C_i	A ciphertext generated by SM_i
$(\bar{k}, \bar{n})$	$\bar{k}$ is a threshold, and $\bar{n}$ is the total member of edge nodes
$\bar{C}_{i,j}$	A ciphertext share generated by EN_j by transforming C_i
$\bar{C}_j$	An aggregated ciphertext share generated by EN_j
$\{\overline{RK}_{i,j}\}$	A proxy key share generated from $SM_i(1 \le i \le N)$ and distributed to $EN_j(1 \le j \le \bar{n})$
$\{\overline{TC}_j\}$	A ciphertext share and its signature generated by $EN_j(1 \le j \le \bar{n})$

Step 1: System initialization

CC bootstraps the system by *HTV-PRE.ParamsGen*$(\lambda) \rightarrow PP$. The signing key is kept secret by CSP. Then $SM_i(i = 1, 2, \ldots, N)$ registers and gets its key pairs by *HTV-PRE.KeyGen*$(PP) \rightarrow (PK_i, SK_i)$. CC gets its key pair *HTV-PRE.KeyGen*$(PP) \rightarrow (PK_c, SK_c)$. Note that CC prepares multiple public keys to enable proxy re-encryption.

$SM_i(i = 1, 2, \ldots, N)$ calculates the proxy key shares from SM_i to CC by *HTV-PRE.ReKeyGen*$(SK_i, PK_c) \rightarrow \{\overline{RK}_{i,j}\}(1 \le j \le \bar{n})$, and then distributes them to the corresponding edge nodes.

Step 2: Data reporting

In each reporting period, SM_i collects its total power consumption $M_i \in \mathbb{Z}$, then encodes M_i as $\bar{m}_i \in R_2$. It is easy to get $\bar{m}_i$ with 2-base decomposition on M_i such that $M_i = \sum_{k=0}^{n-1}(2^k \cdot \bar{m}_i[k])$.

SM_i encrypts $\bar{m}_i$ by *HTV-PRE.Enc*$(PK_i, \bar{m}_i) \rightarrow C_i$, and sends C_i to the edge.

Step 3: Data aggregation and proof generation

At the end of the current aggregation period, we assume $\tilde{N}(\tilde{N} \leq N)$ ciphertexts successfully received from $SM_i(i = 1, 2, ..., \tilde{N})$ without loss of generality. Similarly, there are currently $\tilde{n}(\tilde{n} \leq \bar{n})$ edge nodes available. $EN_j(1 \leq j \leq \tilde{n})$ performs re-encryption by *HTV-PRE.ReEnc*$(\{\overline{RK}_{i,j}\}, C_i) \rightarrow \{\overline{C}_{i,j}, \bar{\sigma}'_{i,j}\}$ in parallel.

At last, $EN_j(1 \leq j \leq \tilde{n})$ aggregates all the SMs' ciphertext shares and their signatures by *HTV-PRE.Agg*$(\{\overline{C}_{i,j}, \bar{\sigma}'_{i,j}\}) \rightarrow \{\overline{TC}_j = (\bar{C}_j, \bar{\sigma}_j)\}$.

Step 4: Proof verification and data combination

When $\tilde{n}$ edge nodes have finished their works, a public auditor Ψ is selected from $\tilde{n}$ edge nodes randomly to finish the rest of the work.

Ψ checks *HTV-PRE.Verify*$(PP, \{\overline{TC}_j\}) \rightarrow 1$ or not. If verification fails, Ψ abandons $\overline{TC}_j$. An edge node providing the wrong aggregated results will be published by reducing its credit score. Once the credit score is lower than the predefined value, its service provider will be removed from the list.

Let S' be a set of valid edge nodes, and its size be $|S'| = k'$. If $k' < \bar{k}$, output $\perp$, and the data aggregation fails; Otherwise, Ψ takes only $\bar{k}$ elements from S' randomly as a new subset S.

Ψ generates a whole ciphertext by *HTV-PRE.Comb* $(\{\overline{TC}_j\}, \{C_i\}) \rightarrow C_c$, where $(1 \leq i \leq N, j \in S)$. Ψ forwards C_c to CC.

Step 5: Decryption

CC runs *HTV-PRE.Dec*$(SK_c, C_c) \rightarrow m'$. Then CC gets $M = \sum_{k=0}^{n-1}(2^k \cdot m'[k])$.

The correctness of the proposed PPDA is as follows.

$$\begin{aligned} M &= \sum_{k=0}^{n-1}\left(2^k \cdot m'[k]\right) \\ &= \sum_{k=0}^{n-1}\left(\sum\nolimits_{i=1}^{\tilde{N}} 2^k \cdot \bar{m}_i[k]\right) \\ &= \sum_{i=1}^{\tilde{N}}\left(\sum\nolimits_{k=0}^{n-1} 2^k \cdot \bar{m}_i[k]\right) \\ &= \sum_{i=1}^{\tilde{N}} M_i \end{aligned} \tag{12}$$

Our PPDA scheme is correct as long as there is no wrap-around modulo p, so $p \geq N + 1$ is sufficient.

B. Analysis of the PPDA Scheme for Smart Grid

- *Confidentiality*. In the beginning, each SM_i's power consumption is encrypted by its own public key PK_i. None else can decrypt C_i except SM_i itself. HTV-PRE provides a means for authorizing CC to access only the total consumption. $EN_j(1 \leq j \leq \bar{n})$ handles the re-encryption process without extracting any information about the plaintext. Theorem 2 claims HTV-PRE is semantically secure (CPA security). Thus, throughout its lifecycle, SM_i's consumption data are always kept encrypted (C_i or $\bar{C}_{i,j}$) and remains confidential against outside attackers and edge nodes. Besides, The Ring-LWE-based PPDA scheme is considered post-quantum secure.
- *Strong privacy preservation*. The application of the threshold technique makes our proposed PPDA scheme immune to collusion attack, by spitting a whole proxy key RK_i into $\{\overline{RK}_{i,j}\}(1 \leq j \leq \bar{n})$ for multiple edge nodes EN_j. A final ciphertext C_c will be obtained if and only if at least $\bar{k}$ nodes are willing to perform transformations honestly. Intuitively, it seems possible to corrupt one edge node but many. Even better, we require edge nodes to provide aggregated ciphertext shares $\bar{C}_j$ after HTV-PRE.Agg, making an individual ciphertext decryptable for CC will never appear even in memory temporarily. In short, CC cannot violate individual user privacy, even colluding with up to $\bar{k} - 1$ edge nodes. The collusion attack existing in HE+PRE will never happen in HTV-PRE. Our PPDA achieves strong privacy preservation.
- *Cheap fault tolerance*. Our PPDA scheme can normally work even if several SMs or a small number of ENs are faulty.

  (1) SMs' faults. When SM_i fails, the ciphertext C_i will never be uploaded in the current aggregation period. Accordingly, edge nodes will only receive $\tilde{N}(\tilde{N} \leq N)$ ciphertexts. The absence of faulty SMs does not affect other peers. EN_j converts and aggregates the received ciphertexts as normal. That is,

  $$\overline{C}_j = \sum\nolimits_{i=1}^{\tilde{N}} \overline{C}_{i,j} = \left(\sum\nolimits_{i=1}^{\tilde{N}} \bar{g}_{i,j}, \sum\nolimits_{i=1}^{\tilde{N}} \bar{h}_{i,j}\right) \tag{13}$$

  The subsequent decryption is not affected in any way.

  (2) ENs' faults. Shamir's secret sharing will still work if at least $\bar{k}$ out of $\bar{n}$ shares are available. If the number of available ENs satisfies $\tilde{n} \leq \bar{k}$, the correct ciphertext can still be reconstructed successfully.

  Note that our scheme supports fault tolerance naturally without relying on online TA to deal with SM's dropouts or increasing computation/communication overhead on the user side.
- *Smart meter dynamics*. Our PPDA scheme admits efficient user dynamics, including meter enrollment, revocation, and update. This sharply contrasts existing schemes requiring key updates or key redistribution among unchanged users.

  (1) Enrollment. All that the added SM_i needs to do is distributing proxy key shares and their signatures to the edge. In the next data reporting period, once SM_i uploads its ciphertext, ENs will employ the corresponding proxy key shares for conversion and aggregation. Therefore, the final aggregated result will contain SM_i's power consumption.

  (2) Revocation. When SM_i is revoked from the deployed system, non-revoked users are unaffected. All that caused

by user revocation is simply notifying all the edge nodes to erase the revoked SM_i's proxy key shares. In this way, ENs cannot perform re-encryption for SM_i anymore.
(3) Update. SM_i generates a new key pair (PK_i, SK_i), and then replaces the old proxy key shares and their signatures with the new ones.

- *Verifiable aggregation*. This property can be derived from Theorem 3 easily. Incorrect or forged aggregated results will be identified immediately.

VII. Performance Evaluation

A. Instantiation

Recent advances in homomorphic signature schemes over lattices provide an important tool for verifiable computation. Gorbunov et al. [37] constructed a leveled fully homomorphic signature scheme with the ability to evaluate arbitrary circuits over signed data. It achieves compactness, unforgeability, and context-hiding, which are sufficient for our work. Based on the ideas from [39], we modify the original scheme[37] to ideal lattice settings. Working over ring allows for shorter signatures, smaller key sizes, and faster computation. Because the Bits function in our evaluation circuit exactly decides specific computation types, we further tailor the original scheme[37] to addition and multiplication by a constant. Therefore, the maximal depth of the circuit can be fixed as small as possible.

B. Evaluation

To evaluate the performance of the proposed PPDA, we mainly focus on: (1) Data encryption on the user side. (2) Computation cost of achieving verifiable aggregation. (3) Total computation cost for smart meters, aggregators, and control center, respectively. (4) Different methods of fault-tolerance. (5) Communication overhead among related entities.

There are two experimental environments. A laptop is configurated with an Intel Core i7-7700HQ CPU rated at 3.8GHz and 16GB memory. It runs Ubuntu 20.04 operating system. A Raspberry Pi 4 Model B is configurated with a 1.5GHz ARM Cortex-A72 CPU and 4GB memory. It runs Ubuntu MATE 21.04. Only the first experiment is conducted on Raspberry Pi to simulate a resource-constrained device; others are not.

We require all the related schemes to provide at least 100 bits of security. We implement HTV-PRE by PALISADE Library [40]. The basic parameters include plaintext modulus $p = 1023$, ciphertext modulus $q \approx 2^{56}$, security parameter $n = 512$, discrete Gaussian distribution with parameter $\sigma_\epsilon = 4$, and key switching window $r = 8$. Damien Giry[41] provides the appropriate key length for our desired security level, which allows us to compare different encryption methods easily. For the cryptographic schemes[32], [42], [43], [44] based on the bilinear pairings or elliptic curve, we employ a type A curve and set the group order as 224-bit, and the order of base field as 512-bit. For the cryptographic schemes[6], [7], [14] based on the Paillier cryptosystem, the secure primes are 1024-bit, so the ciphertext modulus is about 4096-bit. We pick a random integer from 1 to 10000 as each SM's reading. Table III lists some operations and execution time on the laptop. Note that our timekeeping requires an element from its whole space randomly. We do not consider any optimization or pre-calculation.

TABLE III
Symbols Definition and Execution Time on Laptop

Notation	Description	Runtime(ms)
M_{poly}	polynomial multiplication over rings	2.62
A_{poly}	polynomial addition over rings	0.45
Add_{ecc}	point addition over elliptic curve	0.05
Mul_{ecc}	point multiplication over elliptic curve	14.64
Exp_{pair}	exponentiation operation from bilinear map	15.12
Mul_{pair}	multiplication operation from bilinear map	0.03
BP_{pair}	bilinear pairing operation	8.98
$Hash_{pair}$	hash-to-point operation from bilinear map	20.46
$Exp_{Paillier}$	exponentiation operation in Paillier	30.57
$Mul_{Paillier}$	multiplication operation in Paillier	0.004

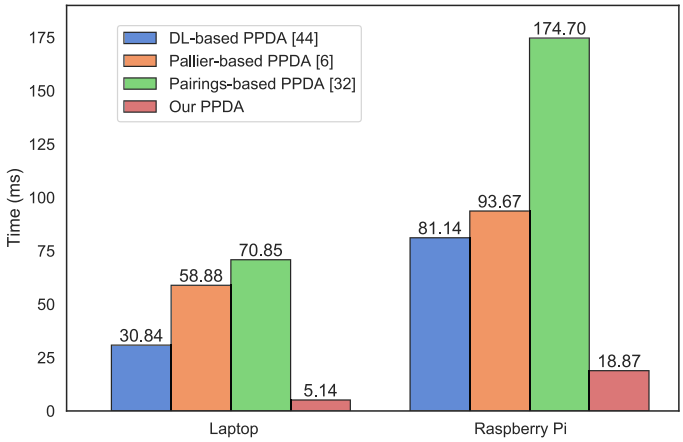


Fig. 4. Encryption time for each smart meter simulated by Raspberry Pi and Laptop, respectively.

(1) Data encryption on the user side.

We concentrate on data encryption for each SM, so we temporarily separate additional computations (including signature and hash) from the original constructions to make a fair comparison. Fig. 4 shows the runtime of different PPDA schemes on Raspberry Pi and Laptop, respectively.

The typical PPDA schemes from HE are categorized into four types. (i) *Paillier-based PPDA*, represented by FESDA[6] and PPSO[7]. FESDA [6] involves $2Exp_{Paillier}$+$Mul_{Paillier}$ to encrypt a message. (ii) *Discrete logarithm-based PPDA*, usually instantiated by lifted ElGamal or Elliptic Curve (EC). This category includes 3PDA[43], Boudia et al. 's PPDA [44], and BB-MDA[45]. The selected scheme[44] takes $3Mul_{ecc}$+$2Add_{ecc}$ where Mul_{ecc} to encode a message over the specified finite field and other $2Mul_{ecc}$ to conceal it. (iii) *Pairing-based PPDA*, represented by Emura et al. [32], P [2] SM [12], CB-DA[46], and Wang's[42] PPDA. The selected scheme [32] takes $3Exp_{pair} + 2Hash_{pair} + 2Mul_{pair}$. (iv) *Ring-LWE-based PPDA*. Our scheme falls into this category. It takes only $2Mul_{poly}$+$3Add_{poly}$.

The bottleneck of computation capability on smart grid is smart meter. Data encryption on the user side should impose

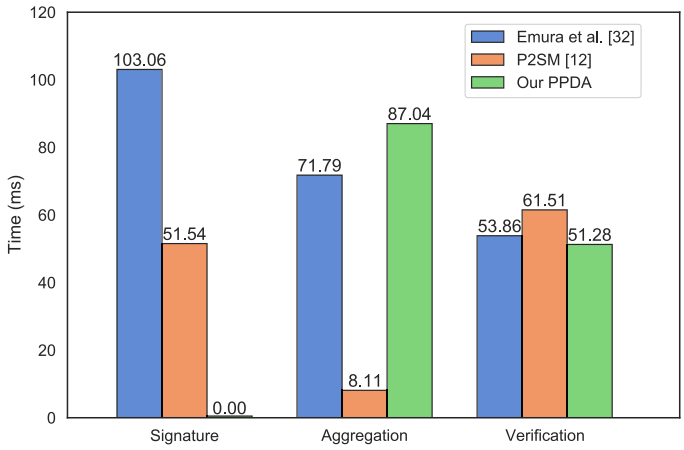


Fig. 5. Runtime of different verifiable aggregation methods.

as minimal overhead as possible. Fortunately, our HTV-PRE is quite efficient for smart meters. Compared with the best performance from Elliptic Curve-based PPDA [44], our PPDA can be up to 4.3x faster on Raspberry Pi.

(2) Verifiable aggregation

Our scheme is compared with Emura et al. 's scheme [32] and P ^{2} SM [12] in Fig. 5. These three schemes can verify aggregators' behaviors. Our PPDA is based on homomorphic signature over ideal lattices. Emura et al. 's scheme [32] is constructed over modified computational Diffie-Hellman hard assumption in the generic bilinear group. P ^{2} SM [12] uses proxy re-signature supporting homomorphism. The number of smart meters N should be kept reasonably small to perceive the basic performance of different signatures. We take the smallest value $N = 1$.

Our PPDA never provides signatures over current consumption data in each data reporting period. It provides proxy key shares to ENs only once in the initialization phase. In contrast, Emura et al. 's scheme[32] and P ^{2} SM [12] have to generate a signature (taking $4Exp_{pair} + 3Mul_{pair} + 3Hash_{pair}$ and $3Exp_{pair} + 2Mul_{pair} + Hash_{pair}$, respectively) in every reporting period, which inevitably increases the computation cost on the user side. On signature aggregation, Emura et al. 's scheme [32] and P ^{2} SM require $(N+1)Mul_{pair}$+$2Hash_{pair}$+$2Exp_{pair}$ and $(N-1)Mul_{pair}$+$(N)BP_{pair}$, while our PPDA transforms each proxy key share's signature with HS.SignEval, requiring about $2(m^2)Mul_{poly} + 2(m-1)Add_{poly}$ for each SM. All the edge nodes can work in parallel. Finally, our PPDA takes $2(N-1)Add_{poly}$ to aggregate the evaluated signatures. On public verification, our PPDA needs $4mMul_{poly} + 2(m+1)Add_{poly}$, while Emura et al. 's scheme[32] and P ^{2} SM take $2BP_{pair} + Hash_{pair} + Exp_{pair} + Mul_{pair}$ and $BP_{pair} + (N)Hash_{pair} + 3Exp_{pair} + (N+1)Mul_{pair}$, respectively.

The most attractive advantage of our verifiable aggregation is adding no computation/communication overhead on the user side.

(3) Total computation cost

We compare our scheme with David et al. 's[16] and P ^{2} SM [12]. Total computation cost includes all the encryption and signature operations performed by smart meters, aggregators, and control centers, respectively. Fig. 6 shows total computation cost grows approximately linearly as more SMs are added. Our PPDA incurs a lower computation cost than the other two schemes on both SMs and CC sides.

Our PPDA runs Shamir's secret sharing under a predefined threshold $(\bar{k} = 3, \bar{n} = 5)$. Data encryption and signatures have already been evaluated above. On data aggregation, its computation cost includes two parts. The first part is re-encryption and aggregation on each edge node (computation cost for F_δ is neglected). Each edge node goes parallelly with the others. So the first part will take about $2(mN)Mul_{poly}$ + $2(m+1)NAdd_{poly}$. The second part is ciphertext reconstruction with $2(\bar{k}-1)\ Add_{poly}$ by the public auditor. The decryption performed by CC is quite simple, only Mul_{poly}+Add_{poly}.

David et al. 's PPDA[16] is built upon the PRE scheme[47]. For the desired homomorphism, it encodes a message m_i into the exponent and then encrypts $\boldsymbol{g}^{m_i}$. Each SM needs to perform $3Exp_{pair}$+Mul_{pair}. After receiving N ciphertexts, an aggregator re-encrypts them with $(N)BP_{pair}$, and finally multiples them together with $(N-1)Mul_{pair}$. Decryption yields $T = \boldsymbol{g}^{\sum_{i=1}^{N} m_i}$ and additionally needs to compute $\sum_{i=1}^{N} m_i = log_{\boldsymbol{g}} T$ with Pollard's lambda method[48]. This method is expected to compute $O(\sqrt{M})$ group operations and to store $O(logM)$ group elements, where $\boldsymbol{M}$ is plaintext space. However, solving discrete logarithm problems is time-consuming when $\boldsymbol{M}$ turns quite large. David et al. 's PPDA[16] also employs the same proxy re-signature as P ^{2} SM [12]. Therefore, their computation costs on signature are similar.

P ^{2} SM [12] belongs to Pairing-based PPDA. The data aggregation is quite simple, only $2(N-1) * Mul_{pair} + (N-1) * Add_{pair}$. Decryption first requires $Exp_{pair} + BP_{pair} + Mul_{pair}$ and then recovers the result using Pollard's lambda method [48].

(4) Fault tolerance

We compare some fault-tolerant PPDA schemes in Table IV. Knirsch et al. [20] proposed a masking approach including two

TABLE IV
COMPARISON OF RELATED FAULT-TOLERANT SCHEMES

Schemes	F1	F2	F3	F4	F5	F6	F7
Knirsch *et al.* [20]	Y	Y	Y	N	Y	Y	N
DiPrism [23]	N	N	Y	N	Y	Y	Y
PDAFT [17]	N	Y	N	N	Y	N	N
PPFA [21]	N	Y	N	N	Y	N	N
DG-APED [8]	N	N	Y	N	Y	Y	Y
FTMA [22]	Y	Y	N	N	N	N	N
Xue *et al.* [9]	Y	Y	Y	N	N	N	N
Our PPDA	Y	Y	Y	Y	Y	Y	Y

(1) F1: No Trusted Authority. F2: Accurate aggregated result. F3: Non-interactivity. F4: Aggregator failure is considered. F5: No additional computation cost for other SMs is required if a SM is faulty. F6: No additional computation cost for CC or TA is required if some SMs are faulty. F7: No additional communication overhead is required if some SMs are faulty.
(2) DiPrism [23] refers to its enhanced version.
(3) Y: Yes; N: No.

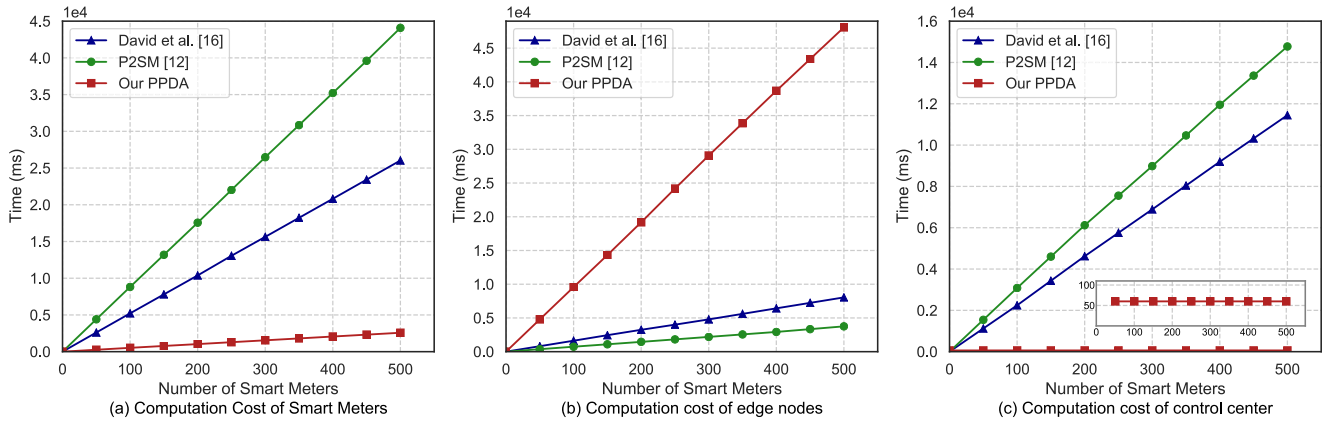


Fig. 6. Comparison of total computation cost.

communication lines. Each SM_i calculates a blinding factor. Then it simultaneously submits the masked value to the aggregator and its blinding factor to the successive SM. If SM_i fails, its blinding factor and masked value will both be excluded from data aggregation. Sending an acknowledgment for SM's availability will incur additional traffic flows and delays. DiPrism [23] allows the gateway to construct an equivalent ciphertext for correct decryption. However, its basic version cannot prevent the operation center from decrypting individual ciphertext, while its enhanced version will cause inaccurate aggregated results. In PDAFT[17], TA knows the private keys $x_i (1 \leq i \leq N)$. When SM_i malfunctions, CC requires TA to provide an equivalent ciphertext to decrypt correctly. This method needs interactivity with TA and suffers from delays. In PPFA [21], TA returns SM_i's blind factor $k_{i,j}$ back to EN_j when SM_i fails. To ensure the correct decryption, the connected EN_j's blind factor is updated as the negative sum of the remaining connected nodes. In DG-APED[8], CC discards the groups which contain at least a malfunctioning SM. Then it aggregates the normal groups. In FTMA[22], if SM_i breaks down, its cooperators must remove their shared key from their current blinding factor, making the remaining blinding factors rebalanced. Finally, all SMs upload the ciphertext again except SM_i. In Xue et al. [9], when SM_i breaks down, CC broadcasts a message to SM_i's building area networks and collects more than t shares of SM_i. Then CC reconstructs SM_i's static secret and its blind factor.

The existing fault tolerance schemes require either real-time online TA to deal with SM's dropouts[7], or some SMs to cooperatively calculate an equivalent ciphertext[14]. Our PPDA scheme achieves fault tolerance without imposing expensive computation on other smart meters or requiring additional trust assumptions.

(5) Communication overhead

The communication overhead is calculated in terms of the size of the message. (i) Communication overhead between different entities. We compare ours with Paillier-based[6] and Elliptic Curve-based[44] schemes. Only ciphertext overhead is considered because our PPDA does not need to exchange signatures among different entities. The aggregated ciphertexts and original individual ciphertexts in those three schemes remain the same formats, so communication overhead between $SM_i \rightarrow$ Edge and Edge $\rightarrow$ CC is identical, as shown in Table V.

TABLE V
COMMUNICATION OVERHEAD BETWEEN DIFFERENT ENTITIES

Schemes	Expression	Size	Description
Paillier-based [6]	$\log_2 N^2_{paillier}$	2*2048 =4Kb	an element under modulus $N^2_{paillier}$
Elliptic Curve-based [44]	$2 * \|P\|$	2*1024 =2Kb	a ciphertext with 2 points from elliptic curve
Our PPDA	$2 * n * \log_2 q$	2*512*56 =56Kb	a ciphertext with two polynomials

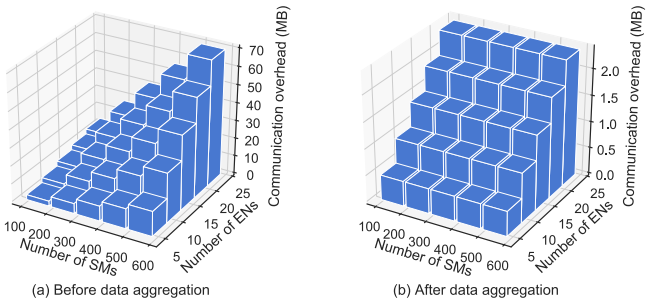


Fig. 7. Communication overhead within the edge.

(ii) Communication overhead within the edge. There are multiple edge nodes within the edge layer in our PPDA. Each SM's ciphertext will be distributed to every EN before data aggregation, $N\bar{n} * 56$Kb in total. After that, a public auditor will collect the aggregated ciphertext share and its signature from each EN for reconstruction, about $(\bar{n} + 1) * 56$Kb. Fig. 7 shows that total communication overhead grows approximately linearly as more SMs and ENs are added before aggregation, while it becomes irrespective of the number of SMs after aggregation. If we suppose there is a high-speed network within the edge nodes, the impact of data exchange between edge nodes is negligible.

Our PPDA from ideal lattices is still slightly larger than other public-key cryptosystems. Fortunately, edge computing could efficiently manage large ciphertext overhead by large-scale communication/storage capabilities. Therefore, our scheme is still at an acceptable level for smart grid.

VIII. Conclusion

This paper designs a homomorphic threshold proxy re-encryption with re-encryption verifiability (HTV-PRE). By combining HTV-PRE with a smart grid scenario, we give a robust and lightweight privacy-preserving data aggregation scheme. It provides cheap fault tolerance for both SMs and ENs, verifiable computation on aggregators, and flexible user dynamics. After being instantiated with ideal lattices, our PPDA is computationally lightweight, especially on the user side. Our work makes a meaningful attempt to design and implement a practical data aggregation scheme with privacy preservation for post-quantum IoT systems in smart grid.

Acknowledgments

The authors would like to thank the anonymous reviewers of IEEE TDSC whose comments significantly helped to improve the quality of this paper.

References

[1] P. Kumar, Y. Lin, G. Bai, A. Paverd, J. S. Dong, and A. Martin, "Smart grid metering networks: A survey on security, privacy and open research issues," *IEEE Commun. Surv.s Tut.*, vol. 21, no. 3, pp. 2886–2927, Third Quarter 2019.

[2] E. Quinn, "Privacy and the new energy infrastructure," *SSRN Electron. J.*, vol. 2, pp. 1–41, 2009.

[3] H. Bao and R. Lu, "A new differentially private data aggregation with fault tolerance for smart grid communications," *IEEE Internet Things J.*, vol. 2, no. 3, pp. 248–258, Jun. 2015.

[4] J. Zhao, T. Jung, Y. Wang, and X. Li, "Achieving differential privacy of data disclosure in the smart grid," in *Proc. IEEE Conf. Comput. Commun.*, 2014, pp. 504–512.

[5] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: Privacy preserving fog-enabled aggregation in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3733–3744, Aug. 2018.

[6] A. Saleem et al., "FESDA: Fog-enabled secure data aggregation in smart grid IoT network," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6132–6142, Jul. 2020.

[7] K. Xue et al., "PPSO: A privacy-preserving service outsourcing scheme for real-time pricing demand response in smart grid," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2486–2496, Apr. 2019.

[8] Z. Shi, R. Sun, R. Lu, L. Chen, J. Chen, and X. ShermanShen, "Diverse grouping-based aggregation protocol with error detection for smart grid communications," *IEEE Trans. Smart Grid*, vol. 6, no. 6, pp. 2856–2868, Nov. 2015.

[9] K. Xue, B. Zhu, Q. Yang, D. S. L. Wei, and M. Guizani, "An efficient and robust data aggregation scheme without a trusted authority for smart grid," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1949–1959, Mar. 2020.

[10] P. Gope and B. Sikdar, "Lightweight and privacy-friendly spatial data aggregation for secure power supply and demand management in smart grids," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 6, pp. 1554–1566, Jun. 2019.

[11] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "EPPA: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1621–1631, Sep. 2012.

[12] J. Ni, K. Zhang, X. Lin, and X. S. Shen, "Balancing security and efficiency for smart metering against misbehaving collectors," *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 1225–1236, Mar. 2019.

[13] S. Zhao et al., "Smart and practical privacy-preserving data aggregation for fog-based smart grids," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 521–536, 2021.

[14] L. Wu, M. Xu, S. Fu, Y. Luo, and Y. Wei, "FPDA: Fault-tolerant and privacy-enhanced data aggregation scheme in fog-assisted smart grid," *IEEE Internet Things J.*, vol. 9, no. 7, pp. 5254–5265, Apr. 2022.

[15] C. Huang, D. Liu, J. Ni, R. Lu, and X. Shen, "Reliable and privacy-preserving selective data aggregation for fog-based IoT," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.

[16] D. David, R. Sebastian, and S. Daniel, "Homomorphic proxy re-authenticators and applications to verifiable multi-user data aggregation," in *Financial Cryptography and Data Security*, K. Aggelos Ed., Berlin, Germany: Springer, 2017, pp. 124–142.

[17] L. Chen, R. Lu, and Z. Cao, "PDAFT: A privacy-preserving data aggregation scheme with fault tolerance for smart grid communications," *Peer-to-Peer Netw. Appl.*, vol. 8, no. 6, pp. 1122–1132, 2015.

[18] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, H. Gilbert, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–23.

[19] S. Ebrahimi and S. Bayat-Sarmadi, "Lightweight and fault-resilient implementations of binary ring-lwe for IoT devices," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6970–6978, Aug. 2020.

[20] F. Knirsch, G. Eibl, and D. Engel, "Error-resilient masking approaches for privacy preserving data aggregation," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3351–3361, Jul. 2018.

[21] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: Privacy preserving fog-enabled aggregation in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3733–3744, Aug. 2018.

[22] X. Wang, Y. Liu, and K. R. Choo, "Fault tolerant multi-subset aggregation scheme for smart grid," *IEEE Trans. Ind. Informat.*, vol. 17, no. 6, pp. 4065–4072, Jun. 2021.

[23] J. Ni, K. Zhang, K. Alharbi, X. Lin, N. Zhang, and X. S. Shen, "Differentially private smart metering with fault tolerance and range-based filtering," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2483–2493, Sep. 2017.

[24] T. Dimitriou and G. Karame, "Privacy-friendly tasking and trading of energy in smart grids," in *Proc. 28th Annu. ACM Symp. Appl. Comput.*, New York, NY, USA: Association for Computing Machinery, 2013, pp. 652–659, doi: 10.1145/2480362.2480488.

[25] C.-I. Fan, S.-Y. Huang, and Y.-L. Lai, "Privacy-enhanced data aggregation scheme against internal attackers in smart grid," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 666–675, Feb. 2014.

[26] D. He, N. Kumar, S. Zeadally, A. Vinel, and L. T. Yang, "Efficient and privacy-preserving data aggregation scheme for smart grid against internal adversaries," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2411–2419, Sep. 2017.

[27] Y. Ding, B. Wang, Y. Wang, K. Zhang, and H. Wang, "Secure metering data aggregation with batch verification in industrial smart grid," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6607–6616, Oct. 2020.

[28] J. Zhang, Y. Zhao, J. Wu, and B. Chen, "LVPDA: A lightweight and verifiable privacy-preserving data aggregation scheme for edge-enabled IoT," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4016–4027, 2020.

[29] J. Liu, J. Weng, A. Yang, Y. Chen, and X. Lin, "Enabling efficient and privacy-preserving aggregation communication and function query for fog computing-based smart grid," *IEEE Trans. Smart Grid*, vol. 11, no. 1, pp. 247–257, Jan. 2020.

[30] Z. Liu et al., "EPMDA-FED: Efficient and privacy-preserving multidimensional data aggregation scheme with fast error detection in smart grid," *IEEE Internet Things J.*, vol. 9, no. 9, pp. 6922–6933, May 2022.

[31] A. Alsharif, M. Nabil, S. Tonyali, H. Mohammed, M. Mahmoud, and K. Akkaya, "EPIC: Efficient privacy-preserving scheme with EtoE data integrity and authenticity for ami networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3309–3321, Apr. 2019.

[32] K. Emura, H. Kimura, T. Ohigashi, T. Suzuki, and L. Chen, "Privacy-preserving aggregation of time-series data with public verifiability from simple assumptions and its implementations," *Comput. J.*, vol. 62, no. 4, pp. 614–630, 2019.

[33] J. Ni, K. Alharbi, X. Lin, and X. Shen, "Security-enhanced data aggregation against malicious gateways in smart grid," in *Proc. IEEE Glob. Commun. Conf.*, 2015, pp. 1–6.

[34] T. Dimitriou and M. K. Awad, "Secure and scalable aggregation in the smart grid resilient against malicious entities," *Ad Hoc Netw.*, vol. 50, pp. 58–67, 2016.

[35] X. Zhang, C. Huang, Y. Zhang, and S. Cao, "Enabling verifiable privacy-preserving multi-type data aggregation in smart grids," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 4225–4239, Nov./Dec. 2022.

[36] Y. Polyakov, K. Rohloff, G. Sahu, and V. Vaikuntanathan, "Fast proxy re-encryption for publish/subscribe systems," *ACM Trans. Privacy Secur.*, vol. 20, no. 4, pp. 1–31, 2017.

[37] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled fully homomorphic signatures from standard lattices," in *Proc. 47th Annu. ACM Symp. Theory Comput.*, 2015, pp. 469–477.

[38] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[39] D. Shaar, "Packed leveled fully homomorphic signatures from ideal lattices," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2018.
[40] Y. Polyakov, K. Rohloff, G. W. Ryan, and D. Cousins, "PALISADE lattice cryptography library user manual," Duality Technologies, 2022. [Online]. Available: https://gitlab.com/palisade/palisade-release
[41] D.Giry, Cryptographic key length recommendations, 2021. [Online]. Available: https://www.keylength.com/
[42] Z. Wang, "Identity-based verifiable aggregator oblivious encryption and its applications in smart grids," *IEEE Trans. Sustain. Comput.*, vol. 6, no. 1, pp. 80–89, Jan./Mar. 2021.
[43] Y. Liu, W. Guo, C. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3PDA) scheme for smart grid," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1767–1774, Mar. 2019.
[44] O. R. Merad Boudia, S. M. Senouci, and M. Feham, "Elliptic curve-based secure multidimensional aggregation for smart grid communications," *IEEE Sensors J.*, vol. 17, no. 23, pp. 7750–7757, Dec. 2017.
[45] G. Verma, P. Gope, N. Saxena, and N. Kumar, "CB-DA: Lightweight and escrow-free certificate-based data aggregation for smart grid," *IEEE Trans. Dependable Secure Comput.*, early access, doi: 10.1109/TDSC.2022.3169952.
[46] X. Zhang, L. You, and G. Hu, "An efficient and robust multidimensional data aggregation scheme for smart grid based on blockchain," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 4, pp. 3949–3959, Dec. 2022.
[47] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.
[48] A. Menzies, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, vol. 516. Boca Raton, FL, USA: CRC Press, 1997.

Liqiang Wu received the BS and MS degrees in information security from the Engineering University of PAP, in 2009 and 2012, respectively. He is currently working toward the PhD degree in the National University of Defense Technology. His research interests focus on information security and cryptography.

Shaojing Fu received the PhD degree in applied cryptography from the National University of Defense Technology, in 2010. During his doctoral studies, he also spent a year as a Joint Doctoral Student with the University of Tokyo. He is currently a professor with the College of Computer, National University of Defense Technology. His research interests include cryptography theory and application in cloud and mobile computing.

Yuchuan Luo received the PhD degree in computer science and technology from the National University of Defense Technology (NUDT), in 2019. He is currently a lecturer with the College of Computer of NUDT. His research interests focus on security and privacy in cloud and crowd sourcing.

Hongyang Yan received the PhD degree in computer science and technology from Nankai University, in 2019. Her research interests include information privacy in machine learning, federated learning, and cloud storage. She has published more than 30 papers in international conferences and journals, including Information Sciences, ACM TOIT, FGCS, Applied Soft Computing, etc. She has received the best paper awards at international conferences such as CSS 2019, and ProvSec 2021. She is an academic editor of the International Journal of Intelligent Systems. She also serves as program committee and publication chair of many international conferences, such as ML4CS 2022, DMBD 2021, ML4CS 2020.

Heyuan Shi received the BS degree in the school of information science and engineering, Central South University, Changsha, China, in 2015, and the PhD degree in School of Software, Tsinghua University, Beijing, China. His current research interests include software safety, machine learning and operating systems.

Ming Xu is currently a professor with the Computer Science Department, College of Computer, the National University of Defense Technology, China. His major research interests include mobile computing, wireless network, cloud computing, and network security.