Differentially-Private Data Aggregation over Encrypted Location Data for Range Counting Query

Taisho Sasada*†, Nesrine Kaaniche‡, Maryline Laurent‡, Yuzo Taenaka*, and Youki Kadobayashi*

*Nara Institute of Science and Technology, Nara, Japan
†Research Fellow of the Japan Society for the Promotion of Science
‡SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris
{sasada.taisho.su0, yuzo, youki-k}@is.naist.jp, {kaaniche.nesrine, Maryline.Laurent}@telecom-sudparis.eu

***Abstract*—Location data has the potential to uncover patterns of congestion and overcrowding during specific times of day and days of the week. By pooling location data across different organizations, valuable insights can be derived that would be challenging to obtain independently. For instance, combining binary flag data (0 or 1), such as hotel stays, medical histories, and purchase records, with location data can facilitate range counting to reveal stay trends and the prevalence of infectious diseases in each region. However, the practice of aggregating data from various organizations introduces a critical concern: privacy leakage. When organizations share their data for aggregation, there is a risk that sensitive information could be exposed. To address this privacy challenge, it is imperative to aggregate the data of each organization while preserving privacy, and to make it impossible to infer sensitive information. In this research, we introduce an innovative differentially-private data aggregation protocol, facilitating the analysis of range counting across various organizations while maintaining data encryption throughout the process. Our proposed protocol leverages Homomorphic Encryption to secure both flag data and location information, confidentially merging only shared records to generate a unified table. Subsequently, our approach introduces encrypted noise to the resulting table until Differential Privacy guarantees privacy protection, even upon decryption. However, applying differential privacy to encrypted data carries the risk of enabling adversaries to inject manipulated data at their discretion. To counteract the potential mixing of manipulated and encrypted data, we have developed an algorithm within our proposed protocol to validate the content of encrypted data.**

***Index Terms*—Data Aggregation, Differential Privacy, Homomorphic Encryption, Location Data, Data Privacy**

I. Introduction

Location data has the capability to reveal trends of traffic congestion and excessive occupancy during particular hours of the day and days of the week. We can retrieve the count of data items that fall within a specific range in location data, and this query is commonly referred to as a Range Counting Query (RCQ), which represents the most prevalent utilization of location data [1]–[3]. RCQ counts the sum of flag values[1] for each region and can identify spatial/temporal flag trends in user behavior for each region. However, location data collection and analysis require large databases and high-performance computing environments, which not all entities can afford. Cloud technology has solved this challenge. The spread of cloud computing, which provides large-capacity databases and high-performance computing environments instead of on-premise systems, enables data aggregation among organizations. By aggregating data from entities that can collect location data and entities that have the data containing flag values (flag data) to be analyzed with each other in the cloud, it is now possible to grasp spatial and temporal trends that could not be created by a single organization. Nevertheless, the utilization of personal location data is bound by stringent data protection regulations, such as the EU's General Data Protection Regulation (GDPR), necessitating the safeguarding of data privacy to avoid breaching these regulations.

Given that multiple entities share their data through data aggregation, the relationships among participating entities and the selection of privacy protection technologies that align with their needs become immensely important. Homomorphic Encryption (HE) allows arbitrary operations, including addition and multiplication, to be performed on encrypted data without requiring decryption. When employed in data aggregation, HE enables the extraction of only those records that match in both datasets (location data and flag data), all while maintaining encryption. By implementing this process within the data store (DS), They can effectively respond to RCQ from any DAs. However, if query results are repeatedly provided to the same DA under the application of HE to data aggregation, the DA can back-calculate the original data from the difference between query results [4]. In short, the idempotency of the query result leads to privacy leakage. As a technique capable of removing the idempotency of the query result, Differential Privacy (DP) was introduced by Dwork et al [5]. DP thwarts attempts to deduce original data by introducing noise into the query result even if same query results are produced. In the context of data aggregation, noise is incorporated into the aggregate results prior to presenting the query outcomes. This ensures that even if DAs submit same contents of queries, the noise introduced by DP prevents any inference regarding the original data. However, the DP requires the DA to provide the data once to the untrustworthy DS, even when DAs are uncertain about the proper protect their privacy on DS.

Since HE and DP by themselves cannot prevent privacy leaks in data aggregation, there are research on a combination

[1]Flag value is data that represents a category variable as 1 or 0 with one hot encoding. In the case of the flag in the product data, if "1" is registered, it means that the item was purchased.

of HE and DP that utilizes both techniques to protect the privacy of the original data provided by the data owner (DO) while preventing privacy leakage from query results [4], [6], [7]. The presumption here does not involve DAs actively uploading their data, but it's worth noting that there have been documented instances of attacks against location data aggregation [8]. If even a single adversary is present among the DAs, they can manipulate query outcomes with ease by introducing crafted data into the mix. This becomes problematic because, due to the encryption of query results and the inherent noise addition by DP, idempotency is not maintained. Consequently, any distortion in the results remains undetectable. In summary, when both HE and DP are employed simultaneously, it becomes challenging to identify whether the results have been tampered with by an adversary.

In this study, we design a data aggregation protocol that combines HE and DP to facilitate encrypted data aggregation from each organization. This approach enables the execution of RCQ without necessitating the decryption of original data. Through the proposed protocol, location data and flag data encrypted using HE are merged within the DS. This enables DAs to actively participate in data aggregation, ensuring mutual protection of their respective data privacy from the DS. Although only DAs receive the analysis results as query responses after data aggregation, the idempotency is not maintained in the query results by applying DP. Therefore, even if the DA send the query repeatedly, privacy does not leak from the difference between query results.

In cases where certain DAs may have malicious intent and attempt to upload crafted data to manipulate query results, Our method creates a formidable challenge due to the encryption of all data using HE alongside DP. Consequently, distinguishing manipulated data from query results becomes exceedingly hard. To address this complexity, we introduce a data poisoning verification (DPV) algorithm tailored for encrypted data. If the DPV algorithm detects an attack to data aggregation by DAs, it refrains from furnishing a query response, identifies the breach, and alerts about the crafed data, thereby preventing from attack to data aggregation. As a summary, our contribution is below:

- Providing the result of RCQ for DAs that cannot collect location data themselves.
- Detecting attack to data aggregation while all data is encrypted and notifies details of attack.
- Providing RCQ at a very fine granularity using large-scale location data collected from DOs.

The rest of this paper is as follows: In Section II, we first describe related work. In Section III, we explains the design of proposed aggregation protocol. In Section IV, we evaluate our protocol from the perspective of performance. Finally, we conclude this paper in the Section V.

II. Related Work

Chowdhury et al. [4] have integrated HE and DP to establish a secure framework for data collection and analysis within their study. In their system, the DOs encrypts and transmits the original data to the DS. The encrypted data is then aggregated by the data store. While the DS does decrypt the data during this process, it introduces noise to the ciphertext prior to decryption, thereby ensuring the application of DP. Consequently, the DS remains unaware of the actual values of the original data. The output of query responses is guaranteed to adhere to DP, thereby preventing the DAs from predicting the original data. Although Cryptϵ is applicable to the scenario in our study, it does not assume the uploading of data by DAs.

As a potential attack targeting location data aggregation, Zhao et al. [8] propose the concept of "Poisoning Attacks on Location Data Aggregation" (PALDA). In the PALDA framework, a malicious DAs (adversary) crafts specific locations, referred to as "poisoned locations," and transmits them to a DS with the intent of manipulating or poisoning the aggregation outcomes carried out by the DS. If the DS possesses foreknowledge regarding the expected pattern of received locations (for example, congested or underpopulated areas), it may be capable of detecting instances where an adversary submits conspicuously crafted locations. To bypass such detection, the PALDA technique employs a two-step process for data manipulation: generation and adjustment. Initially, the adversary generates a composite location according to their manipulation goal. Subsequently, the adversary computes the mean squared error between the composite point and the actual ground truth data, and then fine-tunes the composite point to appear more plausible while effectively distorting the aggregate results.

III. Proposed Aggregation Protocol

In this section, we present a location-counting query processing system that offers prompt responses to queries from DAs when integrated with HE and DP. Indeed, the proposed method combines GPS data and flag data encrypted by SHE and counts them by region, providing the result of differentially-private RCQ (DP-RCQ) while keeping both GPS data and flag data confidential. A Somewhat Homomorphic Encryption (SHE) scheme refers to a scheme that enables evaluation of any circuit for data encrypted up to a certain depth. The maximum depth depends on the encryption parameters chosen, and generally, selecting larger depth parameters incurs a significant performance penalty. In this study, the BFV scheme is utilized as the SHE to encrypt the input. Since all data is processed in encrypted form, DAs only need to send GPS data indexing information (e.g., cell phone number or user ID) and flag data to DS, enabling range counting without the ability to collect GPS data themselves. Moreover, to prevent data poisoning such as PALDA, we design a data poisoning verification (DPV) algorithm for encrypted data. This algorithm can detect data poisoning attacks and notify entities who join data aggregation. We describe the system model in Sect III-A and the DPV algorithm in Sect III-B, respectively.

A. System Model

Fig. 1 depicts an overview of our system model. In our system, we assume four entities; Cryptographic Service Server

(CSS), DOs, DS, and DAs. We assume that the CSS and DOs are trusted, DS follows the "honest but curious" mode, i.e., correctly perform the protocol while trying to infer the original GPS data owned by DOs. DAs are assumed to be malicious by at least one person. We provide a detailed description of each entity below.

- Cryptographic Service Server (CSS)
  It is responsible of the key managment process. They generate public and private keys for SHE and provide them to DAs and DOs. They also decrypt the results of applying DP received from the DS with the private key and return them to the DAs. We assume that CSS is trusted; It does not infer DO/DA's data.
- Data Owners (DOs)
  They participate in the proposed system and view the results of RCQ published by DAs. DOs encrypt their location data using the SHE public key received from CSS and upload it to DS. After uploading the data, they do not interfere with the system in any way. We define them as trusted and assumes mobile carrier as the profile.
- Data Store (DS)
  It have the ability to collect location data and the necessary environment to do so. DS receives SHE-encrypted data from DOs and DAs, combines the data of DOs data with the data of DAs, applies DP to the combined data, and transfers it to the CCS. Since DS only process SHE encrypted data, the original data is never viewed in cleartext. However, they try to infer DOs and DAs data from the received data. We set it to the "honest but curious model", and assume it as cloud service providers.
- Data Analysts (DAs)
  DAs send flag data to DS and attempt to receive RCQs coupled with GPS data. Then, malicious DAs are present among at least one DAs and have the same affiliation as the DAs. By receiving incorrect query results from CSS as the organization they belong to, they plan to degrade the quality of service provided by DAs and intentionally manipulate the service. For this purpose, they are capable of data poisoning. They only aim at manipulating query results and collapsing protocols.

The proposed system protects the privacy of GPS data owned by DOs and flag data owned by DAs from CSS and DS; CSS and DS can only store data protected by DP, and by guaranteeing DP in the results of location count queries after aggregation allowed to store query results in plain text. Only DOs and DAs hold the encryption keys, only they encrypt their data, and only CSS decrypts them. The procedure of the proposed method is shown below based on Fig. 1. In the following procedure, the sequence of processes from "1. Key Generation" to "9. Output" is our data aggregation protocol.

- ① *Key Generation*: CSS generates a SHE public key (pk). The generated (pk) is sent to the DOs, DAs, and DS.
- ② *Encrypt GPS Data*: DOs encode their GPS data $\boldsymbol{D}_{\text{GPS}}$ via quadkey and encrypt it using the (pk) received from the CSS and upload the encrypted data to the DS. Quadkey is a spatial data indexing technique, which split the whole world into a set of tiles and store the tile identifier for each data point.
- ③ *Encrypt Flag Data*: DAs encrypt their flag data $\boldsymbol{D}_{\text{flag}}$ using the (pk) received from the CSS and upload the encrypted data to the DS.
- ④ *Generate Verification Array*: DS generates an array to verify that the data received from DAs has not been crafted, and sends it to CSS.
- ⑤ *Decrypt Verification Array*: CSS verifies the contents of the verification array and detects data poisoning. If there is no crafted value in the array, CSS notifies it to DS; otherwise, CSS does not demand DP result from DS, and notifies data poisoning attack with DAs.
- ⑥ *Private Join*: DS joins the encrypted GPS data (DO's data) with the encrypted flag data (DA's data) using SHE and gets the encrypted result of RCQ $\boldsymbol{D}_{\text{RCQ}}$.
- ⑦ *Applying DP*: DS calibrates and adds noise to RCQ result using the Laplace mechanism without decryption followed by sending the DP-RCQ $\boldsymbol{D}_{\text{RCQ}}{}'$ to the CSS.
- ⑧ *Decryption*: CSS decrypts $\boldsymbol{D}_{\text{RCQ}}{}'$.
- ⑨ *Output*: CSS sends $\boldsymbol{D}_{\text{RCQ}}{}'$ to the DAs.

B. Detecting Poisoning Data (DPV) Algorithm

The proposed method designs an algorithm to enable analysis even when there is an adversary among DAs, i.e., when DS and DAs are mutually untrustworthy. Our algorithm verifies whether data has been poisoned without decrypting the encrypted flag data, and can detect and notify the presence of an adversary among DAs without returning a false query response.

To detect data poisoned by malicious DAs, we design an algorithm to verify that the encrypted input vector $\boldsymbol{x}_{\text{flag}}$ is binary. Algorithm 1 shows the overall process of verification algorithm via calculating inner/hadamard product and applying DP to the result of RCQ. First, we create a verification vector $\boldsymbol{v}_{\text{ver}}$ with all elements initialized to 0 and a verification vector $\boldsymbol{v}_{\text{one}}$ with all elements initialized to 1. Then we substitute $\boldsymbol{v}_{\text{ver}}$ for $\boldsymbol{v}_{\text{flag}}$ minus $\boldsymbol{v}_{\text{one}}$. By this computation, we obtain an array $\boldsymbol{v}_{\text{ver}}$. The method then computes the inner product $\boldsymbol{v}_{\text{flag}} \cdot \boldsymbol{v}_{\text{ver}}$ for data poisoning detection. If the malicious DAs craft at least one element of the flags array, the crafted element of the flags array $\boldsymbol{v}_{\text{flag}}$ and the corresponding index array $\boldsymbol{v}_{\text{ver}}$ element will also have values other than 0 and -1. Therefore, it is possible to detect poisoning by malicious DAs by calculating the inner product $\boldsymbol{v}_{\text{flag}} \cdot \boldsymbol{v}_{\text{ver}}$ and checking whether it is zero or not.

However, calculating the inner product only indicates data poisoning; it does not tell malicious DAs which element has been crafted. The index of the crafted element in the array indicates the interest of the malicious DAs and is also necessary information for identifying the malicious one in the DAs. To identify the index of the crafted element, we then compute the Hadamard product $\boldsymbol{v}_{\text{flag}} \bigodot \boldsymbol{v}_{\text{ver}}$ for identification of crafted element in array. Hadamard product is the product of arrays determined by taking a component-by-component product over arrays of the same size. If malicious DAs craft

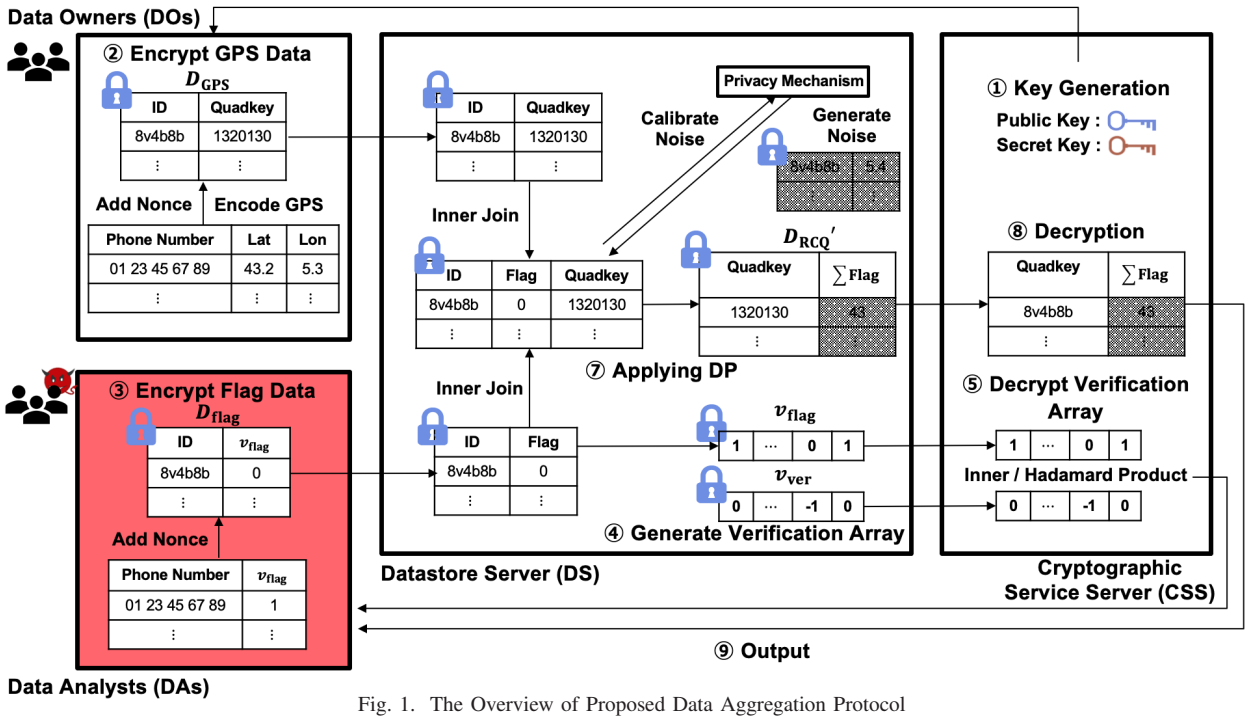

Fig. 1. The Overview of Proposed Data Aggregation Protocol

Algorithm 1 Verification Flag Array $\boldsymbol{v}_{\text{flag}}$ and Applying DP

Input: $\text{Enc}(\boldsymbol{v}_{\text{flag}})$
Output: $\boldsymbol{D}_{\text{RCQ}}{}'$ (or $\boldsymbol{v}_{\text{flag}} \bigodot \boldsymbol{v}_{\text{ver}}$)
DS : generate verification array $\boldsymbol{v}_{\text{one}}$ ($\boldsymbol{v}_{\text{one}} : [1, 1, \ldots, 1]$)
DA : $\boldsymbol{v}_{\text{ver}} \rightarrow$ **DS**
for $i \in n$ **do**
  DS : $\boldsymbol{v}_{\text{ver}}[i] \leftarrow \text{Enc}(\boldsymbol{v}_{\text{flag}}[i]) - \text{Enc}(\boldsymbol{v}_{\text{one}}[\text{i}])$
end for
if $\text{Enc}(\boldsymbol{v}_{\text{flag}} \cdot \boldsymbol{v}_{\text{ver}}) = 0$ **then**
  DS : generate appropriate noise η \\ Generate noise
  DS : $\boldsymbol{D}_{\text{RCQ}}{}' \leftarrow \text{Enc}(\boldsymbol{D}_{\text{RCQ}}) + \text{Enc}(\eta)$ \\ Add noise
  DS : $\boldsymbol{D}_{\text{RCQ}}{}' \rightarrow$ **CSS**
  CSS : $\boldsymbol{D}_{\text{RCQ}}{}' \rightarrow$ **DAs** \\ Return DP-RCQ result
else
  CSS : $\boldsymbol{v}_{\text{flag}} \bigodot \boldsymbol{v}_{\text{ver}} \rightarrow$ **DAs** \\ Notify data poisoning
end if

flag data, the value of the index corresponding to the crafted element in the Hadamard product $\boldsymbol{v}_{\text{flag}} \bigodot \boldsymbol{v}_{\text{ver}}$ of the flags array and the verification array becomes non-zero. If there is no data poisoning, all elements in the flag array are binary, so the inner product $\boldsymbol{v}_{\text{flag}} \cdot \boldsymbol{v}_{\text{ver}}$ is zero, and the Hadamard product $\boldsymbol{v}_{\text{flag}} \bigodot \boldsymbol{v}_{\text{ver}}$ is also an array of all zero elements. In the malicious case, on the other hand, at least one element of the array has been crafted, so the inner product is never a zero and the element of Hadamard product is not all zero.

C. Applying Differential Privacy

This section describes how to apply DP to encrypted data. DP protects data privacy by calibrating and adding noise. There is a trade-off between the strength of privacy protection and the value of differentially-private data. Increasing the noise amount enhances protection strength, but it also increases the deviation of differentially-private data from the original data, resulting in decreased value. However, the data received from DOs/DAs and the results of private inner joins are encrypted in our protocol, thus the DS cannot calibrate the noise amount. Naive methods that simply use the privacy mechanism to guarantee DP spoil the data value by producing more than minimal errors

In this protocol, the DS and CSS work together to calibrate the noise. The private inner join result $\boldsymbol{D}_{\text{RCQ}}$ and the number of users in quadkey are transferred to CSS, which divides $\boldsymbol{D}_{\text{RCQ}}$ by the number of users and divides it equally. The DS calibrates noise from the equally divided data. We use the Laplace mechanism as a privacy mechanism, which is a function that adds a random value to its input to satisfy DP. The Laplace mechanism calibrates noise from the Laplace distribution with zero mean, represented as $\mathcal{M}'_{\text{LAP}}(\boldsymbol{D}) = q(\boldsymbol{D}) + r$. Here, q denotes a query, r is sampled from $\text{Lap}\left(\frac{\Delta_q}{\epsilon}\right)$, and Δ_q represents the sensitivity of query q. The DS adds noise to $\boldsymbol{D}_{\text{RCQ}}$ by Laplace mechanism, obtains $\boldsymbol{D}_{\text{RCQ}}{}'$, and transfers it to the CSS. Finally, the CSS decrypts the query result $\boldsymbol{D}_{\text{RCQ}}{}'$ and returns it to DAs.

D. Security Analysis

We describe the security assumptions of our system.

1) We assume that DOs and CSS are trusted. CSS issues pk to DOs according to our proposed protocol, and DOs encrypt and transmit data using the received pk.
2) DS is assumed to be "honest but curious". They collect data according to the proposed protocol, but try their

best to steal the original $\boldsymbol{D}_{\text{GPS}}$ and $\boldsymbol{D}_{\text{flag}}$ owned by DOs and DAs.

3) There is at least one malicious (adversary) among the DAs, and they cannot be trusted by DOs, DS, and CSS.

The proposed protocol protects the privacy of $\boldsymbol{D}_{\text{GPS}}$ owned by DOs and $\boldsymbol{D}_{\text{flag}}$ owned by DAs. The original data is encrypted by SHE and sent to DS, which naturally cannot check the original data. DS then asks CSS to verify that the received data has not been crafted. If the data integrity is maintained, DS joins the $\boldsymbol{D}_{\text{GPS}}$ and $\boldsymbol{D}_{\text{flag}}$ in encrypted form without knowing the contents of the received data, applies DP, and sends the $\boldsymbol{D}_{\text{RCQ}}{}'$ to CSS (if the data has been edited, the protocol halts at that point). The CSS can decrypt $\boldsymbol{D}_{\text{RCQ}}{}'$, but since DS applies DP before decryption, the CSS cannot deduce the original data ($\boldsymbol{D}_{\text{GPS}}$ or $\boldsymbol{D}_{\text{flag}}$) from the $\boldsymbol{D}_{\text{RCQ}}{}'$ no matter how many times they receive and decrypt the $\boldsymbol{D}_{\text{RCQ}}{}'$.

IV. Evaluation

We evaluate the execution time and memory usage. To measure these performance on our protocol, we implement all program on ASRockRack 3U8G+/C621E workstation, CPU is 40-core Intel Xeon Gold 6230 Processor at 2.10 GHz, 262 GB RAM, and the host OS is Ubuntu 18.04 LTS. To implement homomorphic operations, we use the SEALv3.6 library, a fast and actively developed open source library maintained by Microsoft Research. The plaintext modulus $\log_2(p)$ of the BFV parameter is set to 42. We evaluate the overhead with the privacy parameter $\epsilon = 1$, which determines the strength of DP protection. However, our proposal is not influenced by the value of ϵ. We perform experiment by generating a random number representing quadkey (e.g. 1320130) as the location vector and a random number representing the phone number (e.g. 01-12-34-56-7) as the user identifier.

A. Execution Time and Data Size

To verify the feasibility of the proposed method, we measure the execution time until the data poisoning verification process is completed by the DPV algorithm by inner joining encrypted $\boldsymbol{D}_{\text{GPS}}$ and $\boldsymbol{D}_{\text{flag}}$ using SHE. Since the execution time depends on the number of DOs (size of $\boldsymbol{D}_{\text{GPS}}$) and the granularity of $\boldsymbol{D}_{\text{GPS}}$ (the domain size of SHE), we measure the execution time by changing these two parameters. Since multiple entities are involved in the proposed aggregation protocol, we divide the execution time into three steps: encryption, data aggregation (including data poisoning verification), and decryption, in order to easily understand the load of each step.

The left side of the Table I shows the execution time for each process of encryption (Enc), aggregation (Agg), and decryption (Dec). While encryption is not affected by the # of users and grids, aggregation and decryption are greatly affected. In particular, (16000,65536) has four times as many users and grids as (8000,16384), and its aggregation speed is also approximately four times faster. However, when comparing (16000,65536) and (32000, 262144), the aggregation speed is about 8 times faster. In other words, an exponential growth trend can be read.

The right side of Table I provides the size of input/output data, the size of public key pk, the size of galois key gk, and the size of relinearization key rk. The pk serves as a fundamental element to achieve the required security level, while gk are indispensable for executing rotation operations in homomorphic computations. Here, rk play a critical role in transforming the outcomes of ciphertext-ciphertext multiplications into a linear structure. The data size remains the same for pk, gk, and rk, while only the output is affected by the number of users/grids. Calculating the throughput from the execution time and data size, we get (8000, 16384) : 2.63Mbps, (16000, 65536) : 690.81kBps, (32000, 262144) : 83.85kBps, respectively.

B. Memory Usage

We measure the memory usage to show the load on the workstations. Since memory usage also varies greatly with the amount of data to be computed, i.e., the number of DOs and the granularity of $\boldsymbol{D}_{\text{GPS}}$, we measure the change for each encryption, data aggregation, and decryption. In order to focus only on processes related to the proposed protocol, we use htop, a process viewer for Linux. Fig. 3 shows the memory usage (MB) for patterns where we varied the number of users/quadkey grids (collection granularity). The maximum memory usage is measured for each process, and the numbers in the squares indicate means of memory usage. One characteristic that was common to all of the patterns was that the maximum and average values for encryption were of the same magnitude for all patterns. It can be seen that the speed of encryption is not proportional to the number of quadkey grids. On the other hand, data aggregation is quite different for each pattern, showing a proportional increase with the number of users and grids.

C. Data Value

We also conduct evaluation experiments to evaluate that proposed protocol can preserve data value before and after applying DP. We focused on the experimental results on the pattern (b) (# of user = 16000, # of grid = 65536) in Sect IV-B and set parameter. Figure 6 shows the mean absolute error before and after applying DP to the result of RCQ for each privacy budget ϵ. In the logarithmic scale graph, the quartile ranges in each ϵ are approximately equal, and it can be seen that the mean error decreases almost linearly. When the ϵ was small, the infrequent locations (i.e., underpopulated areas) in Quadkey had values that were quite different from the original locations due to noise.

D. Discussion

The proposed protocol has some limitations. First, the protocol cannot guarantee that DOs will use truthful data in the first place. For example, if the DOs are malicious, there could be cases where they craft $\boldsymbol{D}_{\text{GPS}}$ and send it to the DS. In that case, the proposed protocol guarantees the privacy of these wrong inputs, but the query results returned to the DAs will be distorted. Another case could be where DAs craft in the binary

TABLE I
EXECUTION TIME AND DATA SIZE

(# of user, # of grid)	Execution Time (Sec) Enc	Agg	Dec	**Total**	Data Size (MB) Input	Output	pk	gk	rk	**Total**
(8000, 16384)	11.039	213.518	0.482	225.039	0.913	0.896	1.028	584.579	8.228	595.646
(16000, 65536)	11.597	850.361	3.725	865.683	0.913	3.587	1.028	584.582	8.228	598.340
(32000, 262144)	13.452	7204.469	88.943	7306.864	0.913	17.935	1.028	584.584	8.228	612.688

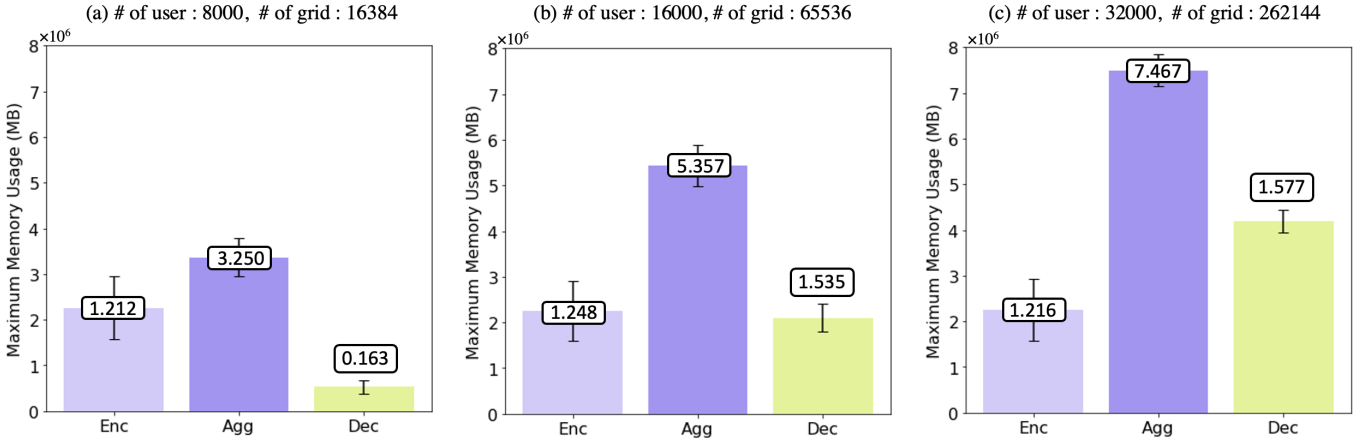

Fig. 2. Memory Usage

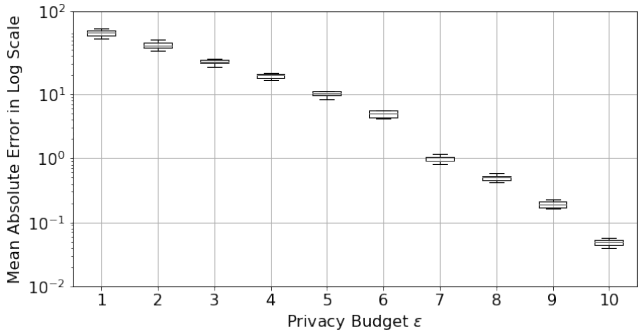

Fig. 3. Data Value

range (e.g., editing 0 to 1) rather than extreme data poisoning. In this case, the DPV algorithm included in the proposed protocol will overlook it because the input data is binary, but the query results will be distorted due to crafted data. This reliance on the truthfulness of input data is unfortunately a general problem for any computation (plaintext and privacy protection) and cannot be prevented by cryptographic means. We therefore recommend the use of "bullet proof", a zero-knowledge proof that guarantees that the sum or product of data falls within a certain range while concealing input data.

V. CONCLUSION

In this paper, we introduced a novel approach to differentially-private data aggregation, harnessing the power of CDP to operate on encrypted data. It also emphasize the capability to perform RCQ for DAs that might not possess access to GPS data. Unlike the existing systemq, which lack provisions for DAs to submit their data for RCQ and are consequently vulnerable to data poisoning attacks, our data aggregation protocol has successfully thwarted such attacks, thanks to the incorporation of a robust DPV algorithm that rigorously validates the uploaded data. Our experiments have evaluated factors like throughput, memory load, execution time, and message size, shedding light on the incurred overhead. Future endeavors will delve into devising methodologies for preemptively addressing data poisoning attacks, where DAs manipulate binary data.

ACKNOWLEDGEMENTS

This work was supported in part by Japan Society for the Promotion of Science KAKENHI Grant Number JP22J23910, and IPA's ICS-CoE Program.

REFERENCES

[1] Y. Yan, X. Gao, A. Mahmood, T. Feng, and P. Xie, "Differential Private Spatial Decomposition and Location Publishing based on Unbalanced Quadtree Partition Algorithm," *IEEE Access*, vol. 8, pp. 104775–104787, 2020.

[2] H. Mahdikhani, R. Lu, Y. Zheng, J. Shao, and A. A. Ghorbani, "Achieving $\mathcal{O}$ ($\log^3$n) Communication-Efficient Privacy-Preserving Range Query in Fog-Based IoT," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5220–5232, 2020.

[3] Z. Cai, X. Zheng, J. Wang, and Z. He, "Private Data Trading towards Range Counting Queries in Internet of Things," *IEEE Transactions on Mobile Computing*, 2022.

[4] A. Roy Chowdhury, C. Wang, X. He, A. Machanavajjhala, and S. Jha, "Cryptϵ: Crypto-Assisted Differential Privacy on Untrusted Servers," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pp. 603–619, 2020.

[5] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our Data, Ourselves: Privacy via Distributed Noise Generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 486–503, Springer, 2006.

[6] S. Ushiyama, T. Takahashi, M. Kudo, and H. Yamana, "Construction of Differentially Private Summaries Over Fully Homomorphic Encryption," in *Database and Expert Systems Applications: 32nd International Conference, DEXA 2021, Virtual Event, September 27–30, 2021, Proceedings, Part II*, pp. 9–21, Springer, 2021.

[7] J. Li, H. Ye, T. Li, W. Wang, W. Lou, Y. T. Hou, J. Liu, and R. Lu, "Efficient and Secure Outsourcing of Differentially Private Data Publishing with Multiple Evaluators," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 67–76, 2020.

[8] P. Zhao, H. Jiang, J. Li, Z. Xiao, D. Liu, J. Ren, and D. Guo, "Garbage in, garbage out: Poisoning Attacks Disguised with Plausible Mobility in Data Aggregation," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, pp. 2679–2693, 2021.