Authenticating Aggregate Range Queries over Dynamic Multidimensional Dataset

Jia Xu

National University of Singapore
Department of Computer Science
xujia@comp.nus.edu.sg

Abstract. We are interested in the integrity of the query results from an outsourced database service provider. Alice passes a set **D** of d-dimensional points, together with some authentication tag **T**, to an untrusted service provider Bob. Later, Alice issues some query over **D** to Bob, and Bob should produce a query result and a proof based on **D** and **T**. Alice wants to verify the integrity of the query result with the help of the proof, using only the private key. In this paper, we consider aggregate query conditional on multidimensional range selection. In its basic form, a query asks for the total number of data points within a d-dimensional range. We are concerned about the number of communication bits required and the size of the tag **T**. Xu and Chang [1] proposed a new method to authenticate aggregate count query conditional on d-dimensional range selection over static dataset, with $O(d^2 \log^2 N)$ communication bits, where N is the number of points in the dataset **D**. We extend their method to suport other types of queires, including summing, finding of the minimum/maximum/median and usual (non-aggregate) range selection, with similar complexity. Furthermore, dynamic operations, like insertion and deletion, over the outsourced dataset are also supported.

Keywords: Authentication, Multidimensional Aggregate Query, Secure Outsourced Database, Dynamic Database, Count, Sum, Average, Min, Max, Median, Range Selection

1 Introduction

Alice has a set **D** of d-dimensional points. She preprocesses the dataset **D** using her private key to generate some authentication tag **T**. She sends (outsources) **D** and **T** to an untrusted service provider Bob. Then Alice deletes the original copy of dataset **D** and tag **T** from her local storage. Later Alice may issue a query over **D** to Bob, for example, an aggregate query conditional on a multidimensional range selection, and Bob should produce the query result and a proof based on **D** and **T**. Alice wants to authenticate the query result, using only her private key. This problem fits in the framework of the outsourced database applications [2,3], which emerged in early 2000s as an example of "software-as-a-service" (SaaS).

We are concerned about the communication cost and the storage overhead on Alice/Bob's side. Such requirements exclude the following straightforward approaches: (1) Bob sends back the whole dataset **D** with its tag **T**; (2) Alice keeps a local copy of the dataset; (3) During preprocessing, Alice generates and signs answers to all possible queries.

Very recently, Xu and Chang [1] proposed a new method to authenticate aggregate count query over a static d-dimensional outsourced dataset, with $O(d^2 \log^2 N)$ communication bits, where N is the number of points in the dataset. Their method combined two customer designed primitives: (1) a **GKEA** (Generalized Knowledge of Exponent Assumption) based homomorphic authentication tag; (2) a functional encryption scheme supporting multidimensional range query. In this paper, we extend their method in two directions without sacrifying the communication complexity: (1) support other types of queries, including summing, finding of the minimum/maximum/median and usual (non-aggregate) range selection; (2) support dynamic operations like insertion and deletion over the outsourced dataset.

Table 1: Worst case performance of different authentication schemes for aggregate range query or range selection query. This table consists of two parts: the first three rows are for aggregate query; the rest four rows are for range selection query.
Note: (1) The symbol "-" indicates that the authors do not provide such information in their paper. (2) Our scheme is much more efficient in computation cost in 1D case, compared with high dimensional case (See annotation $\star$). (3) $dN \log N \le \log^d N$, if $d > \log(dN)/\log\log N$. We point out that the high computation cost on prover can be mitigated with horizontal partition of the dataset and parallel execution on each partition. (4) We do not include [4, 5] in this table, since these works do not provide concise asymptotic bound on their schemes. However, their performances are limited by the underlying data structure they adopted, i.e. KD-tree [4] and R-Tree [5], which require exponential (in dimension) communication overhead in the worst case. (5) Our scheme supports private key verification, while the other works in this table support public key verification.

Scheme	Dimension d	Communication overhead (bits)	Storage overhead	Computation (Verifier Alice)	Computation (Prover Bob)	Query	Techniques
PDAS [6]	$d = 1$	$O(\lvert S\rvert \log N)$	$O(N)$	$O(\lvert S\rvert \log N)$	$O(\lvert S\rvert + K^2)$	SUM,COUNT	Aggregated commitment + Shamir's Secret-Sharing Scheme
Li *et al.* [7]	$d \ge 1$	$O(dN + 2^d)$	$\Omega(dN)$	$O(dN + 2^d)$	$\Omega(N^{1-\frac{1}{d}})$	SUM or COUNT or MIN or MAX (*One authentication data structure per query type*)	MHT-like authentication structure for B-Tree/R-Tree
This paper and Xu *et al.* [1]	$d \ge 1$	$O(d^2 \log^2 \mathcal{Z})$	$O(dN)$	$O(d^2 \log^2 \mathcal{Z})\dagger\star$	$O(dN \log \mathcal{Z})\ddagger\star$	SUM,COUNT,MIN,MAX, MEDIAN	(customer designed) functional encryption + **GKEA** based homomorphic tag
Atallah *et al.* [8]	$d = 1, 2$	$O(1)$	$O(N)$	$O(\lvert S\rvert)$	$O(1)$	Range Selection	Precomputed prefix sum + **BLS** signature
Martel *et al.* [9]	$d \ge 1$	$O(\log^{d-1} N + \lvert S\rvert)$	-	-	-	Range Selection	Authentication Data Structure + Geometry Partition
Chen *et al.* [10]	$d \ge 1$	$O(\log^d \mathcal{Z})$	$O(N \log^d \mathcal{Z})$	$O(\log^d \mathcal{Z})$	$O(\log^d \mathcal{Z})$	Range Selection	Authentication Tree Structure + Access Control
This paper	$d \ge 1$	$O(d^2 \log^2 \mathcal{Z})$	$O(dN)$	$O(d^2 \log^2 \mathcal{Z} + \lvert S\rvert)\dagger\star$	$O(dN \log \mathcal{Z} + \lvert S\rvert)\ddagger\star$	Range Selection	(customer designed) functional encryption + **GKEA** based homomorphic tag

N: The number of tuples in the dataset.
K: The number of servers in PDAS [6].
$\dagger$: $O(d^2 \log^2 \mathcal{Z})$ group multiplications.
$\star$: If the query range is 1D, the cost is $O(\lvert S\rvert)$.
S: The set of tuples satisfying the query condition.
$\mathcal{Z}$: The domain size of attributes/points in one dimension.
$\ddagger$: $O(dN \log \mathcal{Z})$ bilinear map operations.

1.1 Contribution

The main contribution of this paper can be summarized as below.

1. We propose a method to authenticate aggregate queries over static multidimensional dataset, including SUM, MIN, MAX, MEDIAN, with $O(d^2 \log^2 \mathcal{Z})$ communication bits, based on [1]. We prove that the new authentication method is secure.
2. We propose a method to authenticate range selection query over multidimensional static dataset, with $O(d^2 \log^2 \mathcal{Z})$ communication bits, based on [1]. We prove that the new authentication method is secure.
3. We propose a method to authenticate aggregate range query and non-aggregate range selection query over dynamic multidimensional dataset. We prove that the proposed metho is secure.
4. We extend our method to support privacy protection and prevent frame attack.

The comparison between our result and previous work is given in Table 1.

2 Related work

Researches in secure outsourced database focus on two major aspects: (1) privacy (i.e. protect the data confidentiality against both the service provider and any third party) e.g. [3, 11, 12, 13], and (2) integrity (i.e. authenticate the soundness and completeness of query results returned by the service provider) e.g. [2, 9, 14, 15, 16, 4, 17, 5, 18, 19, 8, 20, 21, 22, 23, 6, 7]. In the "integrity" track, a lot of works (e.g. [9, 15, 16, 4, 8, 5]) are done for "identity query" [17], i.e. the query result is a subset of the database. [15, 4] authenticated 1D range selection queries, with linear (in the number of tuples selected by the query condition) communication cost and storage overhead. [16] verified range selection queries using aggregated signatures (like RSA [24], BLS [25]). [5] proposed a linear (or superlinear) scheme, which uses chained signatures over a "verification R-Tree" built on a multidimensional data space, to authenticate windows query, range query, kNN query, and RNN query. To the best of our knowledge, the current most efficient authentication scheme for range selection queries is [8], which proposed an efficient authentication scheme for 1D and 2D range selection queries over a grid dataset (e.g. GIS or image data) with $O(1)$ communication cost and linear storage overhead. [17] claimed to authenticate arbitrary queries, but their security model is too weak: a playful adversary can easily break their scheme. Aggregate range query is arguably more challenging and only a few works (e.g. [4, 6, 7, 1]) are devoted to the authentication of aggregate query. We remark that our scheme can also protect privacy for aggregate attributes by using homomorphic encryption scheme like [6, 7]).

There are roughly four categories of approaches for outsourced database authentication in the literatures [2, 9, 14, 15, 16, 4, 17, 5, 18, 19, 8, 20, 21, 22, 23]. (1) (Homomorphic and/or aggregatable) Cryptographic primitives, like collision-resistant hash, digital signature, commitment [16, 26, 6]. (2) Geometry partition and authenticated data structure [9, 5, 8, 21, 18, 7]. For example, Merkle Hash Tree (typically for 1D case) and variants, KD-tree with chained signature [4], R-Tree with chained signature [5], and authenticated B-Tree/R-Tree [7]. (3) Authenticated precomputed partial result, e.g. authenticated prefix sum [8, 7] (the static case solution in [7]) (4) Inserting and auditing fake tuples [19]. Instead of leveraging on the standard or existing cryptographic primitives (e.g. digital signature scheme, cryptographic hash) like most of previous works, [1] designed a new functional encryption scheme, and used it to construct a new homomorphic authentication tag. Consequently, [1] achieves very good asymptotic performance, but their proof of security became much more challenging.

To the best of our knowledge, the existing few works (e.g. [4, 6, 7]) on authentication of aggregate query either only deal with 1D case, or have communication overhead[1] linear (or even superlinear) w.r.t. the number of data points in the query range, and/or exponential in dimension. Even for multidimensional (non-aggregate) range selection query, the communication overhead is still in $O(\log^{d-1} N + |S|)$ (Martel *et al.* [9], Chen *et al.* [10]), where S is the set of data points within the query range, N is the number of data points in the dataset, and d is the dimension.

Recently, Gennaro *et al.* [27] and Chung *et al.* [28] proposed methods to authenticate *any* outsourced (or delegated) function, based on fully homomorphic encryption [29, 30, 31]. They [27, 28] also gave a good discussion on why previous techniques (e.g. interactive proofs, probabilistic checkable proof (PCP), and interactive arguments) are insufficient for authenticating outsourced function from the performance point of view. Without considering the efficiency (particularly ciphertext expansion) of fully homomorphic encryption scheme, Gennaro *et al.* [27] has communication cost which is sublinear w.r.t. the number of points in a dataset and polynomial in dimension, to authenticate aggregate range query. Our work is different in at least these aspects: (1) They authenticate a much more generic class of functions, while our techniques only deal with some aggregate range query (like counting, summing, finding of maximum or minimum or median) and non-aggregate range selection query. (2) Besides integrity authentication, Gennaro *et al.* [27] even provides privacy protection of the outsourced data against the worker (corresponding to Bob in our formulation). (3) Gennaro *et al.* [27] and Chung *et al.* [28] leverage on fully homomorphic encryption scheme [29]. (4) To deal with aggregate range query over outsourced database, both Gennaro *et al.* [27] and Chung *et al.* [28] have to treat the whole database as a single big chunk of data, so the completeness can be easily guaranteed at the cost of efficiency. We adapt a different approach: We bind each data point with an independent random number of special structure, and force Bob to process the dataset along with these random numbers in an inseparable manner. Then we can verify the consistency between the returned query result and the associated randomness.

[1] The original papers either do not provide a tight theoretical asymptotic bound, or do not relate the bound to generic parameters, including database size, domain size, dimension and security parameter.

In this way, we can achieve better[2] complexity than the two generic methods. However, our approach requires more serious attention to deal with the completeness issue. As Gennaro *et al.* [27] pointed out, it is meaningful to design more efficient authentication scheme[3] without using fully homomorphic encryption scheme, even at the cost of sacrificing privacy of outsourced data.

3 Formulation

In this section, we restate the problem formulation and security model from [1], with modifications adapting our extension in this paper.

3.1 Dataset and Query

The dataset $\mathbf{D}$ is a set of N d-dimensional points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ from the domain $[\mathcal{Z}]^d$ where $\mathcal{Z}$ is a big integer (e.g. 64 bits integer). Each point $\boldsymbol{x} \in \mathbf{D}$ is associated with a vector-valued attribute, denoted as $\mathsf{Att}(\boldsymbol{x})$, where each component of the vector $\mathsf{Att}(\boldsymbol{x})$ is an integer. Let $\mathbf{R} = [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_d, b_d] \subseteq [\mathcal{Z}]^d$ be a d-dimensional rectangular range. Xu and Chang [1] focused on aggregate count query function $\textsc{Count}$:

$$\textsc{Count}(\mathbf{D}, \mathbf{R}) \overset{\text{def}}{=} \sum_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{Att}(\boldsymbol{x}) \pmod{p},$$

where the attribute $\mathsf{Att}(\boldsymbol{x}) = 1$ for each point $\boldsymbol{x} \in \mathbf{D}$ is an extreme case, and p is a large prime. Note that p is exponential in the security parameter κ and N is polynomial in κ. In practice, p could be a 400 bits prime and N could be arround a million.

In this paper, we are concerning these queries together with multidimnsional vector-valued attribute $\mathsf{Att}(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbf{D}$:

$\textsc{Sum}$**:** A sum query with range $\mathbf{R}$ asks for the summation of attributes $\mathsf{Att}(\boldsymbol{x})$ for all data points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$.

$$\textsc{Sum}(\mathbf{D}, \mathbf{R}) = \bigoplus_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{Att}(\boldsymbol{x}) \pmod{p} \tag{1}$$

$\textsc{Min}$**:** A min query with range $\mathbf{R}$ and dimension $\iota \in [d]$ asks for the minimum attribute value along the ι-th dimension among all data points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$.

$$\textsc{Min}(\mathbf{D}, \mathbf{R}, \iota) = \min_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{Att}(\boldsymbol{x})[\iota] \tag{2}$$

$\textsc{Max}$**:** A max query with range $\mathbf{R}$ and dimension $\iota \in [d]$ asks for the maximum attribute value along the ι-th dimension among all data points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$.

$$\textsc{Max}(\mathbf{D}, \mathbf{R}, \iota) = \max_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{Att}(\boldsymbol{x})[\iota] \tag{3}$$

$\textsc{Median}$**:** A median query with range $\mathbf{R}$ and dimension $\iota \in [d]$ asks for the meadian attribute value along the ι-th dimension among all data points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$.

$$\textsc{Median}(\mathbf{D}, \mathbf{R}, \iota) = y, \text{ such that } y \in S = \{\mathsf{Att}(\boldsymbol{x})[\iota] : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\} \text{ and } y \text{ is ranked } \lceil \frac{|S|}{2} \rceil\text{-th among the set } S \tag{4}$$

$\textsc{RangeSelect}$**:** A range select query with range $\mathbf{R}$ asks for all data points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$.

$$\textsc{RangeSelect}(\mathbf{D}, \mathbf{R}) = \{\boldsymbol{x} : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\} \tag{5}$$

[2] Ciphertext expansion is not the only reason of communication overhead and storage overhead for Gennaro *et al.* [27] and Chung *et al.* [28]. Gennaro *et al.* [27] assigns each bit of data with a large random number and encrypts this random number using fully homomorphic encryption. Chung *et al.* [28] has to transmit κ number of ciphertexts of similar queries/results to process a single query. The difference between their solutions and our work may become more clear when authenticating non-aggregate range selection query: Both Gennaro *et al.* [27] and Chung *et al.* [28] will require linear or even superlinear communication overhead, while our solution sill requires $O(d^2 \log^2 \mathcal{Z})$ communication cost.

[3] Although our solution only supports a very small range of functions.

3.2 Security Model

Xu and Chang [1] presented a formulation for the authentication problem over outsourced database, as a special case of *Verifiable Computation* [27] particular for query functions over a outsourced dataset. Let us view a (generic) query on a dataset as the function $F : \mathbb{D} \times \mathbb{Q} \rightarrow \{0,1\}^*$, where $\mathbb{D}$ is the domain of datasets, $\mathbb{Q}$ is the domain of queries, and the output of F is represented by a binary string. Note that a query $Q \in \mathbb{Q}$ is represente by combination of query type (like count, sum, etc), query range, and other parameters if any (e.g. a min query $\textsc{Min}(\mathbf{R}, \iota)$). Xu and Chang [1] defined the remote computing protocol as follow:

Definition 1 ($\mathcal{RC}$) *A* Remote Computing *($\mathcal{RC}$) protocol for a function $F : \mathbb{D} \times \mathbb{Q} \rightarrow \{0,1\}^*$, between Alice and Bob, consists of a setup phase and a query phase. The setup phase consists of a key generating algorithm* KGen *and data encoding algorithm* DEnc*; the query phase consists of a pair of interactive algorithms, namely the evaluator* Eval *and the extractor* Ext*. These four algorithms* $(\mathsf{KGen}, \mathsf{DEnc}, \langle\mathsf{Eval}, \mathsf{Ext}\rangle)$ *run in the following way:*

Setup Phase *:*

1. *Given security parameter κ, Alice generates a key K: $K \leftarrow \mathsf{KGen}(1^\kappa)$.*
2. *Alice encodes dataset $\mathbf{D} \in \mathbb{D}$: $(\mathbf{D}_{\mathtt{B}}, \mathbf{D}_{\mathtt{A}}) \leftarrow \mathsf{DEnc}(\mathbf{D}, K)$, then sends $\mathbf{D}_{\mathtt{B}}$ to Bob and keeps $\mathbf{D}_{\mathtt{A}}$.*

Query Phase *: The query phase consists of multiple query sessions. In each query session, Alice and Bob interact as below.*

1. *Alice selects a query $\mathbf{Q} \in \mathbb{Q}$.*
2. *Algorithm $\mathsf{Ext}(\mathbf{D}_{\mathtt{A}}, \mathbf{Q}, K)$ on Alice's side, interacts with algorithm $\mathsf{Eval}(\mathbf{D}_{\mathtt{B}})$ on Bob's side to compute $(\zeta, X, \boldsymbol{\Psi}) \leftarrow \langle\mathsf{Eval}(\mathbf{D}_{\mathtt{B}}), \mathsf{Ext}(\mathbf{D}_{\mathtt{A}}, \mathbf{Q}, K)\rangle$, where $\zeta \in \{\mathtt{accept}, \mathtt{reject}\}$ and $\boldsymbol{\Psi}$ is the proof of result X. If $\zeta = \mathtt{reject}$, then Alice rejects. Otherwise, Alice accepts that X is equal to $F(\mathbf{D}, \mathbf{Q})$.*

Definition 2 (Efficient $\mathcal{RC}$ [1]) *A $\mathcal{RC}$ protocol is* efficient*, if*

1. *the size of K and $\mathbf{D}_{\mathtt{A}}$ are both in $O(poly(d))$ where d is the dimension of dataset $\mathbf{D}$;*
2. *communication complexity is $O(poly(d, \log|\mathbf{D}|))$;*
3. *the size of $\mathbf{D}_{\mathtt{B}}$ is $O(poly(d, |\mathbf{D}|))$ (this implies the complexity of* DEnc *is in $O(poly(d, |\mathbf{D}|))$).*
4. *the algorithm* Ext *must be more efficient than computing F directly (This is similar with models in [27, 28]).*

A $\mathcal{RC}$ protocol is *verifiable*, if the following conditions hold: (1) Alice always accepts, when Bob follows the protocol honestly; (2) Alice rejects with o.h.p. (overwhelming high probability), when Bob returns a wrong result. Here adversaries, i.e. malicious Bob, are allowed to interact with Alice and learn for polynomial number of query sessions, before launching the attack. During the learning, the adversary may store whatever it has seen or leant in a state variable.

Definition 3 ($\mathcal{VRC}$ [1]) *A $\mathcal{RC}$ protocol $\mathcal{E} = (\mathsf{KGen}, \mathsf{DEnc}, \langle\mathsf{Eval}, \mathsf{Ext}\rangle)$ w.r.t. function $F : \mathbb{D} \times \mathbb{Q} \rightarrow \{0,1\}^*$, is called $\mathcal{VRC}$ (*Provable Remote Computing*) protocol, if the following two conditions hold: Let κ be the security parameter.*

– *correctness: for any $\mathbf{D} \in \mathbb{D}$, any $K \leftarrow \mathsf{KGen}(1^\kappa)$ and any $\mathbf{Q} \in \mathbb{Q}$, it holds that $\langle\mathsf{Eval}(\mathbf{D}_{\mathtt{B}}), \mathsf{Ext}(\mathbf{D}_{\mathtt{A}}, \mathbf{Q}, K)\rangle = (\mathtt{accept},\ F(\mathbf{D}, \mathbf{Q}),\ \boldsymbol{\Psi})$ for some $\boldsymbol{\Psi}$, where $(\mathbf{D}_{\mathtt{B}}, \mathbf{D}_{\mathtt{A}}) \leftarrow \mathsf{DEnc}(\mathbf{D}, K)$.*
– *soundness: for any PPT (adaptive) adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}(1^\kappa) \leq negl(\kappa)$ (asymptotically less or equal).*

where $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}(1^\kappa)$ is defined as

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}(1^\kappa) \overset{\text{def}}{=} \mathsf{Pr}\left[\begin{array}{c}(\zeta, X, \boldsymbol{\Psi}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}}, \mathbf{D}, \mathbf{Q}) \leftarrow \mathsf{Exp}_{\mathcal{A}}^{\mathcal{E}}(1^\kappa) : \\ \zeta = \texttt{accept} \ \wedge \ X \neq F(\mathbf{D}, \mathbf{Q})\end{array}\right];$$

```
Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathcal{E}}(1^\kappa)$

$\mathbf{D} \leftarrow \mathcal{A}(\mathsf{view}_{\mathcal{A}}^{\mathcal{E}})$;
$K \leftarrow \mathsf{KGen}(1^\kappa)$;
$(\mathbf{D}_{\mathtt{B}}, \mathbf{D}_{\mathtt{A}}) \leftarrow \mathsf{DEnc}(\mathbf{D}, K)$;
loop until $\mathcal{A}(\mathsf{view}_{\mathcal{A}}^{\mathcal{E}})$ decides to stop
    $\mathbf{Q}_i \leftarrow \mathcal{A}(\mathbf{D}_{\mathtt{B}}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}})$;
    $(\zeta_i, X_i, \boldsymbol{\Psi}_i) \leftarrow \langle \mathcal{A}(\mathbf{D}_{\mathtt{B}}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}}), \mathsf{Ext}(\mathbf{D}_{\mathtt{A}}, \mathbf{Q}_i, K)\rangle$;
$\mathbf{Q} \leftarrow \mathcal{A}(\mathbf{D}_{\mathtt{B}}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}})$;
$(\zeta, X, \boldsymbol{\Psi}) \leftarrow \langle \mathcal{A}(\mathbf{D}_{\mathtt{B}}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}}), \mathsf{Ext}(\mathbf{D}_{\mathtt{A}}, \mathbf{Q}, K)\rangle$;
Output $(\zeta, X, \boldsymbol{\Psi}, \mathsf{view}_{\mathcal{A}}^{\mathcal{E}}, \mathbf{D}, \mathbf{Q})$.
```

The probability is taken over all random coins used by related algorithms, negl$(\cdot)$ is some negligible function, and $\mathsf{view}_{\mathcal{A}}^{\mathcal{E}}$ *is a state variable*[4] *describing all random coins chosen by $\mathcal{A}$ and all messages $\mathcal{A}$ can access during previous interactions with $\mathcal{E}$.*

We remark that this security model is also related to the formulation of $\mathcal{POR}$ (Proof of Retrievability) [32] and it is not surprising that our scheme could imply a (ρ, λ)-valid $\mathcal{POR}$ system, with some proper parameters ρ and λ.

4 Background

In this section, we summarize the authentication scheme for aggregate count query over static multidimensional dataset proposed by Xu and Chang [1], which serves as the base of this paper. For the sake of presentation of our extension in this paper, we make a very slight modification to the original scheme proposed by Xu and Chang [1].

Let $\mathbf{D}$ be a set of N d-dimensional points in domain $[1, \mathcal{Z}]^d$, and each point $\boldsymbol{x} \in \mathbf{D}$ is associated with an attribute $\mathsf{Att}(\boldsymbol{x})$. In [1], the attribute function is $\mathsf{Att}(\boldsymbol{x}) = 1$, since it dealed with COUNT query. This paper will redefine the attribute function later to deal with SUM query.

Their scheme is an interactive protocol between Alice and Bob, and contains a setup phase followed by a query phase. In the setup phase, Alice preprocesses the dataset by generating a tag $\mathsf{DTag}(\boldsymbol{x})$ for each point $\boldsymbol{x}$ in the dataset $\mathbf{D}$. At the end of setup phase, Alice sends both the dataset $\mathbf{D}$ and tags $\mathbf{T} = \{\mathsf{DTag}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbf{D}\}$ to Bob and removes them from her storage. Later in the query phase, Alice may issue many queries over the dataset to Bob. For example, Alice may want to know how many points are within a range $\mathbf{R}$. Alice sends $\mathbf{R}$ to Bob. Meanwhile, in order to help Bob to generate a proof, Alice chooses a random nonce ρ and sends $\Phi = \{\mathsf{QTag}(\boldsymbol{x}, \rho) : \boldsymbol{x} \in \mathbf{R}\}$ to Bob. After receiving $\mathbf{R}$ and Φ, Bob is supposed to return $X = |\mathbf{D} \cap \mathbf{R}|$ as result and $\Psi_1 = \otimes_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{DTag}(\boldsymbol{x})$ and $\Psi_2 = \otimes_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \mathsf{QTag}(\boldsymbol{x}, \rho)$ as proof. Since the tag functions $\mathsf{DTag}, \mathsf{QTag}$ are homomorphic, Alice can verify the consistency between Ψ_1 and Ψ_2 using a secret key. To ensure completeness, Alice has to interact with Bob and perform the above procedure for the complement query range $\mathbf{R}^{\mathsf{C}}$.

However, the size of Φ is propotional to the size of range $\mathbf{R}$, which could be huge. One of main contributions of [1] is that the paper proposed a new functional encryption scheme and use it to reduce communication cost in the following way:

- In the setup phase, Alice produce a ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ for each data point $\boldsymbol{x} \in \mathbf{D}$ using the functional encryption scheme. Alice sends all ciphertexts $\mathsf{CT}_{\boldsymbol{x}}$'s together with the dataset and tags to Bob at the end of setup.
- In a query session, for a count query with range $\mathbf{R}$, Alice chooses a random nonce ρ and generates a *short* delegation key $\boldsymbol{\delta}$ w.r.t. the query range $\mathbf{R}$ and the random nonce ρ, using the functional encryption scheme. Alice sends the delegation key $\boldsymbol{\delta}$ to Bob together with the query.

[4] The adaptive adversary $\mathcal{A}$ may keep updating this state variable.

– For each data point $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$, Bob can decrypt ciphertext $\mathsf{CT}_{\boldsymbol{x}}$ and obtain $\mathsf{QTag}(\boldsymbol{x}, \rho)$ as the decrypted value using the functional encryption scheme and the delegation key $\boldsymbol{\delta}$. For points $\boldsymbol{y} \notin \mathbf{D} \cap \mathbf{R}$, Bob learns nothing about $\mathsf{QTag}(\boldsymbol{y}, \rho)$.

Since the size of delegation key $\boldsymbol{\delta}$ is in $O(d \log^2 \mathcal{Z})$, the communication cost is reduced dramatically.

Xu and Chang [1] *implicitly* defined a homomorphic authentication tag $(\mathsf{DTag}, \mathsf{QTag}, \mathsf{Verify})$ as below: Let key $\mathcal{K} = (\theta, \beta, \gamma) \in \widetilde{\mathbb{G}} \times \mathbb{Z}_p^* \times \mathbb{Z}_p^*$, $v_{\boldsymbol{x}}, w_{\boldsymbol{x}} \in \widetilde{\mathbb{G}}$ be random coins chosen for point $\boldsymbol{x}$, and $\boldsymbol{\Psi} = (\Psi_1, \Psi_2, \Psi_3)$.

$$\mathsf{DTag}_{\mathcal{K}}(\boldsymbol{x}) = \left(\theta^{\mathsf{Att}(\boldsymbol{x})} v_{\boldsymbol{x}},\ v_{\boldsymbol{x}}^{\beta},\ w_{\boldsymbol{x}}\right) \tag{6}$$

$$\mathsf{QTag}_{\mathcal{K}}(\boldsymbol{x}, \rho) = v_{\boldsymbol{x}}^{\gamma} w_{\boldsymbol{x}}^{\rho} \tag{7}$$

$$\mathsf{Verify}_{\rho,\mathcal{K}}(Y, \boldsymbol{\Psi}, \Psi_4) = \begin{cases} 1 & \left(\text{if } \left(\Psi_1 \theta^{-Y}\right)^{\beta} = \Psi_2 \text{ and } \left(\Psi_1 \theta^{-Y}\right)^{\gamma} \Psi_3^{\rho} = \Psi_4\right) \\ 0 & \text{(otherwise)} \end{cases} \tag{8}$$

The authentication tag $(\mathsf{DTag}, \mathsf{QTag}, \mathsf{Verify})$ is homomorphic and satisfies the following properties:

$$\mathsf{Verify}_{\rho,\mathcal{K}}\left(\mathsf{Att}(\boldsymbol{x}),\ \mathsf{DTag}_{\mathcal{K}}(\boldsymbol{x}),\ \mathsf{QTag}_{\mathcal{K}}(\boldsymbol{x}, \rho)\right) = 1 \tag{9}$$

$$\mathsf{Verify}_{\rho,\mathcal{K}}\left(\sum_{\boldsymbol{x} \in \mathbf{R}} \mathsf{Att}(\boldsymbol{x}),\ \bigotimes_{\boldsymbol{x} \in \mathbf{R}} \mathsf{DTag}_{\mathcal{K}}(\boldsymbol{x}),\ \prod_{\boldsymbol{x} \in \mathbf{R}} \mathsf{QTag}_{\mathcal{K}}(\boldsymbol{x}, \rho)\right) = 1 \tag{10}$$

We restate the scheme in [1] in Figure 1 with authentication tag $(\mathsf{DTag}, \mathsf{QTag}, \mathsf{Verify})$ and hide details of the applications of the functional encryption scheme.

4.1 Security

Since the authentication tag function is homomorhpic, an adversary (i.e. a dishonest Bob) may attemp to cheat and convince Alice to accept a wrong result in this way: choose some integer $\mu_{\boldsymbol{x}}$ for each point $\boldsymbol{x}$, and in Step B1 of algorithm $\mathsf{CollRes}$ compute the proof $(\boldsymbol{\Psi}, \Psi_4)$ as below

$$\boldsymbol{\Psi} \leftarrow \bigotimes_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{t}_{\boldsymbol{x}}^{\mu_{\boldsymbol{x}}} = \bigotimes_{\boldsymbol{x} \in \mathbf{D}} \mathsf{DTag}(\boldsymbol{x})^{\mu_{\boldsymbol{x}}},\ \ \Psi_4 \leftarrow \prod_{\boldsymbol{x} \in \mathbf{D}} \mathsf{QTag}(\boldsymbol{x}, \rho)^{\mu_{\boldsymbol{x}}} \tag{13}$$

It is easy to verify that the above forged proof passes the verification in Step A2 of $\mathsf{CollRes}$ in Figure 1, but may not pass the second equality test in Step 3 of Count in Figure 1. Such adversary looks "restricted" in its attack strategy. However, [1] showed that, under **GKEA** assumption, such adversary's power is not restricted at all: If there exists an efficient (arbitrary) adversary that breaks their scheme, then there exists such "restricted" adversary that breaks their scheme.

[1] considered various types of PPT adversaries, which interacts with Alice by playing the role of Bob and intends to output a wrong query result and a forged but valid proof:

– Type I adversary: This adversary is not confined in any way in its attack strategy and produces a tuple $(X, \boldsymbol{\Psi} = (\Psi_1, \Psi_2, \Psi_3), \Psi_4)$ on a query range $\mathbf{R}$.
– Type II adversary: A restricted adversary which can produce the same forgery[5] from the same input as Type I adversary, additionally, it finds N integers[6] μ_i's, $1 \le i \le N$, such that

$$\boldsymbol{\Psi} \leftarrow \bigotimes_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{t}_{\boldsymbol{x}}^{\mu_{\boldsymbol{x}}} = \bigotimes_{\boldsymbol{x} \in \mathbf{D}} \mathsf{DTag}(\boldsymbol{x})^{\mu_{\boldsymbol{x}}},$$

[5] This is possible, if the Type II adversary just invokes Type I adversary as a subroutine using the same random coin.

[6] Note that μ_i can take negative integer value, and $\mu_i > 1$ ($\mu_i < 1$, respectively) corresponds to the case of double counting (undercounting, respectively) point $\boldsymbol{x}_i$.

Fig. 1: Construction of $\mathcal{RC}$ protocol $\mathcal{E} = (\mathsf{KGen}, \mathsf{DEnc}, \langle \mathsf{Eval}, \mathsf{Ext}\rangle)$ where $\langle \mathsf{Eval}, \mathsf{Ext}\rangle$ (namely Count) invokes $\langle \widetilde{\mathsf{Eval}}, \widetilde{\mathsf{Ext}}\rangle$ (namely $\mathsf{CollRes}$) as a subroutine. The attribute function is $\mathsf{Att}(\boldsymbol{x}) = 1$ for each $\boldsymbol{x} \in \mathbf{D}$.

(Alice) $\mathsf{KGen}(1^\kappa)$: Output a private key $\mathcal{K}$.

(Alice) $\mathsf{DEnc}(\mathbf{D}; \mathcal{K})$:

1. For each point $\boldsymbol{x} \in \mathbf{D}$, generate a tag
$$\boldsymbol{t}_{\boldsymbol{x}} = \mathsf{DTag}(\boldsymbol{x}, \mathcal{K}).$$
2. For each point $\boldsymbol{x} \in \mathbf{D}$, generate a ciphertxt $\mathsf{CT}_{\boldsymbol{x}}$, using the functional encryption scheme with key $\mathcal{K}$.
3. Send $\mathbf{D}_{\mathtt{B}} = (\mathbf{D}, \mathbf{T} = \{\boldsymbol{t}_{\boldsymbol{x}} : \boldsymbol{x} \in \mathbf{D}\}, \mathbf{C} = \{\mathsf{CT}_{\boldsymbol{x}} : \boldsymbol{x} \in \mathbf{D}\})$ to Bob, and keep *only* key $\mathcal{K}$ and $\mathbf{D}_{\mathtt{A}} = (N,\ d,\ \varDelta = \bigotimes_{\boldsymbol{x}\in\mathbf{D}} \boldsymbol{t}_{\boldsymbol{x}})$ in local storage.

(Alice, Bob) $\mathsf{Count} = \langle \mathsf{Eval}(\mathbf{D}_{\mathtt{B}}), \mathsf{Ext}(\mathbf{D}_{\mathtt{A}},\ \mathbf{R},\ \mathcal{K})\rangle$: $\mathbf{D}_{\mathtt{A}} = (N,\ d,\ \varDelta), \mathbf{D}_{\mathtt{B}} = (\mathbf{D}, \mathbf{T}, \mathbf{C})$
Precondition: The query range $\mathbf{R} \subset [\mathcal{Z}]^d$ is a rectangular range.

Step 1: Alice partitions the complement range $\mathbf{R}^{\mathsf{C}}$ into $2d$ rectangular ranges $\{\mathbf{R}_\ell \subset [\mathcal{Z}]^d : \ell \in [1, 2d]\}$, and sets $\mathbf{R}_0 = \mathbf{R}$.
Step 2—Reduction: For $0 \leq \ell \leq 2d$, Alice and Bob invokes $\mathsf{CollRes}$ on range $\mathbf{R}_\ell$. Denote the output as $(\zeta_\ell, X_\ell, \boldsymbol{\varPsi}^{(\ell)})$.
Step 3: Alice sets $\zeta = \mathtt{accept}$, if the following equalities hold

$$\forall 0 \leq \ell \leq 2d, \zeta_\ell \overset{?}{=} \mathtt{accept}, \qquad \bigotimes_{0\leq\ell\leq 2d} \boldsymbol{\varPsi}^{(\ell)} \overset{?}{\equiv} \varDelta; \tag{11}$$

otherwise sets $\zeta = \mathtt{reject}$. Alice outputs $(\zeta, X_0, \varDelta)$.

(Alice, Bob) $\mathsf{CollRes} = \langle \widetilde{\mathsf{Eval}}(\mathbf{D}_{\mathtt{B}}), \widetilde{\mathsf{Ext}}(\mathbf{D}_{\mathtt{A}},\ \mathbf{R},\ \mathcal{K})\rangle$: $\mathbf{D}_{\mathtt{A}} = (N,\ d,\ \varDelta), \mathbf{D}_{\mathtt{B}} = (\mathbf{D}, \mathbf{T}, \mathbf{C})$
Precondition. The query range $\mathbf{R} \subset [\mathcal{Z}]^d$ is a rectangular range.

Step A1: (Alice's first step) Alice chooses a random nonce ρ from $\mathbb{Z}_p^*$ and produces a delegation key $\boldsymbol{\delta}$ w.r.t. range $\mathbf{R}$ and nonce ρ, using the functional encryption scheme. Alice sends $(\mathbf{R}, \boldsymbol{\delta})$ to Bob.
Step B1: (Bob's first step) Bob computes the query result X and proof $(\varPsi_1, \varPsi_2, \varPsi_3, \varPsi_4)$ as follows

$$X \leftarrow \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \mathsf{Att}(\boldsymbol{x}); \qquad \boldsymbol{\varPsi} \leftarrow \bigotimes_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \boldsymbol{t}_{\boldsymbol{x}} = \bigotimes_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \mathsf{DTag}_{\mathcal{K}}(\boldsymbol{x}); \qquad \varPsi_4 \leftarrow \prod_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \mathsf{QTag}_{\mathcal{K}}(\boldsymbol{x}, \rho) \tag{12}$$

where for each point $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$, $\mathsf{QTag}_{\mathcal{K}}(\boldsymbol{x}, \rho)$ is obtained by decryting $\mathsf{CT}_{\boldsymbol{x}}$ using the functional encryption scheme with delegation key $\boldsymbol{\delta}$. Bob sends $(X,\ \boldsymbol{\varPsi},\ \varPsi_4)$ to Alice.

Step A2: (Alice's second step) Alice verifies whether $(\boldsymbol{\varPsi}, \varPsi_4)$ are valid tags for X under DTag and QTag respectively, using the private key $\mathcal{K}$ and ρ. If the following equation holds,

$$\mathsf{Verify}_{\rho,\mathcal{K}}(X,\ \boldsymbol{\varPsi},\ \varPsi_4) \overset{?}{=} 1$$

then sets $\zeta = \mathtt{accept}$. Otherwise sets $\zeta = \mathtt{reject}$. Alice outputs $(\zeta, X, \boldsymbol{\varPsi}))$

– Type III adversary: The same as Type II adversary, with additional constraint: $\mu_i = 0$ for $\boldsymbol{x}_i \in \mathbf{D} \cap \mathbf{R}^{\mathsf{C}}$.
– Type IV adversary: The same as Type III adversary, with additional constraint: $\mu_i = 1$ for $\boldsymbol{x}_i \in \mathbf{D} \cap \mathbf{R}$.

It seems that from Type I to Type IV adversaries are more and more restricted, in the sense that

$$\{\text{Type I Adversary}\} \supseteq \{\text{Type II Adversary}\} \supseteq \{\text{Type III Adversary}\} \supseteq \{\text{Type IV Adversary}\} \tag{14}$$

However, [1] showed that, in the above formula (14), (*informally*) each inclusion relation $\supseteq$ can be replaced by equality $=$, under related cryptographic assumptions (**GKEA** etc). Furthermore, [1] proved that there exists no Type IV adversary under certain crypgraphic assumptions.

5 Authenticating Aggregate queries beyond count: SUM, MIN, MAX

5.1 SUM

Suppose each data point $\boldsymbol{x} \in \mathbf{D}$ is associated with an attribute value $\mathsf{Att}(\boldsymbol{x})$. The sum query with range $\mathbf{R}$ asks for the summation $\sum_{\boldsymbol{x} \in \mathbf{D}} \mathsf{Att}(\boldsymbol{x})$.

5.1.1 Summing 1D Attribute

Suppose the attribute value is in 1D, i.e. $\mathsf{Att}(\boldsymbol{x}) \in [\mathcal{Z}]$. The authentication scheme for Sum is identical to the scheme in Figure 1 for Count, except that

– the attribute function $\mathtt{Att}(\boldsymbol{x}) = 1$ is redefined as $\mathtt{Att}(\boldsymbol{x}) = y_{\boldsymbol{x}} \in [\mathcal{Z}]$.
– Alice sends $\{\mathtt{Att}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbf{D}\}$ to Bob in the setup phase, and Bob keeps it along with the dataset and tags.

Denote the modified scheme as $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(1)})$.

Lemma 1 *Suppose attribute value is in 1D. The extended scheme* $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(1)})$ *is a* $\mathcal{VRC}$ *w.r.t.* Sum*, i.e. it is* correct *and* sound *to authenticate* Sum.

5.1.2 Summing Multi-Dimensional Attribute

Suppose the attribute value $\mathsf{Att}(\boldsymbol{x}) = \boldsymbol{x} \in [\mathcal{Z}]^n$ for some integer n. The authentication scheme for summing n-dimensional attribute is identical to the scheme in Section 5.1.1 for 1D case, except that

– The secret key $\mathcal{K}$ generated by KGen contains an additional element $\boldsymbol{s} = (s_1, \ldots, s_n)$ which is randomly chosen from the domain $\left(\mathbb{Z}_p^*\right)^n$;
– The attribute function $\mathtt{Att}(\boldsymbol{x}) = y_{\boldsymbol{x}} \in [\mathcal{Z}]$ and the tag function

$$\mathsf{DTag}(\boldsymbol{x}) = \left(\theta^{\mathsf{Att}(\boldsymbol{x})} v_{\boldsymbol{x}},\ v_{\boldsymbol{x}}^{\beta},\ w_{\boldsymbol{x}}\right) = \left(\theta^{y_{\boldsymbol{x}}} v_{\boldsymbol{x}},\ v_{\boldsymbol{x}}^{\beta},\ w_{\boldsymbol{x}}\right)$$

are redefined as:

$$\mathtt{Att}(\boldsymbol{x}) = \boldsymbol{x} \in [\mathcal{Z}]^n; \qquad \mathsf{DTag}(\boldsymbol{x}) = \left(\theta^{\langle \boldsymbol{x},\ \boldsymbol{s}\rangle} v_{\boldsymbol{x}},\ v_{\boldsymbol{x}}^{\beta},\ w_{\boldsymbol{x}}\right)$$

– Step B1 and Step A2 in $\mathsf{CollRes}$ are modified accordingly (i.e. In Step A2, X is replaced by the inner product $\langle \boldsymbol{X},\ \boldsymbol{s}\rangle$).

Denote the modified scheme as $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(n)})$.

Theorem 2 *The extended scheme* $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(n)})$ *is a* $\mathcal{VRC}$ *w.r.t.* Sum*, i.e. it is* correct *and* sound *to authenticate* Sum.

Proof (of Theorem 2). The correctness part is straightforward due to the homomorphic property (i.e. Equation (10)) of $(\mathsf{DTag}, \mathsf{QTag}, \mathsf{Verify})$. We focus on the soundness part.

Using proof by contradiction, suppose that there exists a PPT adversary $\mathcal{B}$ which can output a wrong query result $\boldsymbol{Y} \neq \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \mathsf{Att}(\boldsymbol{x}) \pmod p$ for sum query with range $\mathbf{R}$ and passes all verifications with non-negligible probability ϵ.

Part I: *We will show that* $\langle \boldsymbol{Y},\ \boldsymbol{s}\rangle = \langle \boldsymbol{X},\ \boldsymbol{s}\rangle \mod p$*, where* $\boldsymbol{s}$ *is a part of private key and* $\boldsymbol{X} = \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \mathsf{Att}(\boldsymbol{x}) \pmod p$ *is the correct query result for the corresponding query* $\mathbf{R}$.

We construct an adversary $\mathcal{B}'$ to against the scheme $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(1)})$ for 1D case, based on $\mathcal{B}$. Adversary $\mathcal{B}'$ will simulate two instances of experiments:

– Experiment $\mathsf{Exp}_{\mathcal{B}'}^{\mathcal{E}_{1D}}$ for scheme $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(1)})$: $\mathcal{B}'$ takes the role of Bob and interacts with Alice.
– Experiment $\mathsf{Exp}_{\mathcal{B}}^{\mathcal{E}_{dD}}$ for scheme $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(d)})$: $\mathcal{B}'$ takes the role of Alice and $\mathcal{B}$ takes the role of Bob.

Alice		$\mathcal{B}'$		$\mathcal{B}$
				Choose a dataset $\mathbf{D} \subseteq [\mathcal{Z}]^d$
			$\xleftarrow{\mathbf{D}}$	
		Choose $\boldsymbol{s} \in_R \left(\mathbb{Z}_p^*\right)^n$; for each $\boldsymbol{x} \in \mathbf{D}$, compute $y_{\boldsymbol{x}} = \langle \boldsymbol{x},\ \boldsymbol{s}\rangle$;		
	$\xleftarrow{\mathbf{D}, \{y_{\boldsymbol{x}}\}}$			
Run DEnc to generate $\mathbf{T}, \mathbf{C}$				
	$\xrightarrow{\mathbf{T}, \mathbf{C}}$			
			$\xrightarrow{\mathbf{T}, \mathbf{C}}$	
				Choose a query $\mathbf{R}_i$
			$\xleftarrow{\mathbf{R}_i}$	
		Forward query $\mathbf{R}_i$ to Alice		
	$\xleftarrow{\mathbf{R}_i}$			
Generate challenge $\boldsymbol{\delta}$				
	$\xrightarrow{\boldsymbol{\delta}}$			
		Forward reply $\boldsymbol{\delta}$ to $\mathcal{B}$		
			$\xrightarrow{\boldsymbol{\delta}}$	
				Generate result and proof $(\boldsymbol{X}, \mathit{\Pi})$
			$\xleftarrow{\boldsymbol{X}, \mathit{\Pi}}$	
		Compute $y = \langle \boldsymbol{X},\ \boldsymbol{s}\rangle$		
	$\xleftarrow{y, \mathit{\Pi}}$			
Verify the proof				

By our hypothesis, with non-negligible probability ϵ, we have $\boldsymbol{Y} \neq \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \boldsymbol{x}$ and the proof $\mathit{\Pi}_{\boldsymbol{Y}}$ passes all verification. As a result, with probability at least ϵ, $\mathcal{B}'$'s reply is $(\langle \boldsymbol{Y},\ \boldsymbol{s}\rangle, \mathit{\Pi}_{\boldsymbol{Y}})$ passes all verification. On the other hand, Lemma 1 says, for any PPT adversary which can output a query result with a proof, if the proof is valid then the query result is correct with overwhelming high probability $1-\epsilon'$ (ϵ' is some negligible function). Hence, with probability at least $\epsilon(1-\epsilon')$, we have

$$\langle \boldsymbol{Y},\ \boldsymbol{s}\rangle = \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} y_{\boldsymbol{x}} = \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \langle \boldsymbol{x},\ \boldsymbol{s}\rangle = \langle \boldsymbol{X},\ \boldsymbol{s}\rangle\,,\ \text{where } \boldsymbol{X} = \sum_{\boldsymbol{x}\in\mathbf{D}\cap\mathbf{R}} \boldsymbol{x} \neq \boldsymbol{Y}.$$

Therefore, with non-negligible probability $\epsilon(1-\epsilon')$, the adversary $\mathcal{B}$ can output two distinct values $\boldsymbol{X}$ and $\boldsymbol{Y}$ such that $\langle \boldsymbol{Y},\ \boldsymbol{s}\rangle = \langle \boldsymbol{X},\ \boldsymbol{s}\rangle \mod p$.

Part II: *We construct an algorithm to solve Discrete Log Problem (**DLP**) based on the adversary $\mathcal{B}$.*

Solve Discrete Log Problem (**DLP**) based on Adversary $\mathcal{B}$

1. Input is $(u, u^a) \in \widetilde{\mathbb{G}}^2$. The goal is to find $a \in \mathbb{Z}_p^*$.
2. For each $i \in [d]$, choose y_i, z_i from $\mathbb{Z}_p^*$ at random and set $\theta_i = (u^a)^{y_i} u^{z_i} \in \widetilde{\mathbb{G}}$.
3. Simulate the authentication scheme $(\mathsf{KGen}, \mathsf{DEnc}, \textsc{Sum}^{(n)})$ with the following modifications:
   - The secret key θ and $\boldsymbol{s} = (s_1, \ldots, s_n)$ are implicitely defined by $\theta_i = \theta^{s_i}$.
   - The simulator does not know values of $(\theta, \boldsymbol{s})$, but still can compute $\theta^{\langle \mathsf{Att}(\boldsymbol{x}),\ \boldsymbol{s} \rangle}$:

$$\theta^{\langle \mathsf{Att}(\boldsymbol{x}),\ \boldsymbol{s} \rangle} = \prod_{i \in [n]} \theta_i^{\lambda_i}. \text{ where } \mathsf{Att}(\boldsymbol{x}) = (\lambda_1, \ldots, \lambda_n)$$

4. Invoke the adversary $\mathcal{B}$ and obtains output $(\boldsymbol{X}, \boldsymbol{Y})$. Let $\boldsymbol{w} = (w_1, \ldots, w_n) = \boldsymbol{X} - \boldsymbol{Y} \mod p$. Applying the result in Part I, with non-negligible probability, we have

$$\theta^{\langle \boldsymbol{w},\ \boldsymbol{s} \rangle} = 1 \text{ and } \boldsymbol{w} \neq \boldsymbol{0} \mod p$$

5. A univariable equation on unknown a can be formed: Let $\boldsymbol{y} = (y_1, \ldots, y_n)$ and $\boldsymbol{z} = (z_1, \ldots, z_n)$.

$$\prod_{i \in [n]} \theta_i^{w_i} = \prod_{i \in [n]} u^{w_i (a y_i + z_i)} = 1.$$

$$a \langle \boldsymbol{y},\ \boldsymbol{w} \rangle + \langle \boldsymbol{z},\ \boldsymbol{w} \rangle = 0 \mod p$$

   Note: Given $\theta_i, i \in [n]$, y_i's are truely random. Hence, the probability that $\langle \boldsymbol{y},\ \boldsymbol{w} \rangle = 0$ is negligible.
6. Solve the equation and get the root a^*. Output a^*.

The constructed PPT algorithm solves DLP with non-negligible probability. The contradiction with DL Assumption, implies that our hypothesis is wrong and such adversary $\mathcal{B}$ does not exist. Consequently, the soundness part of Theorem 2 is proved. □

For the rest of remaining part of this paper, we assume the attribute function is

$$\mathsf{Att}(\boldsymbol{x}) = (x_1, \ldots, x_d, 1), \text{ where } \boldsymbol{x} = (x_1, \ldots, x_d).$$

By Theorem 2, our scheme can support both summing over the first d dimensions of attribute values and counting over the last dimension of attribute values. The Avg query can be authenticated by combination of Sum and Count.

5.2 MIN

A min query $\textsc{Min}(\mathbf{R}, \iota)$ with query range $\mathbf{R}$ and dimension $\iota \in [d]$, asks for the minimum attribute values along the ι-th dimension among all data points within $\mathbf{D} \cap \mathbf{R}$. We find that Min query can be converted to Count query. The conversion is based on this proposition:

Proposition 1 *For any finite set S of numbers,*

$$c = \min S \qquad \Leftrightarrow \qquad c \in S \ \wedge\ |S| = |\{x : x \in S \wedge x \geq c\}|. \tag{15}$$

Suppose Alice asks Bob for the minimum attribute value along the ι-th dimension of points within range $\mathbf{R}$. Bob returns a data point $\boldsymbol{x}$, such that $\mathsf{Att}(\boldsymbol{x})[\iota]$ is minimum in the set S of attribute values along the ι-th dimension of all points within range $\mathbf{R}$ (i.e. $S = \{\boldsymbol{x}[\iota] : \boldsymbol{x} \in \mathbf{R} \cap \mathbf{D}\}$). Meanwhile, Bob also sends a proof to show that $\boldsymbol{x} \in \mathbf{D}$. Then Alice issues two Count queries to Bob: (1) $\textsc{Count}(\mathbf{R})$, i.e. the size of set S; (2) $\textsc{Count}\left(\mathbf{R} \bigcap \left([\mathcal{Z}]^{\iota-1} \times [c, \mathcal{Z}] \times [\mathcal{Z}]^{d-\iota}\right)\right)$ where $c = \mathsf{Att}(\boldsymbol{x})[\iota]$, i.e. the size of set $\{x : x \in S \wedge x \geq c\}$. Bob is expected to return the two count numbers with proofs following the scheme in [1]. Alice believes c is the minimum value if all proofs are valid and the two count nubmers are equal. The algorithm is showed in Figure 2.

Theorem 3 *The extended scheme is a $\mathcal{VRC}$ w.r.t.* Min, *i.e. it is* correct *and* sound *to authenticate* Min.

Similarly, Max query can be authenticated.

5.3 Median

Median can also be converted into Count. Quartile or percentile queries can be handled in a similar way.

Proposition 2 *Let S be a finite set of numbers.*

$$c \text{ is the median in set } S \quad \Leftrightarrow \quad c \in S \;\wedge\; |\{x : x \in S \wedge x \le c\}| \ge \lceil \frac{|S|}{2} \rceil \;\wedge\; |\{x : x \in S \wedge x \ge c\}| \ge \lceil \frac{|S|}{2} \rceil \tag{16}$$

Suppose Alice asks Bob for the median attribute value along the ι-th dimension of points within range $\mathbf{R}$. Bob returns a data point $\boldsymbol{x}$, such that $\mathsf{Att}(\boldsymbol{x})[\iota]$ is the median in the set S of attribute values along the ι-th dimension of all points within range $\mathbf{R}$ (i.e. $S = \{\boldsymbol{x}[\iota] : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}$). Meanwhile, Bob also sends a proof to show that $\boldsymbol{x} \in \mathbf{D}$. Then Alice issues three Count queries to Bob: (1) Count($\mathbf{R}$), i.e. the size of set S; (2) Count $\left(\mathbf{R} \bigcap \left([\mathcal{Z}]^{\iota-1} \times [c, \mathcal{Z}] \times [\mathcal{Z}]^{d-\iota}\right)\right)$ where $c = \mathsf{Att}(\boldsymbol{x})[\iota]$, i.e. the size of set $\{x : x \in S \wedge x \ge c\}$; (3) Count $\left(\mathbf{R} \bigcap \left([\mathcal{Z}]^{\iota-1} \times [1, c] \times [\mathcal{Z}]^{d-\iota}\right)\right)$ i.e. the size of set $\{x : x \in S \wedge x \le c\}$. Bob is expected to return the three count numbers N_1, N_2 and N_3 with proofs following the scheme in [1]. Alice believes c is the median value if all proofs are valid and $N_2 \ge \lceil \frac{N_1}{2} \rceil$ and $N_3 \ge \lceil \frac{N_1}{2} \rceil$.

The algorithm is showned in Figure 2. Note that when the size of S is even, there are two medians. For simplicity of presentation of the algorithm, we request Bob to return either one of the two medians, instead of both.

Fig. 2: Authenticating Min query and Median query.

(Alice, Bob) Min($\mathbf{R}, \iota$):

1. Alice sends $(\mathbf{R}, \iota)$ to Bob.
2. Bob finds $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{x}[\iota]$ and sends $\boldsymbol{x}^*$ to Alice.
3. Alice issues a count query Count($\{\boldsymbol{x}^*\}$) with range $\{\boldsymbol{x}^*\}$ to Bob and gets authenticated query result N_0.
4. Alice sets $c = \boldsymbol{x}^*[\iota]$ and finds the range $\mathbf{R}_c = \mathbf{R} \cap \left([\mathcal{Z}]^{\iota-1} \times [c, \mathcal{Z}] \times [\mathcal{Z}]^{d-\iota}\right)$.
5. Alice issues two count queries Count($\mathbf{R}$) and Count($\mathbf{R}_c$) to Bob and gets authenticated results N_1 and N_2.
6. Alice accepts c as the minimum, if all verifications succeed and $N_0 \ge 1$ and $N_1 = N_2$.

(Alice, Bob) Median($\mathbf{R}, \iota$):

1. Alice sends $(\mathbf{R}, \iota)$ to Bob.
2. Bob finds $\boldsymbol{x}^*$ such that $\boldsymbol{x}^*[\iota]$ is a median among $\{\boldsymbol{x}[\iota] : \boldsymbol{x} \in \mathbf{D}\}$ and sends $\boldsymbol{x}^*$ to Alice.
3. Alice issues a count query Count($\{\boldsymbol{x}^*\}$) with range $\{\boldsymbol{x}^*\}$ to Bob and gets authenticated query result N_0.
4. Alice sets $c = \boldsymbol{x}^*[\iota]$ and finds the range $\mathbf{R}_c^+ = \mathbf{R} \cap \left([\mathcal{Z}]^{\iota-1} \times [c, \mathcal{Z}] \times [\mathcal{Z}]^{d-\iota}\right)$ and range $\mathbf{R}_c^- = \mathbf{R} \cap \left([\mathcal{Z}]^{\iota-1} \times [1, c] \times [\mathcal{Z}]^{d-\iota}\right)$.
5. Alice issues three count queries Count($\mathbf{R}$), Count($\mathbf{R}_c^+$) and Count($\mathbf{R}_c^-$) to Bob and gets authenticated results N_1, N_2 and N_3.
6. Alice accepts c as the median, if all verifications succeed and $N_0 \ge 1$ and $N_2 \ge \lceil \frac{N_1}{2} \rceil$ and $N_3 \ge \lceil \frac{N_1}{2} \rceil$.

5.4 Beyond Aggregate queries: Range Selection

In this section, we extend our method to support range selection and range selection with projection.

5.5 Range Selection

A range selection query with range $\mathbf{R}$ asks for all data points within the range $\mathbf{R}$:

$$\textsc{RangeSelect}(\mathbf{R}) = \{\boldsymbol{x} : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}.$$

We assume the dataset $\mathbf{D}$ is a set of *distinct* points. The authentication scheme for range selection query is as follows:

Authenticating Multidimensional Range Selection Query

1. In the setup, Alice generates a signature $\mathsf{Sig}(\boldsymbol{x})$ for each data point $\boldsymbol{x} \in \mathbf{D}$ using an aggregate signature scheme, and sends all signatures to Bob.
2. To answer a range selection query with range $\mathbf{R}$, Bob finds the set $S = \{\boldsymbol{x} : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}$ and computes an aggregated signature $\mathsf{Sig}(S)$ for set S from signatures $\mathsf{Sig}(\boldsymbol{x})$'s for point $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$, using the aggregate signature scheme. Bob sends $(S, \mathsf{Sig}(S))$ to Alice.
3. Alice verifies: (1) Is S a set of distinct points? (2) Is S a subset of query range $\mathbf{R}$? (3) Is $\mathsf{Sig}(S)$ a valid signature for S?
4. Alice issues a count query with range $\mathbf{R}$ to Bob and gets authenticated result N_0.
5. Alice verifies whether $|S| = N_0$.
6. Alice accepts S as the query result, if all verifications succeed.

The above method has communication overhead equal to that of $\textsc{Count}$ query: $O(d^2 \log^2 \mathcal{Z})$. To the best of our knowledge, this is the first efficient $\mathcal{VRC}$ (See Definition 2) to authenticate multidimensional range selection query.

5.6 Range Selection with Projection

A range selection query with projection on the 1st dimension asks for the 1st dimension of all data points within the query range

$$\textsc{RangeSelect}(\mathbf{R}) = \{\boldsymbol{x}[1] : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}.$$

Authenticating this query with sublinear communication overhead is more challenging that range selection without projection. If we just apply the method in Section 5.5, the communiction overhead will be linear: Since only the 1st dimension of data points within the query range is asked for, but all dimensions of such data points are returned as the query result. The requirement of sublinear communication overhead implies that Alice has to verify whether $\boldsymbol{x} \in \mathbf{R}$, with only the knowledge of the first dimension $\boldsymbol{x}[1]$ of point $\boldsymbol{x}$.

Our idea is that: Alice derives the randomness $v_{\boldsymbol{x}}$ from $\boldsymbol{x}[1]$ only using a pseudorandom function $\mathsf{F}_{\varpi}(\cdot)$ when generating the authentication tag during the setup. Then Alice issues a count query with range $\mathbf{R}$ to Bob and receives from Bob the query result N_0 and its proof $\boldsymbol{\Psi}$. Meanwhile, Alice also receives $S = \{\boldsymbol{x}[1] : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}$. Alice verifices whether the proof $\boldsymbol{\Psi}$ is consistent with $\prod_{x \in S} \mathsf{F}_{\varpi}(x)$ and whether $|S| = N_0$. The detailed algorithm is given in Figure 3.

This solution has a limitation: When generating authentication tag during the setup, if Alice derives the randomness $v_{\boldsymbol{x}}$ from $\boldsymbol{x}[1]$, then the resulting scheme only supports projection on the 1st dimension. To support projection on any combination of dimensions, Alice has to generate 2^d authentication tags for each data point, and one tag for one subset of $[d]$. As a result, we can authenticate range selection with projection, at the cost of $O(d^2 \log^2 \mathcal{Z})$ communication overhead per query and $O(dN \cdot 2^d)$ storage on Bob's side.

Theorem 4 *The extended scheme is a $\mathcal{VRC}$ w.r.t. range selection, i.e. it is* correct *and* sound *to authenticate d-dimensional range selection.*

Fig. 3: Construction of $\mathcal{RC}$ protocol $\mathcal{E} = (\mathsf{KGen}, \mathsf{DEnc}, \langle \mathsf{Eval}, \mathsf{Ext} \rangle)$ to authenticate multidimensional range selection query with projection on the 1st dimension. The attribute function is $\mathsf{Att}(\boldsymbol{x}) = (x_1, \ldots, x_d, 1)$ for each $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbf{D}$.

(Alice) $\mathsf{KGen}(1^\kappa)$:

1. Generate a private key $\mathcal{K}$ as in Figure 2 in MAIA.
2. Let $\{\mathsf{F}_\varpi : [\mathcal{Z}]^d \to \widetilde{\mathbb{G}}\}_{\varpi \in \{0,1\}^\kappa}$ be a pseudoranom function. Choose a random seed $\varpi \in \{0,1\}^\kappa$.
3. Set $\mathcal{K} \leftarrow (\mathcal{K}, \varpi)$.
4. Output the private key $\mathcal{K}$.

(Alice) $\mathsf{DEnc}(\mathbf{D}; \mathcal{K})$: The same as in DEnc in Figure 1 [1], except that the randomness $v_{\boldsymbol{x}} \in \widetilde{\mathbb{G}}$ is generated in this way: Let $\boldsymbol{x}[1]$ denote the first component of vector value $\boldsymbol{x}$.

$$\forall \boldsymbol{x} \in \mathbf{D}, v_{\boldsymbol{x}} = \mathsf{F}_\varpi(\boldsymbol{x}[1]).$$

(Alice, Bob) $\mathsf{RangeSelect} = \langle \mathsf{Eval}(\mathbf{D}_\mathtt{B}), \mathsf{Ext}(\mathbf{D}_\mathtt{A},\ \mathbf{R},\ \mathcal{K}) \rangle$: $\mathbf{D}_\mathtt{A} = (N,\ d,\ \Delta), \mathbf{D}_\mathtt{B} = (\mathbf{D}, \mathbf{T}, \mathbf{C})$
Precondition: The query range $\mathbf{R} \subset [\mathcal{Z}]^d$ is a rectangular range.

Step 1: Alice partitions the complement range $\mathbf{R}^\complement$ into $2d$ rectangular ranges $\{\mathbf{R}_\ell \subset [\mathcal{Z}]^d : \ell \in [1, 2d]\}$, and sets $\mathbf{R}_0 = \mathbf{R}$.
Step 2—Reduction: For $0 \leq \ell \leq 2d$, Alice and Bob invokes $\mathsf{CollRes}$ on range $\mathbf{R}_\ell$. Denote the output as $(\zeta_\ell, X_\ell, \boldsymbol{\Psi}^{(\ell)})$.
Step 3: Alice verifies whether the following equalities hold:

$$\forall 0 \leq \ell \leq 2d, \zeta_\ell \overset{?}{=} \mathtt{accept}, \qquad \bigotimes_{0 \leq \ell \leq 2d} \boldsymbol{\Psi}^{(\ell)} \overset{?}{\equiv} \Delta. \tag{17}$$

Note: Until this point, all are identical to the Count *algorithm.*
Step 4: Bob sends back $S = \{\boldsymbol{x}[1] : \boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}\}$ to Alice.
Step 5: Alice verifies whether the following equalities hold: Let $\boldsymbol{\Psi}^{(0)}[2]$ denote the 2nd component of vector value $\boldsymbol{\Psi}^{(0)}$.

$$\boldsymbol{\Psi}^{(0)}[2] \overset{?}{=} \prod_{\boldsymbol{x} \in S} \mathsf{F}_\varpi(\boldsymbol{x}[1])^\beta; \qquad |S| = X_0 \tag{18}$$

Alice sets $\zeta = \mathtt{accept}$ if all verifications in equation (17) and equation (18) succeed; and sets $\zeta = \mathtt{reject}$ otherwise. Alice outputs (ζ, S, Δ).

(Alice, Bob) $\mathsf{CollRes} = \langle \widetilde{\mathsf{Eval}}(\mathbf{D}_\mathtt{B}), \widetilde{\mathsf{Ext}}(\mathbf{D}_\mathtt{A},\ \mathbf{R},\ \mathcal{K}) \rangle$: $\mathbf{D}_\mathtt{A} = (N,\ d,\ \Delta), \mathbf{D}_\mathtt{B} = (\mathbf{D}, \mathbf{T}, \mathbf{C})$
Identical with $\mathsf{CollRes}$ in Figure 1. Save the details.

6 Dynamic Dataset

6.1 Insertion

$\mathsf{Insert}(\hat{\mathbf{D}}, \mathcal{K})$:
$\mathsf{Precondition}$: Alice has $\mathbf{D}_s = (\mathcal{K}, N, d, \varDelta)$; Bob has $\mathbf{D}_p = (\mathbf{D}, \mathbf{T}, \mathbf{C})$.
Alice runs the algorithm $\mathsf{DEnc}(\hat{\mathbf{D}}, \mathcal{K})$ to generate $(\hat{\mathbf{D}}_s = (\hat{N}, d, \hat{\varDelta}), \hat{\mathbf{D}}_p = (\hat{\mathbf{D}}, \hat{\mathbf{T}}, \hat{\mathbf{C}}))$. Alice updates $\varDelta \leftarrow \varDelta \cdot \hat{\varDelta}, N \leftarrow N + \hat{N}$. Alice sends $(\hat{\mathbf{D}}, \hat{\mathbf{T}}, \hat{\mathbf{C}})$ to Bob. Bob sets $\mathbf{D} = \mathbf{D} \cup \hat{\mathbf{D}}, \mathbf{T} = \mathbf{T} \cup \hat{\mathbf{T}}, \mathbf{C} = \mathbf{C} \cup \hat{\mathbf{C}}$.

We can prove the security if insertion is non-adaptive, i.e. the inserted items are sampled from a particular distribution.

Theorem 5 *The extended scheme is* correct *and* sound *to authenticate d-dimensional* Count*,* Sum*,* Avg*,* Min*,* Max*,* Median *and range selection queries over dynamic dataset that supports insertion.*

6.2 Deletion

Deletion is equivalent to insertion into another dataset.
Let $\mathcal{E} = (\mathsf{KGen}, \mathsf{DEnc}, \mathsf{ProVer})$.

$\mathsf{KGen}(1^\kappa)$: Run $\mathcal{E}.\mathsf{KGen}(1^\kappa)$ twice independently and output two keys $\mathcal{K}$ and $\overline{\mathcal{K}}$.
$\mathsf{DEnc}(\mathbf{D}; \mathcal{K}, \overline{\mathcal{K}})$: Run $\mathcal{E}.\mathsf{DEnc}(\mathbf{D}; \mathcal{K})$ to generate $(\mathbf{D}_{\mathtt{A}}, \mathbf{D}_{\mathtt{B}})$. Set $\overline{\mathbf{D}} = \overline{\mathbf{T}} = \overline{\mathbf{C}} = \emptyset$, $\overline{\mathbf{D}}_{\mathtt{A}} = (\overline{N} = 0, d, \overline{\varDelta} = 1)$, and $\overline{\mathbf{D}}_{\mathtt{B}} = (\overline{\mathbf{D}}, \overline{\mathbf{T}}, \overline{\mathbf{C}}, \overline{pk})$, where $\overline{pk}$ is part of $\overline{\mathcal{K}}$.
$\mathsf{Insert}(\mathbf{D}', \mathcal{K})$: Run $\mathcal{E}.\mathsf{Insert}(\mathbf{D}', \mathcal{K})$.
$\mathsf{Delete}(\mathbf{D}')$: Set $\mathbf{D}' \leftarrow \mathbf{D}' \cap \mathbf{D}$. Run $\mathcal{E}.\mathsf{Insert}(\mathbf{D}', \overline{\mathcal{K}})$ to insert points in $\mathbf{D}'$ into the complement dataset $\overline{\mathbf{D}}$.
$\mathsf{Sum}(\mathbf{R}; \mathcal{K}, \overline{\mathcal{K}})$: Run $\mathcal{E}.\mathsf{Sum}(\mathbf{R}; \mathcal{K})$ over dataset $\mathbf{D}$ to obtain $(\zeta, X, \varDelta)$; run $\mathcal{E}.\mathsf{Sum}(\mathbf{R}; \overline{\mathcal{K}})$ over the complement dataset $\overline{\mathbf{D}}$ to obtain $(\overline{\zeta}, \overline{X}, \overline{\varDelta})$. If $\zeta = \overline{\zeta} = \mathtt{accept}$, then set $\varsigma = \mathtt{accept}$; otherwise set $\varsigma = \mathtt{reject}$. Output $(\varsigma, X - \overline{X}, \varDelta / \overline{\varDelta})$.
$\mathsf{Min}(\mathbf{R}, \iota; \mathcal{K}, \overline{\mathcal{K}})$: Assume $\sum_{\boldsymbol{x} \in \mathbf{D}} \boldsymbol{x} < (p, \ldots, p)$

1. Alice sends range $\mathbf{R}$ to Bob.
2. Bob finds $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{x}[\iota]$. Bob sends $\boldsymbol{x}^*$ back to Alice.
3. Alice issues Sum query with range $\mathbf{R} = \{\boldsymbol{x}^*\}$ to Bob over dataset $\mathcal{D}$ and $\overline{\mathbf{D}}$, and obtains output $(\zeta, X - \overline{X}, \varDelta / \overline{\varDelta})$. If $\zeta = \mathtt{accept}$ and $X - \overline{X} > 0$, then believes that $\boldsymbol{x}^*$ is a valid data point.
4. Alice issues a Count query.

Corollary 6 *The extended scheme is* correct *and* sound *to authenticate d-dimensional* Count*,* Sum*,* Avg*,* Min*,* Max*,* Median *and range selection queries over dynamic dataset that supports both insertion and deletion.*

7 Security beyond authentication

7.1 Privacy

At first, let us distinguish aggregate attributes and selection attributes: (1) Aggregate attributes are dimensions along which a query apply the aggreate operation (like sum, min, max); (2) Selection attributes are dimensions on which a query applys range constraint. Take an example, a sum query which asks for the sum of the 3rd dimension of data points with selection on the 1st and th 2nd dimensions: Let $\mathbf{R} = [a_1, b_1] \times [a_2, b_2] \times [1, \mathcal{Z}]^{d-2}$ be the query range.

$$\sum_{\boldsymbol{x} \in \mathbf{R}} \boldsymbol{x}[3]$$

In this example, the 3rd dimension is the aggregate attribute, and the 1st and the 2nd are selection attributes. Note that in some query, a dimension can be both aggregate attribute and selection attribute.

We found previous working on privacy preserving aggregate range query over outsourced dataset can be divided into two categaries:

- Protect the privacy of aggregate attributes, where any aggregate attribute is not selection attribute, e.g. [6]. These works typically employs homomorphic encryption scheme to hide the aggreate attribute values and reserve the capability of doing aggregation. Particularlly, additive homomorphic encryption scheme (e.g. Paillier system [33]) for aggregate sum query and order preserving encryption scheme (e.g. [34]) for aggregate min/max query.
- Protect the privacy of selection attributes. fully homomorphic encryption scheme [29] Potentially, MRQED scheme [35] can also be adopted for this purpose.

7.1.1 Privacy of Aggregate Attributes [6]

W.L.O.G, we assume only the 1st dimension is aggregate attribute, and in every query range $\mathbf{R}$ has the form $\mathbf{R} = [1, \mathcal{Z}] \times [a_2, b_2] \times \ldots [a_d, b_d] \subseteq [1, \mathcal{Z}]^d$, i.e. the query has no constraint on the 1st dimension. Let $(\mathsf{G}, \mathsf{E}, \mathsf{D}, \mathsf{H})$ be an additive homomorphic encyrption scheme (e.g. Paillier system [33]).

The new scheme is identical to the solution for aggregate sum query in Section 5.1, except that

- Additionally, in the setup, Alice generates a key pair $(K_{\mathsf{E}}, K_{\mathsf{D}})$ by running the key generating algorithm G, and for each point $\boldsymbol{x} \in \mathbf{D}$ replaces the first dimension $\boldsymbol{x}[1]$ with the ciphertext $\mathsf{E}_K(\boldsymbol{x}[1])$.
- To answer a sum query over 1st dimension, Bob "sums" all ciphertexts $\mathsf{E}_K(\boldsymbol{x}[1])$ for points $\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}$ using the homorphic property of the encryption scheme, and sends the resulting ciphertext $\mathsf{CT} = \mathsf{E}(\sum_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{x}[1])$ of sum to Alice as the query result.
- Alice can decrypt ciphertext CT with decryption key K_{D} to recover the sum $\sum_{\boldsymbol{x} \in \mathbf{D} \cap \mathbf{R}} \boldsymbol{x}[1]$.

7.1.2 Privacy of Selection Attributes [27, 35]

A straightforward approach is to apply order preserving encryption scheme [36, 34]. During the setup, Alice can encrypt each selection attribute with an order preserving encryption scheme. Since the order between any two values are preserved, Bob can do comparision directly.

Alternatively, we may apply MRQED, which is predicate encryption scheme supporting multidimensional range query. Under MRQED, a message Msg can be encrypted under an identiy $\boldsymbol{x}$, which is a point in a d-dimensional space $[1, \mathcal{Z}]^d$. From the master secret key, a delegation decryption $\boldsymbol{\delta}$ w.r.t. a d-dimensional rectangular range $\mathbf{R}$ can be derived. With the delegation decryption key $\boldsymbol{\delta}$, the ciphertext for the message Msg under identity point $\boldsymbol{x}$ can be decrypted to recover Msg, iff the identity point $\boldsymbol{x}$ is within the range $\mathbf{R}$.

Let $\mathbb{M}$ be the domain of the messages to be encrypted under MRQED, and $\hat{\mathbb{M}}$ be a subset of $\mathbb{M}$ such that (1) the size of $\hat{\mathbb{M}}$ is superpolynomial; (2) the ratio $\frac{|\hat{\mathbb{M}}|}{|\mathbb{M}|}$ is negligible. During the setup, for each data point $\boldsymbol{x} \in \mathbf{D}$, Alice choose a random message $\mathsf{Msg}_{\boldsymbol{x}} \in \hat{\mathbb{M}}$, and encrypts the message $\mathsf{Msg}_{\boldsymbol{x}}$ under identiy point $\boldsymbol{x}$ usig MRQED encryption scheme. Alice replaces each data point with its corresponding ciphertext and sends all of N resulting ciphertexts to Bob. Later, Alice wants to query range $\mathbf{R}$, then she can derive the delegation decryption key $\boldsymbol{\delta}$ w.r.t $\mathbf{R}$ and sends $\boldsymbol{\delta}$ to Bob. With the delegation decryption key $\boldsymbol{\delta}$, Bob can decide whether a ciphertext is corresponding to a data point $\boldsymbol{x}$ within the query range $\mathbf{R}$, without knowing the value of $\boldsymbol{x}$: Bob decrypts each ciphertext $\boldsymbol{C}_{\boldsymbol{x}}$ with the delegation key, and gets the decrypted value $\mathcal{M}_{\boldsymbol{x}}$. If $\mathcal{M}_{\boldsymbol{x}} \in \hat{\mathbb{M}}$, then $\boldsymbol{x} \in \mathbf{R}$ with o.h.p. Otherwise, $\boldsymbol{x} \notin \mathbf{R}$ definitely.

It is worthy to point out that, the security model of MRQED and fully homomorphic encryption [29] are stronger than [34], which in turn is much stronger than [36].

7.2 Frame Attack

Let $(\mathsf{KG}, \mathsf{Sign}, \mathsf{Verify})$ be a secure digital signature scheme. Suppose Alice has signing key $(PK_{\mathtt{A}, SK_{\mathtt{A}}})$ and Bob has signing key $(PK_{\mathtt{B}, SK_{\mathtt{B}}})$, where only Alice knows the private key $SK_{\mathtt{A}}$, only Bob knows the private key $SK_{\mathtt{B}}$, and the two public keys $PK_{\mathtt{A}}$ and $PK_{\mathtt{B}}$ are known to public. We assume there is no Denial of Service (DOS) attack.

7.3 Static Dataset

If the dataset is static, it will be much easier to prevent frame attack and can be considered as a special case of solution for dynamic dataset.

7.4 Dynamic Dataset

1. During the setup, Alice signs the dataset with her private key and sends the signature to Bob.
2. For each update command $\mathcal{U}$ Alice issues to Bob, Alice has to sign it and sends the signature $\mathsf{Sign}_{SK_{\mathtt{A}}}(\mathcal{U})$ together with the update command to Bob. Then Bob sends an ACK message with signature, i.e. $(\mathcal{U}, \mathsf{Sign}_{SK_{\mathtt{B}}}(\mathcal{U})$ to Alice.
3. For each query $\mathcal{Q}$ Alice issues to Bob, Bob generates the query result $\mathcal{X}$ and proof Π. Bob sends back to Alice the signed result with proof, i.e. $(\mathcal{X}, \Pi, \mathsf{Sign}_{\mathtt{B}}(\mathcal{Q}, \mathcal{X}, \Pi))$.
4. In all above case, the receiver of a signature will verify the validity of the signature using the corresponding public key, and rejects that reply message and redo the task if the signature is not valid.

If Alice claims that Bob returned a wrong query result, then she has to present to the third trusted party two piece of information: (1) the corresponding signed query result: $(\mathcal{X}, \Pi, \mathsf{Sign}_{\mathtt{B}}(\mathcal{Q}, \mathcal{X}, \Pi))$, with $\mathcal{Q}$ as the query of question. (2) the set S_{ACK} all of ACKs to update commands with Bob's signatures. Meanwhile, to prevent his innovant, Bob has to present to the third trusted party the original dataset $\mathbf{D}$ together with its signature, and the set $S_{\mathcal{U}}$ of all update commands signed by Alice.

The third trusted party verifies all signatures using corresponding public key. If Alice presents a message which is wrongly signed by Bob, then decides Alice cheats. Similarly, if Bob presents a message which is wrongly signed by Alice, then decides Bob cheats. The third trusted party then checks the authenticated messages in the following way:

- If $S_{ACK} \subsetneq S_{\mathcal{U}}$, then judges that Bob cheats.
- If $S_{\mathcal{U}} \subsetneq S_{ACK}$, then judges that Alice cheats.
- If $S_{ACK} \neq S_{\mathcal{U}}$, then judges both Alice and Bob cheat.
- Until this point, All parties have a comsesus on the current status of dataset: Let the dataset $\mathbf{D}^{\star}$ be the result after applying the authenticated update command in $S_{\mathcal{U}}$ to the original dataset $\mathbf{D}$. Then, compute the query $\mathcal{Q}$ over the authenticated dataset $\mathbf{D}^{\star}$, and gets the result $\mathcal{Y}$. If $\mathcal{X} \neq \mathcal{Y}$, then judges that Bob cheats.

8 Conclusion

We propose efficient schemes to authenticate queries over static/dynamic outsourced dataset with $O(d^2 \log^2 \mathcal{Z})$ communication overhead, which conquer the "curse of dimensionality". The supported queries include aggregate range query, i.e. Count, Sum, Avg, Min, Max, Median and range selection.

References

1. Xu, J., Chang, E.C.: Authenticating aggregate range queries over multidimensional dataset. Cryptology ePrint Archive, Report 2010/050 (2010) `http://eprint.iacr.org/`.
2. Devanbu, P.T., Gertz, M., Martel, C.U., Stubblebine, S.G.: Authentic Third-party Data Publication. In: Proceedings of the IFIP TC11/ WG11.3 Fourteenth Annual Working Conference on Database Security. (2001) 101–112
3. Hacigümüş, H., Iyer, B., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database-service-provider model. In: SIGMOD '02: ACM SIGMOD International conference on Management of data. (2002) 216–227
4. Pang, H., Tan, K.L.: Verifying Completeness of Relational Query Answers from Online Servers. ACM Trans. Inf. Syst. Secur. **11**(2) (2008) 1–50
5. Cheng, W., Tan, K.L.: Query assurance verification for outsourced multi-dimensional databases. J. Comput. Secur. **17**(1) (2009) 101–126
6. Thompson, B., Yao, D., Haber, S., Horne, W.G., Sander, T.: Privacy-Preserving Computation and Verification of Aggregate Queries on Outsourced Databases. In: PETS '09: Privacy Enhancing Technologies Symposium. (2009)
7. Li, F., Hadjieleftheriou, M., Kollios, G., Reyzin, L.: Authenticated index structures for aggregation queries. ACM Trans. Inf. Syst. Secur. **13** (2010) 32:1–32:35
8. Atallah, M.J., Cho, Y., Kundu, A.: Efficient Data Authentication in an Environment of Untrusted Third-Party Distributors. In: ICDE '08: IEEE International Conference on Data Engineering. (2008) 696–704
9. Martel, C., Nuckolls, G., Devanbu, P., Gertz, M., Kwong, A., Stubblebine, S.G.: A General Model for Authenticated Data Structures. Algorithmica **39**(1) (2004) 21–41
10. Chen, H., Ma, X., Hsu, W.W., Li, N., Wang, Q.: Access Control Friendly Query Verification for Outsourced Data Publishing. In: ESORICS '08: European Symposium on Research in Computer Security. (2008) 177–191

11. Hacigümüs, H., Iyer, B.R., Mehrotra, S.: Efficient Execution of Aggregation Queries over Encrypted Relational Databases. In: DASFAA. (2004) 125–136
12. Mykletun, E., Tsudik, G.: Aggregation Queries in the Database-As-a-Service Model. In: IFIP WG 11.3 Working Conference on Data and Applications Security. (2006) 89–103
13. Ge, T., Zdonik, S.B.: Answering Aggregation Queries in a Secure System Model. In: VLDB '07: International Conference on Very Large Data Bases. (2007) 519–530
14. Devanbu, P., Gertz, M., Martel, C., Stubblebine, S.G.: Authentic data publication over the internet. J. Comput. Secur. **11**(3) (2003) 291–314
15. Pang, H., Jain, A., Ramamritham, K., Tan, K.L.: Verifying completeness of relational query results in data publishing. In: SIGMOD '05: ACM SIGMOD International conference on Management of data. (2005) 407–418
16. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and Integrity in Outsourced Databases. Trans. Storage **2**(2) (2006) 107–138
17. Sion, R.: Query Execution Assurance for Outsourced Databases. In: VLDB '05: International Conference on Very Large Data Bases. (2005) 601–612
18. Li, F., Hadjieleftheriou, M., Kollios, G., Reyzin, L.: Dynamic authenticated index structures for outsourced databases. In: SIGMOD '06: ACM SIGMOD International conference on Management of data. (2006) 121–132
19. Xie, M., Wang, H., Yin, J., Meng, X.: Integrity auditing of outsourced data. In: VLDB '07: International conference on Very large data bases. (2007) 782–793
20. Yang, Y., Papadias, D., Papadopoulos, S., Kalnis, P.: Authenticated join processing in outsourced databases. In: SIGMOD '09: ACM SIGMOD International conference on Management of data. (2009) 5–18
21. Mouratidis, K., Sacharidis, D., Pang, H.: Partially materialized digest scheme: an efficient verification method for outsourced databases. The VLDB Journal **18**(1) (2009) 363–381
22. Pang, H., Zhang, J., Mouratidis, K.: Scalable Verification for Outsourced Dynamic Databases. Proc. VLDB Endow. **2** (2009) 802–813
23. Goodrich, M.T., Tamassia, R., Triandopoulos, N.: Super-Efficient Verification of Dynamic Outsourced Databases. In: CT-RSA '08: The Cryptographer's Track at the RSA Conference on Topics in Cryptology. (2008) 407–424
24. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2) (1978) 120–126
25. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. J. Cryptol. **17**(4) (2004) 297–319
26. Haber, S., Horne, W., Sander, T., Yao, D.: Privacy-Preserving Verification of Aggregate Queries on Outsourced Databases. Technical report, HP Laboratories (2006) HPL-2006-128.
27. Gennaro, R., Gentry, C., Parno, B.: Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers. In: CRYPTO '10: Annual International Cryptology Conference on Advances in Cryptology. (2010) 465–482
28. Chung, K.M., Kalai, Y., Vadhan, S.P.: Improved Delegation of Computation Using Fully Homomorphic Encryption. In: CRYPTO '10: Annual International Cryptology Conference on Advances in Cryptology. (2010) 483–501
29. Gentry, C.: Fully Homomorphic Encryption using Ideal Lattices. In: STOC '09: ACM symposium on Theory of computing. (2009) 169–178
30. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully Homomorphic Encryption over the Integers. In: EUROCRYPT '10: Annual International Conference on Advances in Cryptology. (2010) 24–43
31. Gentry, C.: Toward Basing Fully Homomorphic Encryption on Worst-Case Hardness. In: CRYPTO '10: Annual International Cryptology Conference on Advances in Cryptology. (2010) 116–137
32. Juels, A., Kaliski, Jr., B.S.: Pors: proofs of retrievability for large files. In: CCS '07: ACM conference on Computer and communications security. (2007) 584–597
33. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: EUROCRYPT '99: Annual International Conference on Advances in Cryptology. (1999) 223–238
34. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-Preserving Symmetric Encryption. In: EUROCRYPT '09: Annual International Conference on Advances in Cryptology. (2009) 224–241
35. Shi, E., Bethencourt, J., Chan, T.H.H., Song, D., Perrig, A.: Multi-Dimensional Range Query over Encrypted Data. In: SP '07: IEEE Symposium on Security and Privacy. (2007) 350–364
36. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Order preserving encryption for numeric data. In: SIGMOD '04: ACM SIGMOD international conference on Management of data. (2004) 563–574