Zero-Knowledge Elementary Databases with Function Queries

Xinxuan Zhang[1,2] and Yi Deng[1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{zhangxinxuan, deng}@iie.ac.cn

Abstract. Zero-knowledge elementary databases (ZK-EDBs) enable a prover to commit a database D of key-value pairs and later prove that "x belongs to the support of D and $D(x) = v$" or that "x does not belong to the support of D," without revealing any extra knowledge (including the size of D). Recently, Libert *et al.* (PKC 2019) introduced zero-knowledge expressive elementary databases (ZK-EEDBs) that support richer queries, e.g., range queries over the keys and values.

In this paper, we introduce a new notion called function queriable ZK-EDBs, where the ZK-EDB prover can convincingly answer the query that "*Send all records* (x, v) *in* D *satisfying* $f(x, v) = 1$ *for any Boolean circuit* f," without revealing any extra knowledge (including the size of D).

To construct function queriable ZK-EDBs, we introduce a new variation of zero-knowledge sets (ZKS) which supports verifiable set operations, and present a construction based on groups of unknown order. By transforming the Boolean circuit over databases into the set operation circuit/formula over sets, we present a construction of function queriable ZK-EDBs from standard ZK-EDBs and ZKS supporting verifiable set operations.

1 Introduction

Zero-knowledge sets (ZKS) are a valuable primitive introduced by Micali et al. [MRK03], which enable a prover to commit a finite set S and later prove the membership or non-membership of a certain element without revealing any extra knowledge (including the size of the set). Elementary database (EDB) D is a set with an additional property that each x in the set comes with a *unique* associated value $v = D(x)$. As pointed out in [MRK03], the notion of ZKS can be extended to the one called zero-knowledge elementary databases (ZK-EDBs), which allows a prover to commit an EDB D and later prove that "x belongs to the support of D and $D(x) = v$" or that "x dose not belong to the support of D" in zero-knowledge. Numerous studies on ZK-EDBs have since been developed, including mercurial commitments [CHL+05], updatable ZK-EDBs [Lis05] and independent

ZK-EDBs [GM06]. However, because most constructions follow the paradigm put forward by Chase *et al.* [CHL+05], relying on a Merkle tree and mercurial commitment and making the support of richer queries difficult to achieve, only a few studies have focused on queries supported by a ZK-EDB.

Libert *et al.* [LNTW19] recently introduced zero-knowledge expressive elementary databases (ZK-EEDBs) supporting richer queries such as for a) all records $(x, v) \in D$ whose key x lies within the range $[a_x, b_x]$ of a super-polynomial length, b) all records $(x, v) \in D$ whose value v lies within the range $[a_y, b_y]$ of a super-polynomial length, and c) all records $(x, v) \in D$ whose element x lies within the range $[a_x, b_x]$ of a super-polynomial length and value v lies within the range $[a_y, b_y]$ of a polynomial length. Several more interesting queries such as k-nearest neighbor and k-minimum/maximum can be implemented using the above queries.

However, there are still numerous queries that remain unsupported by ZK-EEDBs, for example, querying for all records $(x, v) \in D$ whose last bit of value v equals zero, or in a more generalized form,

Querying for all records $(x, v) \in D$ *satisfying* $f(x, v) = 1$ *for any Boolean circuit* f*.*

1.1 Our contribution

In this paper, we study the queries supported by ZK-EDBs, and introduce a new notion called function queriable ZK-EDBs, which enable one to commit an elementary database D of key-value pairs $(x, v) \in \{0,1\}^l \times \{0,1\}^l$ and then convincingly answer the query "*Send all records* $(x, v) \in D$ *satisfying* $f(x, v) = 1$ *for any Boolean circuit* $f : \{0,1\}^{2l} \to \{0,1\}$", without revealing any extra knowledge (including the size of D).

To construct a function queriable ZK-EDB, we additionally introduce a new variation of ZKS, which allow one to commit to several sets $\{S_i\}_{i\in[m]}$, and then convincingly answer the extra queries that "*Send all element* x *in* $\mathcal{Q}(\mathsf{S}_1, \cdots, \mathsf{S}_m)$ *for any 'circuit'* $\mathcal{Q}$ *of unions, intersections and set-differences*", without revealing any extra knowledge. We call this variety ZKS supporting verifiable set operations. Building on the standard ZKS protocols constructed by Prabhakaran and Xue [PX09, XLL07, XLL08], we construct a ZKS supporting verifiable set operations based on groups of unknown order, which is secure in the random oracle model and the generic group model.

From ZKS supporting verifiable set operations and standard ZK-EDBs, we present a construction of function queriable ZK-EDBs. This is *the first* construction of ZK-EDBs enabling *generalized queries*. To the best of our knowledge, prior to our approach, the richest queries achieved were only range queries over the keys and values [LNTW19].

We additionally present a construction of standard ZK-EDBs from groups of unknown-order in Appendix.D, which is *the first* construction achieving a

constant commitment and proof size[3]. Applying this ZK-EDB and our ZKS supporting verifiable set operations, the resulting function queriable ZK-EDB is secure in the random oracle model and generic group model and its proof size is only linear to l and the size of circuit f.

1.2 Related Work

Since the notion of ZK-EDB was first introduced by Micali *et al.*[MRK03], numerous approaches concentrating on the performance, security, and functionality of ZK-EDB have been developed.

In [CHL+05], Chase *et al.* introduced the notion of mercurial commitments and presented a widely used paradigm of how to use such a commitment and a Merkel tree to construct a ZK-EDB. Mercurial commitments (and thus ZK-EDBs) can be constructed through one-way functions [CDV06], and efficient mercurial commitments (and thus efficient ZK-EDBs) can be constructed through DL, Factoring, RSA or LWE assumption [CDV06, Zhu09, LNTW19]. Catalano *et al.* [CFM08] introduced the concept of q-mercurial commitments and compressed the non-membership proof size of ZK-EDBs. Libert *et al.* [LY10] and Catalano *et al.* [CF13] developed q-mercurial commitments and further compressed the membership proof size of ZK-EDBs. Li *et al.* [LSY+21] introduced concise mercurial subvector commitments and achieved batch verifiable ZK-EDBs.

There are also several studies on developing the security definition of ZK-EDBs or more generalized, database commitments. Gennaro and Micali [GM06] introduced the notion of independent ZK-EDBs, which prevent the adversary from correlating its set to that of a honest prover. Chase and Visconti [CV12] introduced the notion of secure database commitments, which satisfy simulation security in the real/ideal paradigm and achieve stronger verifier security than ZK-EDBs. Prabhakaran and Xue [PX09] introduced statistically hiding set, providing an information theoretic hiding guarantee, rather than one based on efficient simulation. They provided the first construction that does not rely on Chase's paradigm. Following [PX09], [XLL07, XLL08] constructed ZKS achieving a constant commitment and proof size.

Another research point of ZK-EDBs is how to extend its functionality. Liskov [Lis05] presented updatable ZK-EDBs in the random oracle model. Ghosh *et al.* [GOT15] introduced zero-knowledge lists, which allow one to commit a list and later answer order queries in a convincing manner. Libert *et al.* [LNTW19] recently introduced ZK-EEDBs that support richer queries, e.g., range queries over the keys and values.

Accumulators (e.g. [BdM93, CL02, CHKO08, Ngu05, DHS15]) are an extremely well-studied cryptography primitive related to ZKS. Accumulators allow one to represent a set using an accumulation value, and later provide (non-

[3] Although [PX09] shows how to use a constant-size ZKS[PX09, XLL07, XLL08] to implement a constant-size ZK-EDB, the resulting scheme is not a standard ZK-EDB because the resulting scheme cannot ensure the uniqueness of the value associated to any element.

)membership proofs; however, *hiding and zero-knowledge properties are not necessary for accumulators*. Although Ghosh *et al.* [GOP+16] and Zhang *et al.* [ZKP17] presented the constructions of zero-knowledge accumulators supporting set operations, their schemes only consider collision-freeness security, where the adversary cannot cheat in a proof for an *honestly* generated accumulation value; however, ZKS require the prevention of the adversary from cheating in a proof even for a *maliciously* generated commitment. Boneh *et al.* [BBF19] recently developed a batching technique for accumulators based on groups of unknown-order. Agrawal and Raghuraman [AR20] presented a commitment construction for databases of key-value pairs enabling (non-)membership proof and efficient updating. Their scheme is also based on groups of unknown-order and does not consider hiding or zero-knowledge properties.

An authenticated data structure (ADS) (e.g., [Tam03, PTT11, NZ15]) also allows a trusted database owner to "commit" its database, and an untrusted server can answer the queries on behalf of trusted database owners to any clients knowing the commitment. However, as a three-party scheme in which the committer (database owner) is always trusted, ADS is incomparable to ZK-EDB.

1.3 Organization

In section 2 we provide an overview of our techniques. Preliminaries are described in section 3. In section 4 we introduce and construct ZKS supporting verifiable set operations. In section 5 we introduce the notion of function queriable ZK-EDBs and present a concrete construction. Finally, in section 6, we provide some concluding remarks.

2 Technique Overview

Zero-Knowledge Sets Supporting Verifiable Set Operations. ZKS supporting verifiable set operations allow a committer to generate a sequence of commitments $\{com_i\}_{i\in[t]}$ to sets $\{S_i\}_{i\in[t]}$. Then for any general set operation $\mathcal{Q}$, i.e., a "circuit" of unions, intersections and set-differences, the committer can output $S_{output} = \mathcal{Q}(\{S_i\}_{i\in[t]})$ and a public verifiable proof π for the correctness of result. In this paper, we require ZKS supporting verifiable set operations that additionally satisfy functional binding and zero-knowledge properties.

To present a construction, we first construct a ZKS that supports a single set operation (i.e., one union, intersection or set-difference) over two committed sets, with the exclusion that, for any query, we now require the committer to output a commitment to S_{output} rather than output S_{output} directly. Such a scheme can be naturally extended to our target approach by invoking it iteratively in the support of general set-operation formulas and opening the last commitment to S_{output} to obtain the set.

However, most ZKS constructions follow Chase's paradigm which relies on a Merkle tree and mercurial commitments, and makes set operations difficult to conduct. We therefore decide to build our scheme on top of the ZKS(SHS)

schemes of Prabhakaran and Xue [PX09, XLL07, XLL08], which are based on RSA accumulators rather than Chase's paradigm. For a set $S = \{x_i\}_{i\in[m]}$, the authors hash element x_i into large a prime p_i (i.e., $p_i = \mathcal{H}_{prime}(x_i)$). A ZKS commitment of S is then $\mathsf{g}^{r\Pi_{i\in[m]}p_i}$, where g is a random element from an RSA group and r is a random coin used to hide $\Pi_{i\in[m]}p_i$. It is easy to extend RSA groups to generalized groups of unknown-order by using a proper Σ-protocol as a (non-)membership proof.

Consider a single operation of intersections. Upon inputting two ZKS commitments $\mathsf{C}_0, \mathsf{C}_1$ to the sets S_0, S_1, the prover needs to output a commitment C_I to the set $I = S_0 \cap S_1$ and an associated proof π. Note that to prove $I = S_0 \cap S_1$, one only needs to show that a) $I \subset S_0$, $I \subset S_1$ and b) $(S_0 \backslash I) \cap (S_1 \backslash I) = \emptyset$. Generate two commitments $\mathsf{C}_{J_0}, \mathsf{C}_{J_1}$ to the sets $J_0 = S_0 \backslash I, J_1 = S_1 \backslash I$. In a simple case in which all random coins used in all of these commitments equal 1, the prover can easily conclude the proof by showing that a) $(\mathsf{C}_I, \mathsf{C}_{J_0}, \mathsf{C}_{S_0})$, $(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_1})$ are DDH tuples and b) the exponents of C_{J_0} and C_{J_1} are relatively prime. The challenge comes from the generalized case in which the random coins are chosen from a proper range, i.e., $[0, B]$. Luckily, we can let B be much smaller than these representing primes, which means that a) $(\mathsf{C}_I, \mathsf{C}_{J_0}, \mathsf{C}_{S_0})$ (and $(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_1})$) is actually close to a DDH tuple and b) the great common division of the exponents of C_{J_0} and C_{J_1} is small. We will show that the prover can still prove the above two statements by relying on Boneh's PoKE (Proof of knowledge of exponent) protocol [BBF19] and a classic zero-knowledge proof for small exponents. [CS97, FO97]. Other single set operations can also be similarly conducted.

Function Queriable Zero-knowledge Elementary Databases. Function queriable ZK-EDBs allow one to commit a database D of key-value pairs and later convincingly answer the query "*Send all records* (x, v) *in* D *satisfying* $f(x, v) = 1$ *for any Boolean circuit* f", without revealing any extra knowledge (including the size of D). Function queriable ZK-EDBs should satisfy functional binding and zero-knowledge properties.

In this paper, we present the construction of function queriable ZK-EDBs from standard ZK-EDBs and ZKS supporting verifiable set operations. The main idea here is to convert a Boolean circuit into a set-operation formula.

Consider a Boolean circuit f consisting of wires and gates. Upon inputting a concrete (x, v) in D, each wire in circuit f will have a deterministic value in $\{0, 1\}$. Now, let each wire associate two subsets of database D, the set of (x, v) that makes this wire equal to 0 and the set of (x, v) that makes this wire equal to 1. Then, for any gate in f, the sets associated with the output wire can be represented as a set-operation formula over the sets associated with the input wires. For example, consider an "AND" gate with two input wires a, b and an output wire c. Denote by A_b (res. B_b, C_b) the set of (x, v) that makes wire a (res. b, c) equal to $b \in \{0, 1\}$. We then have $C_0 = A_0 \cup B_0, C_1 = A_1 \cap B_1$. In this way, we can write the set of (x, v) satisfying $f(x, v) = 1$ using the set-operation formula over the sets associated with the input wire.

Therefore, we use ZKS supporting verifiable set operations to commit the sets $S_{i,b}$, where $S_{i,b} = \{x | (x, v) \in D \wedge$ the i-th bit of $x||v$ is $b\}$. We also use a

ZK-EDB scheme to commit D. Upon input a query f, the prover converts it into a set-operation formula $\mathcal{Q}$ over $S_{i,b}$, then outputs $S_{output} = Q(\{S_{i,b}\})$ and the corresponding proofs (for which we need a ZKS supporting set operations), and finally opens each $x \in S_{output}$ using the ZK-EDB commitment to obtain the associated value v.

3 Preliminaries

In this paper, we denote by λ the security parameter and by $[m]$ the set $\{1, 2, \cdots, m\}$. A non-negative function $f : \mathbb{N} \to \mathbb{R}$ is negligible if $f(\lambda) = \lambda^{-w(1)}$. We use the standard abbreviation PPT to denote probabilistic polynomial time.

An elementary database D is a set of key-value pairs $(x, v) \in \{0,1\}^l \times \{0,1\}^l$ such that if $(x, v) \in D$ and $(x, v') \in D$, then $v = v'$. Here l is a public polynomial in λ. We denote by $Sup(D)$ the support of D, i.e., the set of $x \in \{0,1\}^l$ for which $\exists v$ such that $(x, v) \in D$. We denote such unique v as $D(x)$, and if $x \notin Sup(D)$, we then also write $D(x) = \perp$. For consistency, for any set S of elements $x \in \{0,1\}^l$, we write $S(x) = 1$ if $x \in S$ and write $S(x) = \perp$ if $x \notin S$.

3.1 Zero-Knowledge Elementary Databases and Sets

ZKS allow one to commit a set S and later prove the (non-)membership of certain elements without revealing any extra knowledge. The notion of ZKS can be extended to ZK-EDBs, which allow one to commit an elementary database D. Because a ZKS can be seen as a special case of a ZK-EDB, where $D(x) = 1$ if $x \in Sup(D)$, we skip the definition of ZKS here. Following [MRK03, GM06, LNTW19], we present the following formal definition of ZK-EDBs:

Definition 1 (Zero-Knowledge Elementary Database). *A zero-knowledge elementary database consists of four algorithms* (Setup, Com, Prove, Verify)*:*

- $\delta \leftarrow \mathsf{Setup}(1^\lambda)$*: On input the security parameter* 1^λ, Setup *outputs a random string (or a structured reference string)* δ *as the CRS.*
- $(com, \tau) \leftarrow \mathsf{Com}(\delta, D)$*: On input the CRS* δ *and an elementary database* D*,* Com *outputs a commitment of database com and an opening information* τ*.*
- $\pi \leftarrow \mathsf{Prove}(\delta, com, \tau, x, v)$*: On input the CRS* δ*, the pairing of the commitment and opening information* (com, τ)*, and a key* x *and its associated value* v *(i.e.,* $(x, v) \in D$ *or* $x \notin Sup(D), v = \perp$*),* Prove *outputs a proof* π *of* $v = D(x)$
- $0/1 \leftarrow \mathsf{Verify}(\delta, com, x, v, \pi)$*: On input the CRS* δ*, commitment com, key-value pair* (x, v) *and proof* π*,* Verify *either outputs* 1 *(denoting accept) or* 0 *(denoting reject).*

It satisfies the following three properties:

- **Completeness:** *For any elementary database* D *and any* x*,*

$$\Pr\left[\mathsf{Verify}(\delta, com, x, v, \pi) = 1 \;\middle|\; \begin{array}{c}\delta \leftarrow \mathsf{Setup}(1^\lambda); (com, \tau) \leftarrow \mathsf{Com}(\delta, D); \\ \pi \leftarrow \mathsf{Prove}(\delta, com, \tau, x, D(x))\end{array}\right] = 1$$

- **Soundness:** *For any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that:*

$$\Pr\left[\begin{array}{c} v \neq v' \wedge \\ \mathsf{Verify}(\delta, com, x, v, \pi) = 1 \wedge \\ \mathsf{Verify}(\delta, com, x, v', \pi') = 1 \end{array} \middle| \begin{array}{c} \delta \leftarrow \mathsf{Setup}(1^\lambda); \\ (com, x, v, v', \pi, \pi') \leftarrow \mathcal{A}(\delta) \end{array}\right] \leq negl(\lambda)$$

- **Zero-Knowledge:** *There exists a simulator Sim such that for any PPT adversary $\mathcal{A}$, the absolute value of the difference*

$$\Pr\left[\mathcal{A}^{\mathsf{Prove}(\delta, com, \tau, \cdot, D(\cdot))}(\delta, state_{\mathcal{A}}, com) = 1 \middle| \begin{array}{c} \delta \leftarrow \mathsf{Setup}(1^\lambda), \\ (D, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ (com, \tau) \leftarrow \mathsf{Com}(\delta, D) \end{array}\right] -$$

$$\Pr\left[\mathcal{A}^{Sim(state_S, \cdot, D(\cdot))}(\delta, state_{\mathcal{A}}, com) = 1 \middle| \begin{array}{c} (\delta, state_\delta) \leftarrow Sim(1^\lambda), \\ (D, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ (com, state_S) \leftarrow Sim(\delta, state_\delta) \end{array}\right]$$

is negligible in λ.

3.2 Groups of Unknown-Order and Assumptions

In this paper, the schemes are constructed on groups of unknown order, for which the order is difficult to compute for the committer. Groups of unknown order are a useful tool in the construction of polynomial commitments, integer commitments, and accumulators, among other aspects.

The strong RSA assumption is a useful assumption for groups of unknown orders. We introduce it in the following.

Assumption 1 *(Strong RSA Assumption)[BP97, AR20]. The strong RSA assumption states that an efficient adversary cannot compute lth roots for a given random group element, where l is an odd prime chosen by the adversary. Specifically, it holds for GGen if for any probabilistic polynomial time adversary $\mathcal{A}$,*

$$\Pr\left[\mathsf{u}^l = \mathsf{g} \text{ and } l \text{ is an odd prime} \middle| \begin{array}{c} \mathbb{G} \leftarrow GGen(\lambda), \mathsf{g} \xleftarrow{\$} \mathbb{G}, \\ (\mathsf{u}, l) \in \mathbb{G} \times \mathbb{N} \leftarrow \mathcal{A}(\mathbb{G}, \mathsf{g}) \end{array}\right] \leq negl(\lambda).$$

Generic group model. In this paper, we use the generic group model for groups of unknown order as defined by Damgard and Koprowski[DK02], and as used in [BBF19]. Portions of the definition of the generic group model are taken verbatim from [BBF19].

In the generic group model, the group is parameterized by two public integers A and B, and the group order is sampled uniformly from $[A, B]$. The group $\mathbb{G}$ is defined by a random injective function $\sigma : \mathbb{Z}_{|\mathbb{G}|} \to \{0,1\}^l$ for some l, where $2^l \gg |\mathbb{G}|$. The group elements are $\sigma(0), \cdots, \sigma(|\mathbb{G}|)$. A generic group algorithm $\mathcal{A}$ is a probabilistic algorithm. Let $\mathcal{L}$ be a list that is initialized with the encodings

given to $\mathcal{A}$ as input. $\mathcal{A}$ can query two generic group oracles. The first oracle $\mathcal{O}_1$ samples a random $r \in \mathbb{Z}_{|\mathbb{G}|}$ and returns $\sigma(r)$, which is appended to the list of encodings $\mathcal{L}$. The second oracle $\mathcal{O}_2(i, j, \pm)$ takes two indices $i, j \in [p]$, where p is the size of $\mathcal{L}$, as well as a sign bit, and returns $\sigma(i \pm j)$, which is appended to $\mathcal{L}$. Note that herein $\mathcal{A}$ is not given the order of $\mathbb{G}$.

As shown in [DK02], the strong RSA assumption holds in the generic group model.

Proof of Knowledge of Exponent (PoKE). Recently, Boneh *et al.* [BBF19] introduced a way to present an argument of knowledge protocol for the following relation.

$$\mathcal{R}_{PoKE} := \{(\mathsf{u}, \mathsf{w} \in \mathbb{G}; x \in \mathbb{Z}) | \mathsf{w} = \mathsf{u}^x \in \mathbb{Z}\}$$

Let $\mathcal{P}$ be the prover and $\mathcal{V}$ be the verifier. Their protocol is as follows:

Protocol PoKE (Proof of knowledge of exponent)

Params: $\mathbb{G} \leftarrow GGen(\lambda), \mathsf{g} \in \mathbb{G}$; Input: $\mathsf{u}, \mathsf{w} \in \mathbb{G}$; Witness: $x \in \mathbb{Z}$

1. $\mathcal{P}$ sends $\mathsf{z} = \mathsf{g}^x \in \mathbb{G}$ to $\mathcal{V}$.
2. $\mathcal{V}$ samples $l \leftarrow \mathsf{Primes}(\lambda)$ and sends l to $\mathcal{P}$.
3. $\mathcal{P}$ finds the quotient $q \in \mathbb{Z}$ and residue $r \in [l]$ such that $x = ql + r$. $\mathcal{P}$ sends $\mathsf{Q} = \mathsf{u}^q$, $\mathsf{Q}' = \mathsf{g}^q$, and r to $\mathcal{V}$.
4. $\mathcal{V}$ accepts if $r \in [l]$, $\mathsf{Q}^l \mathsf{u}^r = \mathsf{w}$, and $\mathsf{Q}'^l \mathsf{g}^r = \mathsf{z}$.

Fig. 1: PoKE protocol [BBF19]

Theorem 1 ([BBF19] Theorem 3.). Protocol PoKE *is an argument of knowledge for the relation* $\mathcal{R}_{PoKE}$ *in the generic group model.*

In practice, there are two common methods used to instantiate groups of unknown order.

RSA group: The multiplicative group $\mathbb{Z}_n^*$ of integers modulo a product $n = pq$ of large primes p and q. Any efficient algorithm that calculates the order can be transformed into an efficient algorithm factoring n. In addition, we need to point out that it is difficult to generate the RSA group in a publicly verifiable way without exposing the order. Therefore, we need a trusted party to generate the group.

Class group: The class group of an imaginary quadratic order with discriminant Δ where $-\Delta$ is a prime and $\Delta \equiv 1 \mod 4$. Aa an important property, one can choose a security class group $Cl(\Delta)$ by choosing the "good" discriminant Δ randomly without a trusted party. For more details, one can refer to Buchmann and Hamdy's survey[BH01] and Straka's accessible blog post[Str19] for more details.

At the end of this section, we provide several simple but widely used lemmas.

Lemma 1. *For any positive integers a, A, and B satisfying $B > A$, we have:*

$$\mathsf{Dist}(\{x \leftarrow [a]\}, \{x \mod a | x \leftarrow [A, B]\}) \leq \frac{a}{B - A}$$

where Dist *indicates the distance between distributions.*

Lemma 2. *For any integers s_1,s_2 and positive integers a,A,B satisfying $B > A$,* $\gcd(s_1, s_2) = 1$*, we have:*

$$\mathsf{Dist}(\{x \leftarrow [a]\}, \{xs_1 + ys_2 \mod a | x, y \leftarrow [A, B]\}) \leq \frac{3a}{B - A}$$

where Dist *indicates the distance between distributions.*

Lemma 3. *For any multiplicative group $\mathbb{G}$ and group elements $\mathsf{g}, \mathsf{h} \in \mathbb{G}$, if there exists coprime integers a, p satisfying $\mathsf{g}^a = \mathsf{h}^p$, then one can easily compute h' satisfying $\mathsf{g} = \mathsf{h}'^p$ from a, p, g and h.*

The proofs of the above three lemmas are shown in Appendix.A.

4 Zero-Knowledge Set with Verifiable Set Operations

In this section, we introduce the notion of ZKS supporting verifiable set operations, which is the key ingredient for our final goal, that is, function queriable zero-knowledge elementary databases. We then give several *weak* zero-knowledge argument of knowledge protocols[4] for several special statements over groups of unknown orders and finally present the construction of ZKS supporting verifiable set operations.

4.1 Definition

Informally, ZKS supporting verifiable set operations allow one to commit to several sets $\{S_i\}_{i \in [m]}$, and then convincingly answer a) the (non-)membership queries on each set and b) the queries in the form as "*Send all records (x, v) in $\mathcal{Q}(S_1, \cdots, S_m)$ for any set operation $\mathcal{Q}$ represented as a formula of unions, intersections and set-differences*," without revealing any extra knowledge. The formal definition of ZKS supporting verifiable set operations is as follows:

Definition 2 (ZKS Supporting Verifiable Set Operations). *A ZKS supporting verifiable set operations consists of six algorithms,* (Setup, Com, Prove, Verify, ProveS, VerifyS)*, where* Setup, Com, Prove, Verify *are in the same form as a standard ZKS and*

[4] These zero-knowledge protocols only satisfy a type of special purpose knowledge soundness, which are sufficient for our construction of ZKS supporting verifiable set operations.

- $\pi \leftarrow \mathsf{ProveS}(\delta, \widetilde{com}, \widetilde{\tau}, \mathcal{Q}, S_{output})$*: On input the CRS* δ*, the list of set commitments and the associated opening information* $\widetilde{com} = (com_1, \cdots, com_m), \widetilde{\tau} = (\tau_1, \cdots, \tau_m)$ *where* $(com_i, \tau_i) \in \mathsf{Com}(\delta, S_i)$*, a set operation* $\mathcal{Q}$ *represented as a "circuit"/formula of unions, intersections and set-differences and the target output set* S_{output}*.* Prove *outputs a proof* π *of* $S_{output} = \mathcal{Q}(S_1, \cdots, S_m)$*.*
- $0/1 \leftarrow \mathsf{VerifyS}(\delta, \widetilde{com}, \mathcal{Q}, S_{output}, \pi)$*: On input the CRS* δ*, the list of commitments* $\widetilde{com}$*, the set operation* $\mathcal{Q}$*, the target output* S_{output}*, and the proof* π*,* Verify *either outputs* 1 *(denoting accept) or* 0 *(denoting reject).*

and satisfies the following properties:

- **(Non-)Membership Query Completeness:** *For any set* S *and any* x*,*

$$\Pr\left[\mathsf{Verify}(\delta, com, x, S(x), \pi) = 1 \;\middle|\; \begin{array}{c}\delta \leftarrow \mathsf{Setup}(1^\lambda); (com, \tau) \leftarrow \mathsf{Com}(\delta, S); \\ \pi \leftarrow \mathsf{Prove}(\delta, com, \tau, x, S(x))\end{array}\right] = 1$$

- **Set Operation Query Completeness:** *For any sets* $\{S_i\}_{i\in[m]}$ *and a set operation* $\mathcal{Q}$ *which takes* m *sets as input and outputs one set, let* $S_{output} = \mathcal{Q}(S_1, \cdots, S_m)$*, and thus*

$$\Pr\left[\mathsf{VerifyS}(\delta, \widetilde{com}, \mathcal{Q}, S_{output}, \pi) = 1 \;\middle|\; \begin{array}{c}\delta \leftarrow \mathsf{Setup}(1^\lambda); \\ \forall i \in [m], (com_i, \tau_i) \leftarrow \mathsf{Com}(\delta, S_i); \\ \pi \leftarrow \mathsf{ProveS}(\delta, \widetilde{com}, \widetilde{\tau}, \mathcal{Q}, S_{output})\end{array}\right] = 1$$

  where $\widetilde{com} = (com_1, \cdots, com_m)$ *and* $\widetilde{\tau} = (\tau_1, \cdots, \tau_m)$*.*
- **Functional Binding:** *For any PPT adversary* $\mathcal{A}$*, the probability that* $\mathcal{A}$ *wins the following game is negligible:*
  1. *The challenger generates a CRS* δ *by running* $\mathsf{Setup}(1^\lambda)$ *and gives* δ *to the adversary* $\mathcal{A}$*.*
  2. *The adversary* $\mathcal{A}$ *outputs a set of commitments* $\{com_i\}_{i\in[m]}$ *and the following tuples:*
     (a) *A series of (non-)membership query-proof tuples* $\{(com'_j, x_j, v_j, \pi_j)\}_{j\in[n_1]}$*, where* $com'_j \in \{com_i\}_{i\in[m]}$ *(suppose that* $com'_j = com_{t_j}$*).*
     (b) *A series of set operation query-proof tuples* $\{(\widetilde{com}_j, \mathcal{Q}_j, S'_j, \pi'_j)\}_{j\in[n_2]}$*, where* $\widetilde{com}_j$ *is a list of commitments contained in* $\{com_i\}_{i\in[m]}$ *(supposing that* $\widetilde{com}_j = (com_{t_{j1}}, com_{t_{j2}}, \cdots)$*).*
  3. *The adversary* $\mathcal{A}$ *wins the game if the following hold:*
     (a) *For each* $j \in [n_1]$*,* $\mathsf{Verify}(com'_j, x_j, v_j, \pi_j) = 1$
     (b) *For each* $j \in [n_2]$*,* $\mathsf{Verify}(\widetilde{com}_j, \mathcal{Q}_j, S'_j, \pi'_j) = 1$*.*
     (c) *There do* ***not*** *exist sets* $\{S_i\}_{i\in[m]}$ *satisfying* $S_{t_j}(x_j) = v_j$ *for each* $j \in [n_1]$ *and* $\mathcal{Q}_j(\widetilde{S}_j) = S'_j$ *for each* $j \in [n_2]$*, where* $\widetilde{S}_j = (S_{t_{j1}}, S_{t_{j2}}, \cdots)$*.*

- **Zero-Knowledge:** *There exists a simulator Sim such that for any PPT adversary $\mathcal{A}$, the absolute value of the difference*

$$\Pr\left[\mathcal{A}^{\mathcal{O}_P}(\delta, state_{\mathcal{A}}, \{com_i\}_{i\in[m]}) = 1 \;\middle|\; \begin{array}{c} \delta \leftarrow \mathsf{Setup}(1^\lambda), \\ (\{S_i\}_{i\in[m]}, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ \text{for } i \in [m], (com_i, \tau_i) \leftarrow \mathsf{Com}(\delta, S_i) \end{array}\right] -$$

$$\Pr\left[\mathcal{A}^{\mathcal{O}_S}(\delta, state_{\mathcal{A}}, \{com_i\}_{i\in[m]}) = 1 \;\middle|\; \begin{array}{c} (\delta, state_\delta) \leftarrow Sim(1^\lambda), \\ (\{S_i\}_{i\in[m]}, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ (\{com_i\}_{i\in[m]}, state_S) \leftarrow Sim(\delta, m, state_\delta) \end{array}\right]$$

is negligible in λ, where $\mathcal{O}_P$ and $\mathcal{O}_S$ are defined as follows:

$\mathcal{O}_P$: *On input (com_i, x) for some $i \in [m]$, $\mathcal{O}_P$ outputs $\pi \leftarrow \mathsf{Prove}(\delta, com_i, \tau_i, x, S_i(x))$. On input $(\widetilde{com}, \mathcal{Q})$ where $\widetilde{com} = (com_{t_1}, \cdots, com_{t_n})$ for some $t_1, \cdots, t_n \in [m]$, $\mathcal{O}_P$ outputs $\pi \leftarrow \mathsf{ProveS}(\delta, \widetilde{com}, \widetilde{\tau}, \mathcal{Q}, \mathcal{Q}(S_{t_1}, \cdots, S_{t_n}))$ where $\widetilde{\tau} = (\tau_{t_1}, \cdots, \tau_{t_n})$. In other cases, $\mathcal{O}_P$ outputs $\perp$.*

$\mathcal{O}_S$: *On input (com_i, x) for some $i \in [m]$, $\mathcal{O}_S$ outputs $\pi \leftarrow Sim(\delta, com, state_S, x, S_i(x))$. On input $(\widetilde{com}, Q)$ where $\widetilde{com} = (com_{t_1}, \cdots, com_{t_n})$ for some $t_1, \cdots, t_n \in [m]$, $\mathcal{O}_S$ outputs $\pi \leftarrow Sim(\delta, \widetilde{com}, state_S, \mathcal{Q}, \mathcal{Q}(S_{t_1}, \cdots, S_{t_n}))$. In other cases, $\mathcal{O}_S$ outputs $\perp$.*

4.2 Several Zero-Knowledge Protocols over Unknown-Order Groups

In this section, we introduce several zero-knowledge protocols over the groups of unknown order, which will be used in our main constructions.

Zero-knowledge protocol for bounded discrete-log. The classical sigma protocol can be used to prove the discrete-log relation when the exponent is small (i.e., $\mathcal{R}_{boundedDL} = \{(\mathsf{u}, \mathsf{w}, T; x) | \mathsf{u}^x = \mathsf{w} \wedge |x| \leq T\}$). It only satisfies a weak soundness, which is sufficient for our goal. Following [DF02, CS97, FO97], the construction is as follows.

Protocol $\mathsf{ZK}_{boundedDL}$ (Zero-knowledge protocol for $\mathcal{R}_{boundedDL}$)

Params: $\mathbb{G} \leftarrow GGen(\lambda)$; Input: $\mathsf{u}, \mathsf{w} \in \mathbb{G}, T \in \mathbb{N}$; Witness: $x \in \mathbb{Z}$

1. $\mathcal{P}$ samples $r \leftarrow [2^{2\lambda}T]$ and sends $\mathsf{z} = \mathsf{u}^r \in \mathbb{G}$ to $\mathcal{V}$.
2. $\mathcal{V}$ sends challenge $c \leftarrow [2^\lambda]$.
3. $\mathcal{P}$ computes $s = r + cx$ and sends s to $\mathcal{V}$.
4. $\mathcal{V}$ accepts if $\mathsf{u}^s = \mathsf{z}\mathsf{w}^c$ and $s \leq 2^{2\lambda}T$.

Fig. 2: Protocol $\mathsf{ZK}_{boundedDL}$ [CS97, FO97]

Lemma 4. Protocol $\mathsf{ZK}_{boundedDL}$ *is an honest-verifier statistically zero-knowledge protocol achieving a type of weak soundness defined as follows: There exists an extractor such that for any polynomial* p *and any prover* $\mathcal{P}^*$ *convincing verifier of statement* $(\mathsf{u}, \mathsf{w}, T)$ *with probability* p^{-1}*, the extractor can extract* (x', t) *within an expected polynomial time such that* $|x'| \leq 2^{2\lambda}T, |t| \leq 2^{\lambda}$ *and* $\mathsf{u}^{x'} = \mathsf{w}^t$*.*

proof sketch. The honest-verifier statistically zero-knowledge property directly follows [DF02]. The weak knowledge soundness follows that: Suppose $\mathcal{P}^*$ can convince the verifier of the statement $(\mathsf{u}, \mathsf{w}, T)$, the extractor can then extract two valid proofs with the same first message (i.e., (z, c, s) and (z, c', s')) by rewinding. We then have $\mathsf{u}^s = \mathsf{z}\mathsf{w}^c$ and $\mathsf{u}^{s'} = \mathsf{z}\mathsf{w}^{c'}$, and therefore, $\mathsf{u}^{\Delta s} = \mathsf{w}^{\Delta c}$, where $\Delta s = s - s' \in [-2^{2\lambda}T, 2^{2\lambda}T]$ and $\Delta c = c' - c \in [-2^{\lambda}, 2^{\lambda}]$.

Zero-knowledge protocol for discrete-log. As we have shown, Boneh *et al.* [BBF19] presented a proof of knowledge of exponent protocol. Combined with the above protocol, we obtain a zero-knowledge protocol for the relation $\mathcal{R}_{DL} = \{(\mathsf{u}, \mathsf{w}; x) | \mathsf{u}^x = \mathsf{w}\}$. The construction is as follows:

Protocol ZK_{DL} (Zero-knowledge protocol for $\mathcal{R}_{DL}$)

Params: $\mathbb{G} \leftarrow GGen(\lambda), B \geq |\mathbb{G}|$; Input: $\mathsf{u}, \mathsf{w} \in \mathbb{G}$; Witness: $x \in \mathbb{Z}$

1. $\mathcal{P}$ samples $r \leftarrow [2^{2\lambda}B]$ and sends $\mathsf{z} = \mathsf{u}^r \in \mathbb{G}$ to $\mathcal{V}$.
2. $\mathcal{V}$ sends challenge $c \leftarrow [2^{\lambda}]$.
3. $\mathcal{P}$ computes $s = r + cx$.
4. $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{PoKE}(\mathsf{u}, \mathsf{z}\mathsf{w}^c; s)$.

Fig. 3: Protocol ZK_{DL}

Lemma 5. *In the generic group model,* Protocol ZK_{DL} *is an honest-verifier statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor such that for any polynomial* p*, for any prover* $\mathcal{P}^*$ *convincing the verifier of statement* (u, w) *with probability* p^{-1}*, the extractor can extract* (x', t) *within an expected polynomial time such that* $|t| \leq 2^{\lambda}$ *and* $\mathsf{u}^{x'} = \mathsf{w}^t$*.*

Proof. Completeness is obvious. In addition, the special purpose knowledge soundness follows Lemma.4 and the argument of knowledge property of PoKE (which is used to extract s) directly.

The proof of honest-verifier statistically zero-knowledge property is as follows. The simulator is constructed in the following way: On input random challenges c, l (where l is the challenge of PoKE) and statement $(\mathsf{u}, \mathsf{w} = \mathsf{u}^x)$, the simulator Sim chooses $s \leftarrow [2^{2\lambda}B]$ and computes $\mathsf{z} = \mathsf{u}^s / \mathsf{w}^c$. Then Sim runs $\mathsf{PoKE}(\mathsf{u}, \mathsf{u}^s; s)$ with challenge l honestly to conclude the proof. Therefore, for any fixed $\mathsf{u}, \mathsf{w}, c, l$, the distribution of the simulation transcript Tran_{Sim} is

$\{((\mathsf{u},\mathsf{w}),\mathsf{z}=\mathsf{u}^s/\mathsf{w}^c,c,\mathsf{Tran}_{\mathsf{PoKE}})|s\leftarrow[2^{2\lambda}B]\}$, where $\mathsf{Tran}_{\mathsf{PoKE}}=(\mathsf{g}^s,l,\mathsf{u}^{\lfloor s/l\rfloor},\mathsf{g}^{\lfloor s/l\rfloor},s \mod l)$. Similarly, for any fixed $\mathsf{u},\mathsf{w},c,l$, the distribution of the real world transcript Tran_{Real} is $\{((\mathsf{u},\mathsf{w}),\mathsf{z}=\mathsf{u}^r=\mathsf{u}^s/\mathsf{w}^c,c,\mathsf{Tran}_{\mathsf{PoKE}})|r\leftarrow[2^{2\lambda}B],s=r+cx\}$, where $\mathsf{Tran}_{\mathsf{PoKE}}=(\mathsf{g}^s,l,\mathsf{u}^{\lfloor s/l\rfloor},\mathsf{g}^{\lfloor s/l\rfloor},s \mod l)$. Denote by $\mathcal{F}_{\mathsf{u},\mathsf{w},c,l}(s)=((\mathsf{u},\mathsf{w}),\mathsf{u}^s/\mathsf{w}^c,c,(\mathsf{g}^s,l,\mathsf{u}^{\lfloor s/l\rfloor},\mathsf{g}^{\lfloor s/l\rfloor},s \mod l))$. We then have the following:

$$\mathsf{Tran}_{Sim}=\{\mathcal{F}_{\mathsf{u},\mathsf{w},c,l}(s)|s\leftarrow[2^{2\lambda}B]\}$$
$$\mathsf{Tran}_{Real}=\{\mathcal{F}_{\mathsf{u},\mathsf{w},c,l}(s)|r\leftarrow[2^{2\lambda}B],s=r+cx\}$$

As an important observation, $\mathcal{F}_{\mathsf{u},\mathsf{w},c,l}(s)=\mathcal{F}_{\mathsf{u},\mathsf{w},c,l}(s \mod l|\mathbb{G}|)$. As a result, to prove that $\mathsf{Tran}_{Sim}\overset{s}{\approx}\mathsf{Tran}_{Real}$, we only need to prove that $\{s \mod l|\mathbb{G}| \ \ |s\leftarrow[2^{2\lambda}B]\}\overset{s}{\approx}\{s \mod l|\mathbb{G}| \ \ |r\leftarrow[2^{2\lambda}B],s=r+cx\}$, which follows from Lemma.1 (note that $l\leq[2^{\lambda}]$ and $\frac{l|\mathbb{G}|}{2^{2\lambda}B}\leq 2^{-\lambda}$, therefore, these two distributions are both statistically indistinguishable from the uniform distribution over $\mathbb{Z}_{l|\mathbb{G}|}$). □

Above zero-knowledge protocol can be easily extended for the relation $\mathcal{R}_{2DL}=\{(\mathsf{u},\mathsf{v},\mathsf{w};x_1,x_2)|\mathsf{u}^{x_1}\mathsf{v}^{x_2}=\mathsf{w}\}$. We call this ZK_{2DL}. The construction is shown in Fig.4.

Protocol ZK_{2DL} (Zero-knowledge protocol for $\mathcal{R}_{2DL}$)

Params: $\mathbb{G}\leftarrow GGen(\lambda),B\geq|\mathbb{G}|$; Input: $\mathsf{u},\mathsf{v},\mathsf{w}\in\mathbb{G}$; Witness: $x_1,x_2\in\mathbb{Z}$

1. $\mathcal{P}$ samples $r_1,r_2\leftarrow[2^{2\lambda}B]$ and sends $\mathsf{z}=\mathsf{u}^{r_1}\mathsf{v}^{r_2}\in\mathbb{G}$ to $\mathcal{V}$.
2. $\mathcal{V}$ sends challenge $c\leftarrow[2^{\lambda}]$.
3. $\mathcal{P}$ computes $s_1=r_1+cx_1,s_2=r_2+cx_2$ and sends u^{s_1} to $\mathcal{V}$.
4. $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{PoKE}(\mathsf{u},\mathsf{u}^{s_1};s_1)$ and $\mathsf{PoKE}(\mathsf{v},\mathsf{zw}^c/\mathsf{u}^{s_1};s_2)$.

Fig. 4: Protocol ZK_{2DL}

Lemma 6. *In the generic group model,* Protocol ZK_{2DL} *is an honest-verifier statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor such that for any polynomial p, for any prover $\mathcal{P}^*$ convincing the verifier of statement $(\mathsf{u},\mathsf{v},\mathsf{w})$ with probability p^{-1}, then the extractor could extract (x_1',x_2',t) within an expected polynomial time such that $|t|\leq 2^{\lambda}$ and $\mathsf{u}^{x_1'}\mathsf{v}^{x_2'}=\mathsf{w}^t$.*

To prove this lemma, one can use the same proof strategy for Lemma.5. We skip this herein.

Zero-knowledge protocol for nearly-coprime exponents tuple. In this paper, we need to prove the statement that the ZKS commitments u,v to the set X,Y satisfy $X\cap Y=\emptyset$. However, owing to the existence of randomness, the exponents of u,v are not coprime. We only have the fact that the greatest

common division of these exponents is small. Therefore, we call such a tuple nearly coprime exponents tuple and denote it through the following relation:

$$\mathcal{R}_{prime} = \left\{ (\mathsf{u}, \mathsf{v}, T; a_1, a_2) \middle| \begin{matrix} \gcd(a_1, a_2) \leq T & \wedge \\ \mathsf{u} = \mathsf{g}^{a_1}, \mathsf{v} = \mathsf{g}^{a_2} & \end{matrix} \right\}$$

The zero-knowledge protocol for $\mathcal{R}_{prime}$ is as follows.

Protocol ZK_{prime} (Zero-knowledge protocol for $\mathcal{R}_{prime}$)

Params: $\mathbb{G} \leftarrow GGen(\lambda), B \geq |\mathbb{G}|$; Input: $\mathsf{u}, \mathsf{v} \in \mathbb{G}, T \in \mathbb{Z}$; Witness: $a_1, a_2 \in \mathbb{Z}$

1. $\mathcal{P}$ finds integers t_1, t_2 such that $t_1 a_1 + t_2 a_2 = \gcd(a_1, a_2)$. $\mathcal{P}$ samples $r \leftarrow [2^\lambda B]$ and sends $\mathsf{Q} = \mathsf{g}^{r \gcd(a_1, a_2)} = \mathsf{u}^{rt_1} \mathsf{v}^{rt_2}$ to $\mathcal{V}$.
2. $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{ZK}_{boundedDL}(\mathsf{g}, \mathsf{Q}, 2^\lambda BT; r \gcd(a_1, a_2))$ and $\mathsf{ZK}_{2DL}(\mathsf{u}, \mathsf{v}, \mathsf{Q}; rt_1, rt_2)$

Fig. 5: Protocol ZK_{prime}

Lemma 7. *In the generic group model, ZK_{prime} is an honest-verifier statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor such that for any polynomial p, for any prover P^* convincing the verifier with probability p^{-1} over statement $(\mathsf{u}, \mathsf{v}, T)$, the extractor can extract $t_1, t_2, c \in \mathbb{Z}$ such that $|c| \leq 2^{2\lambda} BT$ and $\mathsf{u}^{t_1} \mathsf{v}^{t_2} = \mathsf{g}^c$.*

Proof. **Completeness** is obvious.

Special purpose knowledge soundness follows from the weak knowledge soundness of $\mathsf{ZK}_{boundedDL}$ and ZK_{2DL}. From the weak knowledge soundness of $\mathsf{ZK}_{boundedDL}$, one can extract $x, t \in \mathbb{Z}$ satisfying that $|x| \leq 2^\lambda BT, |r'| \leq 2^\lambda$ and $\mathsf{Q}^{r'} = \mathsf{g}^x$. From the special purpose knowledge soundness of ZK_{2DL}, one can extract $t'_1, t'_2, r'' \in \mathbb{Z}$ satisfying $|r''| \leq 2^\lambda$ and $\mathsf{Q}^{r''} = \mathsf{u}^{t'_1} \mathsf{v}^{t'_2}$. As a result, we have $\mathsf{u}^{r' t'_1} \mathsf{v}^{r' t'_2} = \mathsf{g}^{r'' x}$. Setting $t_1 = r' t'_1, t_2 = r' t'_2, c = r'' x$, we then have $\mathsf{u}^{t_1} \mathsf{v}^{t_2} = \mathsf{g}^c$ and $|c| \leq 2^{2\lambda} BT$.

The simulator *Sim* of the **honest-verifier statistically zero-knowledge** property can be constructed as follows: *Sim* samples $r_1, r_2 \leftarrow [2^\lambda B]$ and sets $\mathsf{Q} = \mathsf{u}^{r_1} \mathsf{v}^{r_2}$. *Sim* then simulates the zero-knowledge protocol in Step 2.

Because both $\mathsf{ZK}_{boundedDL}$ and ZK_{2DL} are honest-verifier statistically zero-knowledge, we only need to show that the distributions of $(\mathsf{u}, \mathsf{v}, \mathsf{Q})$ generated by the simulator and honest prover are statistically indistinguishable. In other words, we need to show the following: For any fixed $\mathsf{u} = \mathsf{g}^{a_1}, \mathsf{v} = \mathsf{g}^{a_2}$, the distance of the distributions $\{\mathsf{g}^{r \gcd(a_1, a_2)} | r \leftarrow [2^\lambda B]\}$ and $\{\mathsf{u}^{r_1} \mathsf{v}^{r_2} = \mathsf{g}^{r_1 a_1 + r_2 a_2} | r_1, r_2 \leftarrow [2^\lambda B]\}$ are exponentially small.

Set $b = \mathsf{Ord}(\mathsf{g}^{\gcd(a_1,a_2)}) \leq B$. From Lemma.1, $\{r \bmod b | r \leftarrow [2^\lambda B]\}$ is exponentially close to the uniform distribution over $\mathbb{Z}_b$. Therefore the distribution $\{\mathsf{g}^{r\gcd(a_1,a_2)} | r \leftarrow [2^\lambda B]\} = \{(\mathsf{g}^{\gcd(a_1,a_2)})^{r \bmod b} | r \leftarrow [2^\lambda B]\}$ is exponential close to the distribution $\{(\mathsf{g}^{\gcd(a_1,a_2)})^r | r \leftarrow [b]\}$. From Lemma.2, $\{r_1 \cdot \frac{a_1}{\gcd(a_1,a_2)} + r_2 \cdot \frac{a_1}{\gcd(a_1,a_2)} \bmod b | r_1, r_2 \leftarrow [2^\lambda B]\}$ is exponentially close to the uniform distribution over $\mathbb{Z}_b$. Therefore, the distribution $\{\mathsf{g}^{r_1 a_1 + r_2 a_2} | r_1, r_2 \leftarrow [2^\lambda B]\} = \{(\mathsf{g}^{\gcd(a_1,a_2)})^{r_1 \cdot \frac{a_1}{\gcd(a_1,a_2)} + r_2 \cdot \frac{a_1}{\gcd(a_1,a_2)} \bmod b} | r_1, r_2 \leftarrow [2^\lambda B]\}$ is also exponentially close to the distribution $\{(\mathsf{g}^{\gcd(a_1,a_2)})^r | r \leftarrow [b]\}$, which concludes the lemma.

Zero-knowledge protocol for DDH-type tuples. In this paper, we need to prove that the ZKS commitments $\mathsf{u}, \mathsf{v}, \mathsf{w}$ to the set A, B, C satisfy $C = A \cup B$ and $A \cap B = \emptyset$. We have already shown how to prove $A \cap B = \emptyset$, and herein we show how to prove $C = A \cup B$. Here, we call such a tuple $(\mathsf{u}, \mathsf{v}, \mathsf{w})$ as a DDH-type tuple and denote it through the following relation:

$$\mathcal{R}_{DDH-type} = \left\{ (\mathsf{u}, \mathsf{v}, \mathsf{w}, T; x, y, a_1, a_2, a_3) \middle| \begin{array}{ll} |a_1|, |a_2|, |a_3| \leq T & \wedge \\ \mathsf{gcd}(xy, \Pi_{i=1}^{2^\lambda |\mathbb{G}|} i) = 1 & \wedge \\ \mathsf{u} = \mathsf{g}^{a_1 x}, \mathsf{v} = \mathsf{g}^{a_2 y}, \mathsf{w} = \mathsf{g}^{a_3 xy} & \end{array} \right\}.$$

Roughly, a DDH-type tuple $(\mathsf{u}, \mathsf{v}, \mathsf{w}, T)$ satisfies that, if a prime $p > T$ divides $log_{\mathsf{g}}\mathsf{u}$ or $log_{\mathsf{g}}\mathsf{v}$, then p also divides $log_{\mathsf{g}}\mathsf{w}$. Building on Boneh's (NI-)PoDDH protocol, a zero-knowledge protocol for DDH-type tuples is as follows:

Protocol $\mathsf{ZK}_{DDH-type}$ (Zero-knowledge protocol for $\mathcal{R}_{DDH-type}$)

Params: $\mathbb{G} \leftarrow GGen(\lambda), B \geq |\mathbb{G}|$; Input: $\mathsf{u}, \mathsf{v}, \mathsf{w} \in \mathbb{G}, T$; Witness: $a_1, a_2, a_3, x, y \in \mathbb{Z}$

1. $\mathcal{P}$ samples $r_1, r_2 \leftarrow [2^{2\lambda} B]$ and sends $\mathsf{u}' = \mathsf{g}^{r_1 x}, \mathsf{v}' = \mathsf{g}^{r_2 y}, \mathsf{w}' = \mathsf{g}^{r_1 r_2 xy}$ to $\mathcal{V}$.
2. $\mathcal{V}$ sends $l_1 \leftarrow \mathsf{Primes}(\lambda), l_2 \leftarrow \mathsf{Primes}(\lambda)$.
3. $\mathcal{P}$ finds the quotient $q_1, q_2 \in \mathbb{Z}$ and residue $t_1 \in [l_1], t_2 \in [l_2]$ such that $r_1 x = q_1 l_1 + t_1$ and $r_2 y = q_2 l_2 + t_2$. $\mathcal{P}$ sends $\mathsf{Q}_1 = \mathsf{g}^{q_1}$, $\mathsf{Q}_1' = \mathsf{v}'^{q_1}$ and $\mathsf{Q}_2 = \mathsf{g}^{q_2}$, t_1 and t_2 to $\mathcal{V}$.
4. $\mathcal{V}$ checks $t_1 \in [l_1], t_2 \in [l_2]$, $\mathsf{Q}_1^{l_1} \mathsf{g}^{t_1} = \mathsf{u}'$, $\mathsf{Q}_1^{l_1} \mathsf{v}'^{t_1} = \mathsf{w}'$, and $\mathsf{Q}_2^{l_2} \mathsf{g}^{t_2} = \mathsf{v}'$.
5. $\mathcal{P}$ samples $r_u, r_v, r_w \leftarrow [2^\lambda B]$ and sends $\mathsf{u}'' = \mathsf{u}^{r_1 r_u}, \mathsf{v}'' = \mathsf{v}^{r_2 r_y}, \mathsf{w}'' = \mathsf{w}^{r_1 r_2 r_w}$ to $\mathcal{V}$.
6. $\mathcal{P}$ and $\mathcal{V}$ run $\mathsf{ZK}_{boundedDL}(\mathsf{u}, \mathsf{u}'', 2^{3\lambda} B^2; r_1 r_u)$, $\mathsf{ZK}_{boundedDL}(\mathsf{u}', \mathsf{u}'', 2^\lambda BT; a_1 r_u)$, $\mathsf{ZK}_{boundedDL}(\mathsf{v}, \mathsf{v}'', 2^{3\lambda} B^2; r_2 r_v)$, $\mathsf{ZK}_{boundedDL}(\mathsf{v}', \mathsf{v}'', 2^\lambda BT; a_2 r_v)$, $\mathsf{ZK}_{boundedDL}(\mathsf{w}, \mathsf{w}'', 2^{5\lambda} B^3; r_1 r_2 r_w)$ and $\mathsf{ZK}_{boundedDL}(\mathsf{w}', \mathsf{w}'', 2^\lambda BT; a_3 r_w)$.

Fig. 6: Protocol $\mathsf{ZK}_{DDH-type}$

Lemma 8. *In the generic group model,* $\mathsf{ZK}_{DDH-type}$ *is an honest-verifier statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor such that for any polynomial* p*, for any prover* $\mathcal{P}^*$ *convincing the verifier with probability* p^{-1} *over statement* $(\mathsf{u}, \mathsf{v}, \mathsf{w}, T)$*, the extractor can extract* $x, y, a_1, a_2, a_3, c_1, c_2, c_3 \in \mathbb{Z}$ *such that* $|a_1|, |a_2|, |a_3| \leq 2^{2\lambda}BT$ $|c_1|, |c_2| \leq 2^{4\lambda}B^2$*,* $|c_3| \leq 2^{6\lambda}B^3$*, and* $\mathsf{u}^{c_1} = \mathsf{g}^{a_1 x}, \mathsf{v}^{c_2} = \mathsf{g}^{a_2 y}, \mathsf{w}^{c_3} = \mathsf{g}^{a_3 xy}$*.*

Proof. **Completeness** is obvious.

Special purpose knowledge soundness roughly follows from the weak knowledge soundness of $\mathsf{ZK}_{boundedDL}$ and the same knowledge extractor in PoKE. In fact, step 2 through step 4 are the PoDDH protocol provided by Boneh et al. in [BBF19], which roughly consists of two PoKE protocols. Using the knowledge extractor provided in [BBF19], one can extract x, y satisfying $\mathsf{u}' = \mathsf{g}^x$, $\mathsf{v}' = \mathsf{g}^y$ and $\mathsf{w}' = \mathsf{g}^{xy}$. Using the extractor of $\mathsf{ZK}_{boundedDL}$, one can extract a_u, c_u, a'_u, c'_u from the first two $\mathsf{ZK}_{boundedDL}$ protocols, satisfying $|c_u|, |c'_u| \leq 2^{\lambda}, |a_u| \leq 2^{3\lambda}B^2, |a'_u| \leq 2^{\lambda}BT$ and $\mathsf{u}^{a_u} = \mathsf{u}''^{c_u}, \mathsf{u}'^{a'_u} = \mathsf{u}''^{c'_u}$. Therefore, we have $\mathsf{u}^{a_u c'_u} = \mathsf{g}^{a'_u c_u x}$. Denoting by $c_1 = a_u c'_u$, $a_1 = a'_u c_u$, then $|c_1| \leq 2^{4\lambda}B^2, |a_1| \leq 2^{2\lambda}BT$ and $\mathsf{u}^{c_1} = \mathsf{g}^{a_1 x}$. Using the same strategy, one can extract c_2, c_3, a_2, a_3, thus meeting our goal.

The simulator Sim of **honest-verifier statistically zero-knowledge** property can be constructed as follows: Sim samples $r_1, r_2 \leftarrow [2^{2\lambda}B]$ and sets $\mathsf{u}' = \mathsf{g}^{r_1}, \mathsf{v}' = \mathsf{g}^{r_2}, \mathsf{w}' = \mathsf{g}^{r_1 r_2}$. In step 3, Sim finds the quotient $q_1, q_2 \in \mathbb{Z}$ and residue t_1, t_2 such that $r_1 = q_1 l_1 + t_1$ and $r_2 = q_2 l_2 + t_2$, and then applies the same action as an honest prover. Then, Sim samples $r_u, r_v, r_w \leftarrow [2^{\lambda}B]$, sets $\mathsf{u}'' = \mathsf{u}^{r_1 r_u}, \mathsf{v}'' = \mathsf{v}^{r_2 r_v}, \mathsf{w}'' = \mathsf{w}^{r_1 r_2 r_w}$, and simulates the $\mathsf{ZK}_{boundedDL}$ protocols to conclude the simulation.

Owning to the honest-verifier statistically zero-knowledge of $\mathsf{ZK}_{boundedDL}$, we only need to show that for any fixed statement $(\mathsf{u}, \mathsf{v}, \mathsf{w}, T)$ and challenges l_1, l_2, the distributions of $(\mathsf{u}', \mathsf{v}', \mathsf{w}', \mathsf{Q}_1, \mathsf{Q}'_1, \mathsf{Q}_2, t_1, t_2, \mathsf{u}'', \mathsf{v}'', \mathsf{w}'')$ generated by the simulator and prover are exponentially close. We denote these two distributions by $\mathcal{D}_{sim}$ and $\mathcal{D}_P$. Then, for any fixed statement $(\mathsf{u}, \mathsf{v}, \mathsf{w}, T)$ and challenges l_1, l_2, we have the following:

$$
\begin{aligned}
\mathcal{D}_{sim} =& \{(\mathsf{g}^{r_1}, \mathsf{g}^{r_2}, \mathsf{g}^{r_1 r_2}, \mathsf{g}^{\lfloor r_1/l_1 \rfloor}, (\mathsf{g}^{r_2})^{\lfloor r_1/l_1 \rfloor}, \mathsf{g}^{\lfloor r_2/l_2 \rfloor}, r_1 \mod l_1, \\
& r_2 \mod l_2, \mathsf{u}^{r_1 r_u}, \mathsf{v}^{r_2 r_v}, \mathsf{w}^{r_1 r_2 r_w}) | r_1, r_2 \leftarrow [2^{2\lambda}B], r_u, r_v, r_w \leftarrow [2^{\lambda}B]\} \\
\mathcal{D}_P =& \{(\mathsf{g}^{r_1 x}, \mathsf{g}^{r_2 y}, \mathsf{g}^{r_1 r_2 xy}, \mathsf{g}^{\lfloor r_1 x/l_1 \rfloor}, (\mathsf{g}^{r_2 y})^{\lfloor r_1 x/l_1 \rfloor}, \mathsf{g}^{\lfloor r_2 y/l_2 \rfloor}, r_1 x \mod l_1, \\
& r_2 y \mod l_2, \mathsf{u}^{r_1 r_u}, \mathsf{v}^{r_2 r_v}, \mathsf{w}^{r_1 r_2 r_w}) | r_1, r_2 \leftarrow [2^{2\lambda}B], r_u, r_v, r_w \leftarrow [2^{\lambda}B]\}
\end{aligned}
$$

For any fixed $\mathsf{u}, \mathsf{v}, \mathsf{w}, l_1, l_2$, denote $f(r_1, r_2, r_u, r_v, r_w) = (\mathsf{g}^{r_1}, \mathsf{g}^{r_2}, \mathsf{g}^{r_1 r_2}, \mathsf{g}^{\lfloor r_1/l_1 \rfloor}, (\mathsf{g}^{r_2})^{\lfloor r_1/l_1 \rfloor}, \mathsf{g}^{\lfloor r_2/l_2 \rfloor}, r_1 \bmod l_1, r_2 \bmod l_2, \mathsf{u}^{r_1 r_u}, \mathsf{v}^{r_2 r_v}, \mathsf{w}^{r_1 r_2 r_w})$. As a key observation, $f(r_1, r_2, r_u, r_v, r_w) = f(r_1 \bmod l_1|\mathbb{G}|, r_2 \bmod l_2|\mathbb{G}|, r_u \bmod |\mathbb{G}|, r_v \bmod |\mathbb{G}|, r_w \bmod |\mathbb{G}|)$. We then have $\mathcal{D}_{sim} = \{f(r_1 \bmod l_1|\mathbb{G}|, r_2 \bmod l_2|\mathbb{G}|, r_u \bmod |\mathbb{G}|, r_v \bmod |\mathbb{G}|, r_w \bmod |\mathbb{G}|) | r_1, r_2 \leftarrow [2^{2\lambda}B], r_u, r_v, r_w \leftarrow [2^{\lambda}B]\}$. Set x' (resp. y') the element in $\mathbb{Z}_{|\mathbb{G}|}$ satisfying $xx' \equiv 1 \mod \mathbb{G}$ (resp. $yy' \equiv 1 \mod \mathbb{G}$) (where x'

and y' exist owing to $\gcd(xy, |\mathbb{G}|) = 1$). Then, $\mathcal{D}_P = \{f(r_1 x \bmod l_1|\mathbb{G}|, r_2 y \bmod l_1|\mathbb{G}|, r_u x' \bmod |\mathbb{G}|, r_v y' \bmod |\mathbb{G}|, r_w x'y' \bmod |\mathbb{G}|)|r_1, r_2 \leftarrow [2^{2\lambda}B], r_u, r_v, r_w \leftarrow [2^{\lambda}B]\}$. From Lemma.1 and the fact that $\gcd(x, l_1|\mathbb{G}|) = 1$, $\gcd(y, l_2|\mathbb{G}|) = 1$, we have $\mathcal{D}_{sim}$ and $\mathcal{D}_P$ being exponentially close to the distribution $\{f(r_1, r_2, r_u, r_v, r_w)|r_1, \leftarrow \mathbb{Z}_{l_1|\mathbb{G}|}, r_2 \leftarrow \mathbb{Z}_{l_2|\mathbb{G}|}, r_u, r_v, r_w \leftarrow \mathbb{Z}_{|\mathbb{G}|}\}$, which concludes the proof.

Non-interactive zero-knowledge protocols. Note that all of the above protocols are constant-round public-coin protocols. One can use the Fiat-Shamir heuristic to obtain the non-interactive version of these zero-knowledge protocols. These non-interactive protocols satisfy zero-knowledge property and the same special purpose knowledge soundness property in the random oracle model. We use $\mathsf{NI\text{-}ZK}_{boundedDL}$, $\mathsf{NI\text{-}ZK}_{DL}$, $\mathsf{NI\text{-}ZK}_{2DL}$, $\mathsf{NI\text{-}ZK}_{prime}$, and $\mathsf{NI\text{-}ZK}_{DDH-type}$ to represent these protocols.

4.3 Construction of Zero-Knowledge Set with Verifiable Set Operations

In this subsection, we present a ZKS supporting verifiable set operations. Our scheme builds on Prabhakaran and Xue's ZKS scheme [PX09, XLL07, XLL08]. We denote by $\mathcal{H}_{prime}$ a hash function that upon inputting a string, outputs a large prime. The construction of Prabhakaran and Xue's ZKS scheme (with proper modification) is shown below.

Theorem 2. *The protocol constructed in Fig.7 is a ZKS scheme in the generic group model and random oracle model.*

The proof follows from [PX09, XLL07, XLL08]. We present a detailed proof in Appendix.B

Remark 1. One can use the batch technique put forward in [BBF19] to batch the (non-)membership proofs. For example, to prove that $x'_1, \cdots, x'_t \in S$, the prover hashes them into primes $p'_1, \cdots p'_t$ by $\mathcal{H}_{prime}$ and then generates the proof $\pi \leftarrow \mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^{\Pi_{i\in[t]}p'_i}, \mathsf{C}; r\Pi_{i\in[m]}p_i/\Pi_{i\in[t]}p'_i)$. To prove that $x'_1, \cdots, x'_t \notin S$, the prover hashes them into primes $p'_1, \cdots p'_t$ by $\mathcal{H}_{prime}$ and finds $a, b \in \mathbb{Z}$ such that $a\Pi_{i\in[t]}p'_i + br\Pi_{i\in[m]}p_i = 1$, and then outputs $\pi \leftarrow \mathsf{NI\text{-}ZK}_{2DL}(\mathsf{g}^{\Pi_{i\in[t]}p'_i}, \mathsf{C}, \mathsf{g}; a, b)$.

Set Operations. In this paper, we denote a set operation $\mathcal{Q}$ by a "circuit" of intersection "$\cap$", union "$\cup$", and set-difference "$\setminus$". As described in the technique overview, we first demonstrate how to prove a single set operation (a "$\cap$", "$\cup$" or "$\setminus$") over committed sets.

Algorithm for Intersection. Here we present a non-interactive protocol to prove that a commitment C_I commits to the intersection of two sets committed in C_{S_1} and C_{S_2}. Our protocol can roughly ensure the following:

- If one can generate a membership proof showing that x belongs to the set committed in C_I, the extractor can generate membership proofs showing that x belongs to the set committed in C_{S_1} and C_{S_2}.

Standard zero-knowledge set scheme

$\mathsf{Setup}(1^\lambda)$: On input the security parameter 1^λ, Setup generates the description of an unknown-order group $\mathbb{G} \leftarrow GGen(\lambda)$ and a random group element $g \leftarrow \mathbb{G}$. Suppose B is the upper bound of $\mathbb{G}$ (i.e., $B \geq |\mathbb{G}|$). Sample the description of a hash function $\mathcal{H}_{prime}$ that on input a string, outputs a random prime larger than $B_{prime} > 2^{6\lambda}B^3$. Output CRS $\delta = (\mathbb{G}, g, B, \mathcal{H}_{prime})$.

$\mathsf{Commit}(\delta, \mathsf{S})$: On input the set $S = \{x_1, \cdots, x_m\}$, Commit hashes them into large primes, i.e., for $i \in [m]$, $p_i \leftarrow \mathcal{H}_{prime}(x_i)$. Then Commit samples $r \leftarrow [2^\lambda B]$, and outputs the commitment $\mathsf{C} = \mathsf{g}^{r\Pi_{i\in[m]}p_i}$ and the open information $\tau = (r, p_1, \cdots, p_m, S)$.

$\mathsf{Prove}(\delta, (C, \tau), x, S(x))$: Parse the input τ as $(r, p_1, \cdots, p_m, S)$.
a) If $S(x) = 1$, which means that $x \in S$, $p = \mathcal{H}_{prime}(x) \in \{p_1, \cdots, p_m\}$, Prove outputs the proof $\pi \leftarrow \mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^p, \mathsf{C}; r\Pi_{i\in[m]}p_i/p)$.
b) If $S(x) = \perp$, which means that $x \notin S$, $p = \mathcal{H}_{prime}(x) \notin \{p_1, \cdots, p_m\}$ and $\gcd(p, r\Pi_{i\in[m]}p_i) = 1$, Prove finds a, b such that $ap + br\Pi_{i\in[m]}p_i = 1$ and outputs $\pi \leftarrow \mathsf{NI\text{-}ZK}_{2DL}(\mathsf{g}^p, \mathsf{C}, \mathsf{g}; a, b)$.

$\mathsf{Verify}(\delta, \mathsf{C}, x, S(x), \pi)$: If $S(x) = 1$, check whether π is a valid $\mathsf{NI\text{-}ZK}_{DL}$ proof for statement $(\mathsf{g}^p, \mathsf{C}) \in \mathcal{R}_{DL}$. If $S(x) = 0$, check whether π is a valid $\mathsf{NI\text{-}ZK}_{2DL}$ proof for statement $(\mathsf{g}^p, \mathsf{C}, g) \in \mathcal{R}_{2DL}$. Verify outputs 1 if the check passes and outputs 0 otherwise.

Fig. 7: Protocol ZKS

– If one can generate a non-membership proof showing that x does not belong to the set committed in C_I, the extractor can generate a non-membership proof showing that x does not belong to the set committed in C_{S_1} or C_{S_2}.

Follow the fact that, for any S_1 and S_2, to prove that $I = S_1 \cap S_2$, one only needs to show that I is a subset of S_1 and S_2 and $J_1 = S_1 \backslash I, J_2 = S_2 \backslash I$ are disjointed. We construct the zero-knowledge protocol $\mathsf{NI\text{-}ZK}_\cap$ shown in Fig. 8. For any set $S = \{x_1, \cdots, x_m\}$, we denote by $\mathcal{H}_{prime}(S) = \{\mathcal{H}_{prime}(x_1), \cdots, \mathcal{H}_{prime}(x_m)\}$.

Lemma 9. $\mathsf{NI\text{-}ZK}_\cap$ *is a statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor* $E = (E_1, E_2)$ *such that for any polynomial* p*, for any prover* $\mathcal{P}^*$ *convincing the verifier with probability* p^{-1} *over statement* $(\delta, \mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_I)$*,* $E_1^{\mathcal{P}^*}$ *can extract* w *in expected polynomial time such that the following holds:*

1. *On input* w*,* $\mathsf{g}_a \in \mathbb{G}$ *and prime* $p \in \mathbb{Z}$ *satisfying* $p \geq 2^{6\lambda}B^3$ *and* $\mathsf{C}_I = \mathsf{g}_a^p$*,* $E_2(w, (\mathsf{g}_a, p))$ *can output* g_b *and* g_c *such that* $\mathsf{C}_{S_1} = \mathsf{g}_b^p$ *and* $\mathsf{C}_{S_2} = \mathsf{g}_c^p$*.*

Protocol $\mathsf{NI\text{-}ZK}_{\cap}$

Input: $\delta = (\mathbb{G}, g, B, \mathcal{H}_{prime})$, $\mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_I$; Witness: $S_1, S_2, I, \tau_{S_1}, \tau_{S_2}, \tau_I$ satisfying that $(\mathsf{C}_{S_1}, \tau_{S_1}) \in \mathsf{Commit}(\delta, S_1)$; $(\mathsf{C}_{S_2}, \tau_{S_2}) \in \mathsf{Commit}(\delta, S_2)$; $(\mathsf{C}_I, \tau_I) \in \mathsf{Commit}(\delta, I)$ and $S_1 \cap S_2 = I$.

Prover:

1. Generate the commitments C_{J_1} and C_{J_2} to the sets $J_1 = S_1 \backslash I$ and $J_2 = S_2 \backslash I$.
2. Run $\pi_1 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_1}, 2^\lambda B)$ and $\pi_2 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_I, \mathsf{C}_{J_2}, \mathsf{C}_{S_2}, 2^\lambda B)$ with a proper witness contained in the opening information. (*This step shows that $I \subset S_1$ and $I \subset S_2$.*)
3. Run $\pi_3 \leftarrow \mathsf{NI\text{-}ZK}_{prime}(\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, 2^\lambda B)$ with a proper witness contained in the opening information. (*This step shows that the sets $J_1 = S_1 \backslash I, J_2 = S_2 \backslash I$ are disjointed.*)
4. Output $\pi = (\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, \pi_1, \pi_2, \pi_3)$

Verifier: Output 1 iff. the proofs contained in π are valid.

Fig. 8: Protocol $\mathsf{NI\text{-}ZK}_{\cap}$

2. *On input w and $a, b, p \in \mathbb{Z}$ such that prime $p \geq 2^{6\lambda}B^3$ and $\mathsf{C}_I^a \cdot \mathsf{g}^{bp} = \mathsf{g}$, $E_2(w, (a, b, p))$ can output $a', b' \in \mathbb{Z}$ such that $\mathsf{C}_{S_1}^{a'} \cdot \mathsf{g}^{b'p} = \mathsf{g}$ or $\mathsf{C}_{S_2}^{a'} \cdot \mathsf{g}^{b'p} = \mathsf{g}$.*

Proof. Because $S_1 \cap S_2 = I, J_1 = S_1 \backslash I$, we have $I \cup J_1 = S_1$ and $I \cap J_1 = \emptyset$. This means that $H_{prime}(I) \cup H_{prime}(J_1) = H_{prime}(S_1)$ and $H_{prime}(I) \cap H_{prime}(J_1) = \emptyset$ (otherwise, the collision-resistant property of H_{prime} is broken). Therefore, it is true that $(\Pi_{p_i \in H_{prime}(I)} p_i) \cdot (\Pi_{p_i \in H_{prime}(J_1)} p_i) = \Pi_{p_i \in H_{prime}(S_1)} p_i$, which means that $(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_1}, 2^\lambda B) \in \mathcal{R}_{DDH-type}$. For the same reason, $(\mathsf{C}_I, \mathsf{C}_{J_2}, \mathsf{C}_{S_2}, 2^\lambda B) \in \mathcal{R}_{DDH-type}$ is also a DDH-type tuple. Similarly, because $J_1 \cap, J_2 = \emptyset$, we have $(\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, [2^\lambda B]) \in \mathcal{R}_{prime}$, which concludes the **completeness**.

The simulator of the **statistically zero-knowledge** property only needs to generate $\mathsf{C}_{J_1} = \mathsf{g}^{r_1}$ and $\mathsf{C}_{J_2} = \mathsf{g}^{r_2}$ by sampling $r_1, r_2 \leftarrow [2^\lambda B]$, and generate π_1, π_2, π_3 using the simulator of $\mathsf{NI\text{-}ZK}_{DDH-type}$ and $\mathsf{NI\text{-}ZK}_{prime}$. Then the statistically zero-knowledge property follows from the fact that the distributions of $\mathsf{C}_{J_1}, \mathsf{C}_{J_2}$ generated by the simulator are statistically indistinguishable from those of the honest prover and the statistically zero-knowledge property of $\mathsf{NI\text{-}ZK}_{DDH-type}$ and $\mathsf{NI\text{-}ZK}_{prime}$.

The proof of the special purpose **knowledge soundness** is as follows.

From the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{DDH-like}$, E_1 can extract $w_1 = (x, y, a_1, a_2, a_3, c_1, c_2, c_3)$ such that $|a_1|, |a_2|, |a_3| \leq 2^{3\lambda}B^2$, $|c_1|, |c_2| \leq 2^{4\lambda}B^2$, $|c_3| \leq 2^{6\lambda}B^3$, and $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x}, \mathsf{C}_{J_1}^{c_2} = \mathsf{g}^{a_2 y}, \mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy}$; and $w_2 = (x', y', a_1', a_2', a_3', c_1', c_2', c_3')$ such that $|a_1'|, |a_2'|, |a_3'| \leq 2^{3\lambda}B^2$, $|c_1'|, |c_2'| \leq 2^{4\lambda}B^2$, $|c_3'| \leq 2^{6\lambda}B^3$,and $\mathsf{C}_I^{c_1'} = \mathsf{g}^{a_1' x'}, \mathsf{C}_{J_2}^{c_2'} = \mathsf{g}^{a_2' y'}, \mathsf{C}_{S_2}^{c_3'} = \mathsf{g}^{a_3' x' y'}$. From the special pur-

pose knowledge soundness of $\mathsf{NI\text{-}ZK}_{prime}$, E_1 can extract $w_3 = (t_1, t_2, c)$ such that $c \leq 2^{3\lambda}B^2$ and $\mathsf{C}_{J_1}^{t_1}\mathsf{C}_{J_2}^{t_2} = \mathsf{g}^c$. Here, E_1 outputs $w = (\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, w_1, w_2, w_3)$.

For the construction of E_2 suppose that (g_a, p) satisfies $p \geq 2^{6\lambda}B^3$ and $\mathsf{C}_I = \mathsf{g}_a^p$. Because $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x}$, we have $\mathsf{g}_a^{c_1 p} = \mathsf{g}^{a_1 x}$. We claim that $p | a_1 x$ otherwise from Lemma.3 one can easily find a p-root of g and break the strong RSA assumption. Because $p > a_1$, we know that $p|x$. Denote $x_p = x/p \in \mathbb{Z}$. Then $\mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy} = (\mathsf{g}^{a_3 x_p y})^p$. As $\gcd(p, c_3) = 1$, E_2 can easily compute g_b such that $\mathsf{C}_{S_1} = \mathsf{g}_b^p$. In addition, E_2 can compute g_c from Lemma.3 such that $\mathsf{C}_{S_2} = \mathsf{g}_c^p$ in the same manner.

Suppose that $a, b, p \in \mathbb{Z}$ satisfies p as a prime larger than $2^{6\lambda}B^3$ and $\mathsf{C}_I^a \cdot \mathsf{g}^{bp} = \mathsf{g}$. Consider the case that $\gcd(p, y) = 1$. Because $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x}$, we have $\gcd(p, x) = 1$, otherwise one can find a p-root of g (from Lemma.3, one can find h such that $\mathsf{h}^p = \mathsf{C}_I$, and therefore we have $(\mathsf{h}^a \cdot \mathsf{g}^b)^p = \mathsf{g}$) and break the strong RSA assumption. Then, because $\gcd(p, y) = 1$, we have $\gcd(p, a_3 xy) = 1$, and E_2 can easily find integers α, β such that $\alpha p + \beta a_3 xy = 1$. Therefore, from $\mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy}$, we have $\mathsf{C}_{S_1}^{\beta c_3}\mathsf{g}^{\alpha p} = \mathsf{g}$. Setting $a' = \beta c_3$ and $b' = \alpha$, then $\mathsf{C}_{S_1}^{a'} \cdot \mathsf{g}^{b' p} = \mathsf{g}$. Similarly, in the case that $\gcd(p, y') = 1$, E_2 can compute $a', b' \in \mathbb{Z}$ such that $\mathsf{C}_{S_2}^{a'} \cdot \mathsf{g}^{b' p} = \mathsf{g}$. Now, we only need to consider the case in which $gcd(p, y) \neq 1, gcd(p, y') \neq 1$. Because p is prime, we have $p|y$ and $p|y'$. Then, from $\mathsf{C}_{J_1}^{c_2} = \mathsf{g}^{a_2 y}, \mathsf{C}_{J_2}^{c_2'} = \mathsf{g}^{a_2' y'}$ and Lemma.3, one can easily compute h_1 and h_2 satisfying $\mathsf{C}_{J_1} = \mathsf{h}_1^p$ and $\mathsf{C}_{J_2} = \mathsf{h}_2^p$. From $\mathsf{C}_{J_1}^{t_1}\mathsf{C}_{J_2}^{t_2} = \mathsf{g}^c$, one can find a p-root of g directly and break the strong RSA assumption, which concludes the proof.

Algorithm for Union. Similarly, herein we present a non-interactive protocol to prove that a commitment C_U commits to the union of two sets committed in C_{S_1} and C_{S_2}. Roughly, our protocol can ensure the following:

- If one can generate a membership proof showing that x belongs to the set committed in C_U, the extractor can generate a membership proof showing that x belongs to the set committed in C_{S_1} or C_{S_2}.
- If one can generate a non-membership proof showing that x does not belong to the set committed in C_I, the extractor can generate non-membership proofs showing that x does not belong to the set committed in C_{S_1} and C_{S_2}.

We construct the zero-knowledge protocol $\mathsf{NI\text{-}ZK}_\cup$ in Fig. 9.

Lemma 10. $\mathsf{NI\text{-}ZK}_\cup$ *is a statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists an extractor* $E = (E_1, E_2)$ *such that for any polynomial* p*, for any prover* P^* *convincing the verifier with probability* p^{-1} *over the statement* $(\delta, \mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_I)$*,* $E_1^{P^*}$ *can extract* w *within an expected time such that the following hold:*

1. *On input* w *and* $\mathsf{g}_a \in \mathbb{Z}$ *and prime* p *satisfying* $p \geq 2^{6\lambda}B^3$ *and* $\mathsf{C}_U = \mathsf{g}_a^p$*,* $E_2(w, (\mathsf{g}^a, p))$ *can output* g_b *such that* $\mathsf{C}_{S_1} = \mathsf{g}_b^p$ *or* $\mathsf{C}_{S_2} = \mathsf{g}_b^p$*.*
2. *On input* w *and* $a, b, p \in \mathbb{Z}$ *such that prime* $p \geq 2^{6\lambda}B^3$ *and* $\mathsf{C}_U^a \cdot \mathsf{g}^{bp} = \mathsf{g}$*,* $E_2(w, (a, b, p))$ *can output* $a', b', a'', b'' \in \mathbb{Z}$ *such that* $\mathsf{C}_{S_1}^{a'} \cdot \mathsf{g}^{b' p} = \mathsf{g}$ *and* $\mathsf{C}_{S_2}^{a''} \cdot \mathsf{g}^{b'' p} = \mathsf{g}$*.*

Protocol $\mathsf{NI\text{-}ZK}_{\cup}$

Input: $\delta = (\mathbb{G}, g, B, \mathcal{H}_{prime})$, $\mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_U$; Witness: $S_1, S_2, U, \tau_{S_1}, \tau_{S_2}, \tau_U$ satisfying that $(\mathsf{C}_{S_1}, \tau_{S_1}) \in \mathsf{Commit}(\delta, S_1); (\mathsf{C}_{S_2}, \tau_{S_2}) \in \mathsf{Commit}(\delta, S_2); (\mathsf{C}_U, \tau_U) \in \mathsf{Commit}(\delta, U)$ and $S_1 \cup S_2 = U$.

Prover

1. Generates the commitment $\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{J_2}$ to the sets $I = S_1 \cap S_2, J_1 = U \backslash S_1, J_2 = U \backslash S_2$.
2. Run $\pi_1 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_2}, 2^{\lambda}B)$ with a proper witness contained in the opening information.
3. Run $\pi_2 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_{S_1}, \mathsf{C}_{J_1}, \mathsf{C}_U, 2^{\lambda}B)$ and $\pi_3 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_{S_2}, \mathsf{C}_{J_2}, \mathsf{C}_U, 2^{\lambda}B)$ with a proper witness contained in the opening information.
4. Output $\pi = (\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{J_2}, \pi_1, \pi_2, \pi_3)$

Verifier: Output 1 iff. the proofs contained in π are valid.

Fig. 9: Protocol $\mathsf{NI\text{-}ZK}_{\cup}$

The proof of this lemma is similar to Lemma.9. We defer it to Appendix.C.

Algorithm for Set-Difference. We present a non-interactive protocol to prove that a commitment C_D commits to the difference of two sets committed in C_{S_1} and C_{S_2}. Roughly, our protocol can ensure the following:

- If one can generate a membership proof showing that x belongs to the set committed in C_D, the extractor can generate a membership proof showing that x belongs to the set committed in C_{S_1} and a non-membership proof showing that x doesn't belong to the set committed in C_{S_2}.
- If one can generate a non-membership proof showing that x does not belong to the set committed in C_D, the extractor can generate a non-membership proof showing that x does not belong to the set committed in C_{S_1} and a membership proof showing that x belongs to the set committed in C_{S_2}.

We construct the zero-knowledge protocol $\mathsf{NI\text{-}ZK}_{\backslash}$ in Fig. 10.

Lemma 11. $\mathsf{NI\text{-}ZK}_{\backslash}$ *is a statistically zero-knowledge protocol achieving a type of special purpose knowledge soundness defined as follows: There exists a extractor* $E = (E_1, E_2)$ *such that for any polynomial* p*, for any prover* P^* *convincing the verifier with probability* p^{-1} *over statement* $(\delta, \mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_D)$*,* $E_1^{P^*}$ *can extract* w *such that the following holds:*

1. *On input* w, $\mathsf{g}_a \in \mathbb{G}$ *and prime* $p \in \mathbb{Z}$ *such that* $p \geq 2^{6\lambda}B^3$ *and* $\mathsf{C}_D = \mathsf{g}_a^p$*,* $E_2(w, (\mathsf{g}_a, p))$ *can output* g_b *and* a, b *such that* $\mathsf{C}_{S_1} = \mathsf{g}_b^p$ *and* $\mathsf{C}_{S_2}^a \mathsf{g}^{pb} = \mathsf{g}$*.*
2. *On input* w *and* $a, b, p \in \mathbb{Z}$ *such that prime* $p \geq 2^{6\lambda}B^3$ *and* $\mathsf{C}_D^a \cdot \mathsf{g}^{bp} = \mathsf{g}$*,* $E_2(w, (a, b, p))$ *can output* $a', b' \in \mathbb{Z}$ *or* $\mathsf{g}_a \in \mathbb{G}$ *such that* $\mathsf{C}_{S_1}^{a'} \cdot \mathsf{g}^{b'p} = g$ *or* $\mathsf{C}_{S_2} = \mathsf{g}_a^p$*.*

Protocol NI-ZK$_\setminus$

Input: $\delta = (\mathbb{G}, g, B, \mathcal{H}_{prime})$, $\mathsf{C}_{S_1}, \mathsf{C}_{S_2}, \mathsf{C}_D$; Witness: $S_1, S_2, S_D, \tau_{S_1}, \tau_{S_2}, \tau_D$ satisfying that $(\mathsf{C}_{S_1}, \tau_{S_1}) \in \mathsf{Commit}(\delta, S_1); (\mathsf{C}_{S_2}, \tau_{S_2}) \in \mathsf{Commit}(\delta, S_2); (\mathsf{C}_D, \tau_D) \in \mathsf{Commit}(\delta, D)$ and $S_1 \setminus S_2 = D$.

Prover:

1. Generate the commitments $\mathsf{C}_I, \mathsf{C}_J$ to the sets $I = S_1 \cap S_2, J = S_2 \setminus I$.
2. Run $\pi_1 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_I, \mathsf{C}_D, \mathsf{C}_{S_1}, 2^\lambda B)$ and $\pi_2 \leftarrow \mathsf{NI\text{-}ZK}_{DDH-type}(\mathsf{C}_I, \mathsf{C}_J, \mathsf{C}_{S_2}, 2^\lambda B)$ with a proper witness contained in the opening information.
3. Run $\pi_3 \leftarrow \mathsf{NI\text{-}ZK}_{prime}(\mathsf{C}_D, \mathsf{C}_{S_2}, 2^\lambda B)$ with a proper witness contained in the opening information.
4. Output $\pi = (\mathsf{C}_I, \mathsf{C}_J, \pi_1, \pi_2, \pi_3)$

Verifier: Output 1 iff. the proofs contained in π are valid.

Fig. 10: Protocol NI-ZK$_\setminus$

The proof of this lemma is similar to Lemma.9, and we defer it to Appendix.C.

Algorithm for Set Operations. Using above algorithms, we now construct the ProveS and VerifierS algorithms to conclude the construction of ZKS supporting verifiable set operations. Note that a set operation $\mathcal{Q}$ over the inputs can be regard as a "circuit" consisting of wires and nodes. Each input wire is related to an input set, each node is related to a set operation and its output wire is related to a set that results in this set operation being applied on the sets related to its input wires. Without loss of generality, let the output wire of the last node also be the output wire of this "circuit". The protocol is shown as follows:

Theorem 3. *The ZKS in Fig.7 together with the algorithms in Fig.11 is a ZKS supporting verifiable set operations in the generic group model and random oracle model.*

Proof. **Completeness** is oblivious.

To prove the **functional binding** property, we will show the existence of a extractor E satisfying that, for any PPT adversary $\mathcal{A}$ generating a series of valid query-proof tuples with a notice property, E can extract a series of sets satisfying all queries or breaking the strong RSA assumption. Here, E is constructed as follows.

1. First, E invokes $\mathcal{A}$ to obtain m commitments, $\mathsf{C}_1, \cdots, \mathsf{C}_m$. Then, E initializes $2m + 1$ sets, $S_0, S_1, S'_1, \cdots, S_m, S'_m$. Here, S_0 is used to record the keys/elements x appearing in the answer to the queries (including the keys appearing in membership and non-membership proofs, and set operation queries). In addition, for any $i \in [m]$, S_i is the sets of keys that, believed

Algorithm for Verifiable Set Operations

$\mathsf{ProveS}(\delta, \widetilde{com}, \widetilde{\tau}, \mathcal{Q}, S_{output})$: On input the CRS δ, the list of commitments and the associated opening information $\widetilde{com} = (\mathsf{C}_1, \cdots, \mathsf{C}_m)$, $\widetilde{\tau} = (\tau_1, \cdots, \tau_m)$ where $(\mathsf{C}_i, \tau_i) \in \mathsf{Com}(\delta, S_i)$, a set operation $\mathcal{Q}$ and the target output set S_{output}. ProveS runs as follows.

1. Recover the sets $\{S_i\}_{i\in[m]}$ from the opening information. Regard $\mathcal{Q}$ as a "circuit" and let l be the number of nodes. Run $\mathcal{Q}(S_1, \cdots, S_m)$ to obtain the set $S'_1, \cdots, S'_l$ corresponding to the output wires of l nodes in $\mathcal{Q}$ (note that $S'_l = S_{output}$). Generate the commitments $\mathsf{C}'_1, \cdots, \mathsf{C}'_l$ to the sets $S'_1, \cdots, S'_l$.
2. For each node i, suppose S_{a_i}, S_{b_i} are the sets associated to its input wires and S_{c_i} is the set associated to its output wire.
   (a) If the node is related to "interaction", ProveS runs $\pi_i \leftarrow \mathsf{NI\text{-}ZK}_{\cap}(\delta, \mathsf{C}_{a_i}, \mathsf{C}_{b_i}, \mathsf{C}_{c_i})$.
   (b) If the node is related to "union", ProveS runs $\pi_i \leftarrow \mathsf{NI\text{-}ZK}_{\cup}(\delta, \mathsf{C}_{a_i}, \mathsf{C}_{b_i}, \mathsf{C}_{c_i})$.
   (c) If the node is related to "set-difference", ProveS runs $\pi_i \leftarrow \mathsf{NI\text{-}ZK}_{\setminus}(\delta, \mathsf{C}_{a_i}, \mathsf{C}_{b_i}, \mathsf{C}_{c_i})$.
3. Output $\pi = (\mathsf{C}'_1, \cdots, \mathsf{C}'_l, \pi_1, \cdots, \pi_l, \tau'_l)$, where τ'_l is the opening information of C'_l.

$\mathsf{VerifyS}(\delta, \widetilde{com}, \mathcal{Q}, S_{output}, \pi)$: Parse π as $(\mathsf{C}'_1, \cdots, \mathsf{C}'_l, \pi_1, \cdots, \pi_l, \tau'_l)$. Regard $\mathcal{Q}$ as a "circuit" in the same way as the prover. For each node i, suppose $\mathsf{C}_{a_i}, \mathsf{C}_{b_i}$ are the commitments committing to the sets associated to the input wires and C_{c_i} is the commitment committing to the set associated to the output wire. If this node is related to "interaction" (resp. "union", "set-difference"), check whether π_i is a valid $\mathsf{NI\text{-}ZK}_{\cap}$ (resp. $\mathsf{NI\text{-}ZK}_{\cup}, \mathsf{NI\text{-}ZK}_{\setminus}$) proof over the statement $\delta, \mathsf{C}_{a_i}, \mathsf{C}_{b_i}, \mathsf{C}_{c_i}$. Use τ'_l to check whether C'_l is a commitment to the set S_{output}. Output 1 iff. all checks pass.

Fig. 11: Protocol for Verifiable Set Operations

by E, are contained in the set committed in C_i; S'_i is the sets of keys that, believed by E, not contained in the set committed in C_i. Here, E invokes $\mathcal{A}$ to obtain the query-proof tuples and the keys x appearing in the answer to the queries and adds the keys to S_0.

2. For each membership proof proving that x belongs to the set committed in C_i, E adds x to S_i, and extracts and records $(x, \mathsf{g}_x, p, \mathsf{C}_i)$ such that $\mathsf{g}_x^p = \mathsf{C}_i$, where $p = \mathcal{H}_{prime}(x)$ (we skip the extraction because such proceedings have appeared several times herein). For each non-membership proof proving that x does not belong to the set committed in C_i, E adds x to S'_i, and extracts and records $(x, a, b, p, \mathsf{C}_i)$ such that $\mathsf{C}_i^a \mathsf{g}^{pb} = \mathsf{g}$. For each set operation query-

proof tuple, E uses the extractors E_1 of $\mathsf{NI\text{-}ZK}_{\cap}$, $\mathsf{NI\text{-}ZK}_{\cup}$ and $\mathsf{NI\text{-}ZK}_{\setminus}$ to extract the corresponding w.

3. For each element $x \in S_0$, E applies the following. For each set operation query-proof tuple, if x belongs to the output set, one can then find a tuple $(x, \mathsf{g}_x, p, \mathsf{C}'_l)$ such that $\mathsf{g}_x^p = \mathsf{C}'_l$ and $\mathcal{H}_{prime}(x) = p$. Then, from the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{\cap}$, $\mathsf{NI\text{-}ZK}_{\cup}$, and $\mathsf{NI\text{-}ZK}_{\setminus}$, one can extract one or two tuples regarding its input wires. Here, E recursively applies the same action to these wires, and can finally find a set of tuples for the input wires. If x does not belong to the output set, E can apply the similar actions. As a result, for each tuple of the form $(x, \mathsf{g}', p, \mathsf{C}_i)$, E adds x to S_i, and for each tuple of the form $(x, a, b, p, \mathsf{C}_i)$, E adds x to S'_i.
4. If a contradiction (i.e., for some key x and $i \in [m]$, $x \in S_i \wedge x \in S'_i$,) does not exist, then E outputs $S_1, \cdots, S_m$. If there exists a contradiction, it means that there exists $(x, \mathsf{g}_x, p, \mathsf{C}_i)$ and $(x, a, b, p, \mathsf{C}_i)$ such that $\mathsf{g}_x^p = \mathsf{C}_i$ and $\mathsf{C}_i^a \mathsf{g}^{pb} = \mathsf{g}$, which breaks the strong RSA Assumption.

Now we only need to show that the sets $S_1, \cdots, S_m$ outputted by E satisfy all queries. From step 2, we can see that these sets already satisfy the (non-)membership queries. As for set operation queries, we need to show that: For each set operation query-proof $(\mathsf{C}_{t_1}, \cdots, \mathsf{C}_{t_k}, \mathcal{Q}, S_{output}, \pi)$ and each $x \in S_0$, $\mathcal{Q}(S^x_{t_1}, \cdots, S^x_{t_k}) = S^x_{output}$, where $S^x_i = \{x\} \cap S_i$ and $S^x_{output} = \{x\} \cap S_{output}$.

We run $\mathcal{Q}(S^x_{t_1}, \cdots, S^x_{t_k})$ to obtain the set $S^x_{m+1}, \cdots, S^x_{m+l}$ corresponding to the output wires of l nodes in $\mathcal{Q}$ (we further require that the input sets of nodes j belong to $\{S_i\}_{i<j}$). Suppose $\{\mathsf{C}_i\}_{i\in[m+1,m+l]}$ are the commitments contained in π, committing the sets associated to internal wires. Remind that E will extract several tuples for each wire in $\mathcal{Q}$ in step 3. Here, we recursively prove the following statements are true for each $i \in [m+l]$:

For each extracted tuple of the form $(x, \mathsf{g}', p, \mathsf{C}_i)$, $x \in S^x_i$; And for each extracted tuple of the form $(x, a, b, p, \mathsf{C}_i)$, $x \notin S^x_i$.

From the description of $S_1, \cdots, S_m$, above statements are true for $i \in [m]$. Suppose the statements are true for $i \leq [m+k-1]$. Then for node k with inputs S_i, S_j, if the node is related to "interactive" and $(x, \mathsf{g}_a, p, \mathsf{C}_k)$ (resp. $(x, a, b, p, \mathsf{C}_k)$) is extracted, then from the knowledge soundness of $\mathsf{NI\text{-}ZK}_{\cap}$, $(x, \mathsf{g}_b, p, \mathsf{C}_i)$ and $(x, \mathsf{g}_c, p, \mathsf{C}_j)$ (resp. $(x, a', b', \mathsf{C}_i)$ or $(x, a', b', \mathsf{C}_j)$) are extracted by E, which means that $x \in S_i, x \in S_j$ (resp. $x \notin S_i$ or $x \notin S_j$), and therefore $x \in S_k = S_i \cap S_j$ (resp. $x \notin S_k = S_i \cap S_j$). The case that node k is related to "union" or "set-difference" can be proved similarly. Therefore above statement is also true for $i = k$. Recursively, the above statements are true for each $i \in [m+l]$. Note that S^x_{m+l} is the output set. Remind that if $x \in S^x_{output}$ (resp. $x \notin S^x_{output}$), E will extract $(x, \mathsf{g}', p, \mathsf{C}_{l+m})$ (resp. $(x, a, b, p, \mathsf{C}_{l+m})$), which means that $x \in S^x_{m+l}$ (resp. $x \notin S^x_{m+l}$). Therefore we have $S^x_{m+l} = S^x_{output}$, which concludes the proof of functional binding.

For the **zero-knowledge** property, because the distribution of the ZKS commitment is statistically indistinguishable from $\{\mathsf{g}^r | r \leftarrow [2^\lambda B]\}$, the simulator can sample element from $\{\mathsf{g}^r | r \leftarrow [2^\lambda B]\}$ as the commitments and then use the simulators of $\mathsf{NI\text{-}ZK}_{\cap}$, $\mathsf{NI\text{-}ZK}_{\cup}$ and $\mathsf{NI\text{-}ZK}_{\setminus}$ to conclude the simulations.

Remark 2. One can use the randomness r'_l applied in the commitment C'_l to replace the opening information τ'_l, which is also sufficient to check whether C'_l is a commitment to S_{output}. Then the proof size is only linear to the size of set operation $\mathcal{Q}$, the length of elements in S.

5 Function Queriable Zero-Knowledge Elementary Databases

In this section, we introduce the notion of function queriable zero-knowledge elementary databases, and then show how to construct it from standard ZE-EDBs and ZKS supporting verifiable set operation.

5.1 Definition

Informally, a function queriable zero-knowledge elementary database allows one to commit a key-value database $D = \{(x, v)\}$ and later convincingly answer the queries in the form of "*Send all records* (x, v) *in* D *satisfying* $f(x, v) = 1$ *for any Boolean circuit* f". Here we write the output database as $D(f)$ and regard the membership queries (supported by the standard ZK-EDB) as a type of special functional query.

Definition 3 (Function Queriable Zero-Knowledge Elementary Database). *A function queriable zero-knowledge elementary database consists of four algorithms* (Setup, Com, ProveF, VerifyF)*,*

- $\delta \leftarrow \mathsf{Setup}(1^\lambda)$*: On input a security parameter* 1^λ*,* Setup *outputs a random string (or a structured reference string)* δ *as the CRS.*
- $(com, \tau) \leftarrow \mathsf{Com}(\delta, D)$*: On input a CRS* δ *and an elementary database* D*,* Com *outputs a commitment of database com and opening information* τ*.*
- $\pi \leftarrow \mathsf{Prove}(\delta, com, \tau, f, D_{output})$*: On input the CRS* δ*, the database commitment and the associated opening information* (com, τ)*, a Boolean circuit* f*, and the target database* D_{output}*,* Prove *outputs a proof* π *for* $D_{output} = D(f)$*.*
- $0/1 \leftarrow \mathsf{Verify}(\delta, com, f, D_{output}, \pi)$*: On input the CRS* δ*, the commitment com, the boolean circuit* f*, the target output* D_{output} *and the proof* π*,* Verify *accepts or rejects.*

It satisfies the following three properties:

- **Completeness:** *For any elementary database* D *and any* x*,*

$$\Pr\left[\mathsf{Verify}(\delta, com, f, D(f), \pi) = 1 \;\middle|\; \begin{array}{c}\delta \leftarrow \mathsf{Setup}(1^\lambda); (com, \tau) \leftarrow \mathsf{Com}(\delta, D); \\ \pi \leftarrow \mathsf{Prove}(\delta, com, \tau, f, D(f))\end{array}\right] = 1$$

- **Functional binding:** *For any PPT adversary* $\mathcal{A}$*, the probability that* $\mathcal{A}$ *wins in the following game is negligible:*
  1. *The challenger generates a CRS* δ *by running* $\mathsf{Setup}(1^\lambda)$ *and gives* δ *to adversary* $\mathcal{A}$*.*

2. *The adversary $\mathcal{A}$ outputs a commitment com and a series of function query-proof tuples $\{(f_i, D_i, \pi_i)\}_{i\in[n]}$.*
3. *The adversary $\mathcal{A}$ wins the game if the following hold: a) For each $i \in [n]$, $\mathsf{Verify}(com, f_i, D_i, \pi_i) = 1$ and b) there does not exist a database D satisfying $D(f_i) = D_i$ for each $i \in [n]$.*

- **Zero-Knowledge:** *There exists a simulator Sim such that for any PPT adversary $\mathcal{A}$, the absolute value of the difference*

$$\Pr\left[\mathcal{A}^{\mathsf{Prove}(\delta,com,\tau,\cdot,D(\cdot))}(\delta, state_{\mathcal{A}}, com) = 1 \;\middle|\; \begin{array}{c} \delta \leftarrow \mathsf{Setup}(1^\lambda), \\ (D, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ (com, \tau) \leftarrow \mathsf{Com}(\delta, D) \end{array}\right] -$$

$$\Pr\left[\mathcal{A}^{Sim(state_S,\cdot,D(\cdot))}(\delta, state_{\mathcal{A}}, com) = 1 \;\middle|\; \begin{array}{c} (\delta, state_\delta) \leftarrow Sim(1^\lambda), \\ (D, state_{\mathcal{A}}) \leftarrow \mathcal{A}(\delta), \\ (com, state_S) \leftarrow Sim(\delta, state_\delta) \end{array}\right]$$

is negligible in λ.

5.2 Construction

In this section, we present a construction of the function queriable ZK-EDB from a standard ZK-EDB $(\mathsf{Setup}_D, \mathsf{Com}_D, \mathsf{Prove}_D, \mathsf{Verify}_D)$ and a ZKS supporting verifiable set operations $(\mathsf{Setup}_S, \mathsf{Com}_S, \mathsf{Prove}_S, \mathsf{Verify}_S, \mathsf{ProveS}_S, \mathsf{VerifyS}_S)$. Before we present the construction, we first introduce a deterministic algorithm that transform a Boolean circuit f into a circuit of set operations $\mathcal{Q}$.

Algorithm $Q \leftarrow \mathsf{Tran}(f)$: On input the boolean circuit $f : \{0,1\}^n \rightarrow \{0,1\}$, $\mathsf{Tran}(f)$ outputs $\mathcal{Q}$, a circuit of set operations having an input of $2n$ sets $(S_1^0, S_1^1, \cdots, S_n^0, S_n^1)$, and outputs a set S'. Here, $\mathcal{Q}$ is constructed as follows:

Tran first associates the i-th input wires of f with two sets (S_i^0, S_i^1). Supposing that f contains l gates $(n_1, \cdots, n_l)$, without loss of generality, we require the input wires of n_i to be either the input wires of f or the output wires of gates $(n_1, \cdots, n_{i-1})$, and the output of gate n_l is also the output of f. Then for i from 1 to l, we have the following:

1. If gate n_i is "AND" gate, and the sets associated with the two input wires are $(S_{input1}^0, S_{input1}^1), (S_{input2}^0, S_{input2}^1)$, then denote the sets associated with the output wire as $(S_{input1}^1 \cap S_{input2}^1, S_{input1}^0 \cup S_{input2}^0)$.
2. If gate n_i is "OR" gate, and the sets associated with the two input wires are $(S_{input1}^0, S_{input1}^1), (S_{input2}^0, S_{input2}^1)$, then denote the sets associated with the output wire as $(S_{input1}^1 \cup S_{input2}^1, S_{input1}^0 \cap S_{input2}^0)$.
3. If gate n_i is "NOT" gate, and the sets associated with the two input wires are $(S_{input}^0, S_{input}^1)$, then denote the sets associated with the output wire as $(S_{input}^1, S_{input}^0)$.

Supposing that (S^0, S^1) are the sets associated with the output wire of gate n_l, $\mathcal{Q}$ outputs S^1.

Denote by Sup the algorithm that on input a key-value database $D = \{(x_1, v_1), \cdots, (x_m, v_m)\}$, outputs the set of keys belonging to D, i.e., $Sup(D) = \{x_1, \cdots, x_m\}$. We then have the following:

Lemma 12. *Tran is a deterministic algorithm satisfying that for any Boolean circuit f and any key-value databases D,*

$$\mathcal{Q}(S_1^0, S_1^1, \cdots, S_n^0, S_n^1) = Sup(D(f))$$

where $S_i^b = \{x \in Sup(D) |$ the i-th bit of "$x||v$" is $b\}$, and $D(f) = \{(x, v) \in D | f(x, v) = 1\}$.

The correctness of Lemma. 12 can be directly checked.
The construction of the function queriable ZK-EDB is shown in Fig.12.

Function queriable ZK-EDB

$\mathsf{Setup}(1^\lambda)$: On input the security parameter 1^λ, Setup generates $\delta_D \leftarrow \mathsf{Setup}_D(1^\lambda)$ and $\delta_S \leftarrow \mathsf{Setup}_S(1^\lambda)$. Output CRS $\delta = (\delta_D, \delta_S)$.

$\mathsf{Commit}(\delta, D)$: On input the database D and $\delta = (\delta_D, \delta_S)$, Commit runs $(C_D, \tau_D) \leftarrow \mathsf{Com}_D(\delta_D, D)$. For each $b \in \{0,1\}$, $i \in [2l]$, denote by $S_i^b = \{x \in Sup(D) |$ the i-th bit of $x||v$ is $b\}$. Commit $\mathcal{S}_i^b$ using the ZKS scheme, i.e., $(\mathsf{C}_i^b, \tau_i^b) \leftarrow \mathsf{Com}_S(\delta_S, \mathcal{S}_i^b)$. Output the commitment $C = (C_D, \{\mathsf{C}_i^b\}_{b\in\{0,1\}, i\in[2l]})$ and the open information $\tau = (\tau_D, \{\tau_i^b\}_{b\in\{0,1\}, i\in[2l]})$.

$\mathsf{Prove}(\delta, C, \tau, f, D(f))$: Prover transforms Boolean circuit f into a circuit of set operations $\mathcal{Q}$ using thealgorithm Tran. Run $\pi_\mathcal{Q} \leftarrow \mathsf{ProveS}_S(\delta_S, \{\mathsf{C}_i^b\}, \{\tau_i^b\}, \mathcal{Q}, Sup(D(f)))$. For each $x \in Sup(\mathcal{D}(f))$, run $\pi_x \leftarrow \mathsf{Prove}_D(\delta_D, C_D, \tau_D, x, (D(x))$ and for $b \in \{0,1\}, i \in [2l]$, run $\pi_{x,i}^b \leftarrow \mathsf{Prove}_S(\delta_S, \mathsf{C}_i^b, \tau_i^b, x, \mathcal{S}_i^b(x))$. Output $\pi = (\pi_Q, \{\pi_x, \pi_{x,i}^b\}_{x\in Sup(D(f)), b\in\{0,1\}, i\in[2l]})$.

$\mathsf{Verify}(\delta, C, f, D(f), \pi)$: Parse C as $(C_D, \{\mathsf{C}_i^b\}_{b\in\{0,1\}, i\in[2l]})$, parse the input π as $(\pi_\mathcal{Q}, \{\pi_x, \pi_{x,i}^b\}_{x\in Sup(D(f)), b\in\{0,1\}, i\in[2l]})$ and run $Q = \mathsf{Tran}(f)$. Output 1 iff. the following checks pass:

1. $\mathsf{VerifyS}_S(\delta_S, \{\mathsf{C}_i^b\}, \mathcal{Q}, Sup(D(f)), \pi_Q) = 1$
2. For each $x \in Sup(\mathcal{D}(f))$, $\mathsf{Verify}_D(\delta_D, C_D, x, D(x), \pi_x) = 1$.
3. For each $x \in Sup(\mathcal{D}(f))$ and $b \in \{0,1\}, i \in [2l]$, $\mathsf{Verify}_S(\delta_S, \mathsf{C}_i^b, x, \mathcal{S}_i^b(x), \pi_{x,i}^b) = 1$

Fig. 12: Functional queriable ZK-EDB

Theorem 4. *The scheme shown in Fig.12 is a function queriable zero-knowledge elementary database scheme.*

Proof. The **completeness** follows from Lemma.12 directly.

To prove the **functional binding** property, we will show that, supposing there exists a PPT adversary $\mathcal{A}$ that on input a random CRS δ, outputs a commitment C and a series of valid query-proof tuples $\{f_i, D_i, \pi_{f_i}\}_{i\in[t]}$ such that $\mathsf{Verify}(\delta, C, f_i, D_i, \pi_{f_i}) = 1$ with noticeable property. Then $D = \cup_{i\in[t]} D_i$ is a database satisfying $D(f_i) = D_i$.

First, we claim that D is indeed a database (for each x belonging to the support of D, there is at most one value v satisfying $(x, v) \in D$), otherwise, one can break the soundness of the ZK-EDB scheme $(\mathsf{Setup}_D, \mathsf{Com}_D, \mathsf{Prove}_D, \mathsf{Verify}_D)$.

Second, we claim that for each $i \in [t]$, $D(f_i) = D_i$. Denote by $S_i^b = \{x \in Sup(D) |$ the i-th bit of $x||v$ is $b\}$. From the functional binding of ZKS with verifiable set operations, we know that there exists sets $S_i'^b$ satisfying the first and third checks of the verifier in each proof, which means the following:

1. $\mathcal{Q}_i(S_1'^0, S_1'^1, \cdots, S_n'^0, S_n'^1) = Sup(D_i)$ where $\mathcal{Q}_i = \mathsf{Tran}(f_i)$.
2. For each $i \in [2l]$ and $x \in Sup(D)$, $x \in S_i'^{b_{x,i}}$ and $x \notin S_i'^{1-b_{x,i}}$ where $b_{x,i}$ is the i-th bit of $x||D(x)$.

From the second property above, we have $S_i^b = S_i'^b \cap Sup(\mathcal{D})$. Now, from the first property, we have that $\mathcal{Q}_i(S_1^0, S_1^1, \cdots, S_{2l}^0, S_{2l}^1) = \mathcal{Q}_i(S_1'^0 \cap Sup(\mathcal{D}), S_1'^1 \cap Sup(D), \cdots, S_{2l}'^0 \cap Sup(D), S_{2l}'^1 \cap Sup(D)) = D_i \cap Sup(D) = D_i$. From Lemma.12, we have $D(f_i) = D_i$, which concludes the proof.

The **zero-knowledge** property directly follows the zero-knowledge property of ZK-EDB and ZKS supporting verifiable set operations.

6 Conclusion

In this paper, we introduced the notion of function queriable zero-knowledge elementary databases and showed that it can be constructed from standard ZK-EDBs and a new variation of ZKS that support verifiable set operations. We presented a concrete construction of ZKS supporting verifiable set operations based on unknown-order group. Such a scheme is secure in the random oracle model and generic group model.

A construction of standard constant-size ZK-EDB from groups of unknown-order is presented in Appendix.D. Instantiated this ZK-EDB scheme and our ZKS scheme supporting verifiable set operations, the resulting function queriable ZK-EDB scheme is secure in the Random Oracle model and Generic Group model and its proof size is only linear to l (length of each record in database) and the size of circuit f.

Bibliography

[AR20] Shashank Agrawal and Srinivasan Raghuraman. Kvac: Key-value commitments for blockchains and beyond. In *Advances in Cryptology - ASIACRYPT'20*, LNCS 12493, pages 839–869. Springer, 2020.

[BBF19] Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to iops and stateless blockchains. In *Advances in Cryptology - CRYPTO'19*, LNCS 11692, pages 561–586. Springer, 2019.

[BdM93] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In *Advances in Cryptology - EUROCRYPT'93*, LNCS 765, pages 274–285. Springer, 1993.

[BH01] Johannes Buchmann and Safuat Hamdy. A survey on iq cryptography. In *In Proceedings of Public Key Cryptography and Computational Number Theory*, pages 1–15, 2001.

[BP97] Niko Barić and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology — EUROCRYPT '97*, pages 480–494. Springer, 1997.

[CDV06] Dario Catalano, Yevgeniy Dodis, and Ivan Visconti. Mercurial commitments: Minimal assumptions and efficient constructions. In *Theory of Cryptography - TCC'06*, LNCS 3876, pages 120–144. Springer, 2006.

[CF13] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *Public-Key Cryptography - PKC'13*, LNCS 7778, pages 55–72. Springer, 2013.

[CFM08] Dario Catalano, Dario Fiore, and Mariagrazia Messina. Zero-knowledge sets with short proofs. In *Advances in Cryptology - EUROCRYPT'08*, LNCS 4965, pages 433–450. Springer, 2008.

[CHKO08] Philippe Camacho, Alejandro Hevia, Marcos A. Kiwi, and Roberto Opazo. Strong accumulators from collision-resistant hashing. In *Information Security, 11th International Conference - ISC'08*, LNCS 5222, pages 471–486. Springer, 2008.

[CHL+05] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In *Advances in Cryptology - EUROCRYPT'05*, LNCS 3494, pages 422–439. Springer, 2005.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Advances in Cryptology - CRYPTO'02*, LNCS 2442, pages 61–76. Springer, 2002.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In *Advances in Cryptology - CRYPTO'97*, LNCS 1294, pages 410–424. Springer, 1997.

[CV12] Melissa Chase and Ivan Visconti. Secure database commitments and universal arguments of quasi knowledge. In *Advances in Cryptology - CRYPTO'12*, LNCS 7417, pages 236–254. Springer, 2012.

[DF02] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *Advances in Cryptology - ASIACRYPT'02*, LNCS 2501, pages 125–142. Springer, 2002.

[DHS15] David Derler, Christian Hanser, and Daniel Slamanig. Revisiting cryptographic accumulators, additional properties and relations to other primitives. In *Topics in Cryptology - CT-RSA'15*, LNCS 9048, pages 127–144. Springer, 2015.

[DK02] Ivan Damgård and Maciej Koprowski. Generic lower bounds for root extraction and signature schemes in general groups. In *Advances in Cryptology - EUROCRYPT'02*, LNCS 2332, pages 256–271. Springer, 2002.

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology - CRYPTO '97*, LNCS 1294, pages 16–30. Springer, 1997.

[GM06] Rosario Gennaro and Silvio Micali. Independent zero-knowledge sets. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP'06*, LNCS 4052, pages 34–45. Springer, 2006.

[GOP+16] Esha Ghosh, Olga Ohrimenko, Dimitrios Papadopoulos, Roberto Tamassia, and Nikos Triandopoulos. Zero-knowledge accumulators and set algebra. In *Advances in Cryptology – ASIACRYPT'16*, pages 67–100. Springer, 2016.

[GOT15] Esha Ghosh, Olga Ohrimenko, and Roberto Tamassia. Zero-knowledge authenticated order queries and order statistics on a list. In *Applied Cryptography and Network Security - ACNS'15*, LNCS 9092, pages 149–171. Springer, 2015.

[Lis05] Moses D. Liskov. Updatable zero-knowledge databases. In *Advances in Cryptology - ASIACRYPT'05, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, LNCS 3788, pages 174–198. Springer, 2005.

[LNTW19] Benoît Libert, Khoa Nguyen, Benjamin Hong Meng Tan, and Huaxiong Wang. Zero-knowledge elementary databases with more expressive queries. In *Public-Key Cryptography - PKC'19*, LNCS 11442, pages 255–285. Springer, 2019.

[LSY+21] Yannan Li, Willy Susilo, Guomin Yang, Tran Viet Xuan Phuong, Yong Yu, and Dongxi Liu. Concise mercurial subvector commitments: Definitions and constructions. In *Information Security and Privacy*, pages 353–371. Springer, 2021.

[LY10] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *Theory of Cryptography - TCC'10*, LNCS 5978, pages 499–517. Springer, 2010.

[MRK03] Silvio Micali, Michael O. Rabin, and Joe Kilian. Zero-knowledge sets. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science - FOCS'03*, pages 80–91. IEEE Computer Society, 2003.

[Ngu05] Lan Nguyen. Accumulators from bilinear pairings and applications. In *Topics in Cryptology - CT-RSA'05*, LNCS 3376, pages 275–292. Springer, 2005.

[NZ15] Moni Naor and Asaf Ziv. Primary-secondary-resolver membership proof systems. In *Theory of Cryptography - TCC'15*, LNCS 9015, pages 199–228. Springer, 2015.

[PTT11] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Optimal verification of operations on dynamic sets. In *Advances in Cryptology - CRYPTO'11*, LNCS 6841, pages 91–110. Springer, 2011.

[PX09] Manoj Prabhakaran and Rui Xue. Statistically hiding sets. In *Topics in Cryptology - CT-RSA'09*, LNCS 5473, pages 100–116. Springer, 2009.

[Str19] Michael Straka. Class groups for cryptographic accumulators, 2019. https://www.michaelstraka.com/posts/classgroups/.

[Tam03] Roberto Tamassia. Authenticated data structures. In *Algorithms - ESA'03*, LNCS 2832, pages 2–5. Springer, 2003.

[XLL07] Rui Xue, Ninghui Li, and Jiangtao Li. A new construction of zero-knowledge sets secure in random oracle model. In *The First International Symposium on Data, Privacy, and E-Commerce - ISDPE'07)*, pages 332–337, 2007.

[XLL08] Rui Xue, Ninghui Li, and Jiangtao Li. Algebraic construction for zero-knowledge sets. *J. Comput. Sci. Technol.*, 23(2):166–175, 2008.

[Zhu09] Huafei Zhu. Mercurial commitments from general rsa moduli and their applications to zero-knowledge databases/sets. *Computer Science and Engineering, International Workshop on*, 2:289–292, 10 2009.

[ZKP17] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. An expressive (zero-knowledge) set accumulator. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 158–173, 2017.

Appendix

A The proofs of Lemma.1, Lemma.2 and Lemma.3

Lemma. 1 *For any positive integers a, A and B satisfying that $B > A$, we have that:*

$$\mathsf{Dist}(\{x \leftarrow [a]\}, \{x \mod a | x \leftarrow [A, B]\}) \leq \frac{a}{B - A}$$

where Dist *means the distance between distributions.*

Proof. For any integer $t \in a$, we have that $\Pr[x = t | x \leftarrow [a]] = 1/a$ and $\Pr[x \bmod a = t | x \leftarrow [A, B]] = \frac{\lfloor (B-A)/a \rfloor}{B-A}$ or $\frac{\lfloor (B-A)/a \rfloor + 1}{B-A} \in [\frac{1}{a} - \frac{1}{B-A}, \frac{1}{a} + \frac{1}{B-A}]$. Therefore we have that $\mathsf{Dist}(\{x \leftarrow [a]\}, \{x \bmod a | x \leftarrow [A, B]\}) = \Sigma_{t \in [a]} |\Pr[x = t | x \leftarrow [a]] - \Pr[x \bmod a = t | x \leftarrow [A, B]]| \leq \frac{a}{B-A}$. □

Lemma. 2 *For any integers s_1,s_2 and positive integers a, A, B satisfying that $B > A$, $\gcd(s_1, s_2) = 1$, we have that:*

$$\mathsf{Dist}(\{x \leftarrow [a]\}, \{xs_1 + ys_2 \mod a | x, y \leftarrow [A, B]\}) \leq \frac{3a}{B - A}$$

where Dist *means the distance between distributions.*

Proof. For any $t \in \mathbb{Z}_b$, denote by S_t the set of pairs $(x, y) \in \mathbb{Z}_a^2$ satisfying that $xs_1 + ys_2 \equiv t \mod a$. Due to that $\gcd(s_1, s_2) = 1$, there exists integers b_1, b_2 such that $b_1 s_1 + b_2 s_2 = 1$. Therefore, for each $i \in \mathbb{Z}_a$, $(tb_1 + is_2 \mod a)s_1 + (tb_2 - is_1 \mod a)s_2 \equiv t \mod a$. And for any $i, j \in \mathbb{Z}_a$, if $(tb_1 + is_2 \mod a, tb_2 - is_1 \mod a) = (tb_1 + js_2 \mod a, tb_2 - js_1 \mod a)$, then $(i - j)s_2 \equiv (i - j)s_1 \equiv 0 \mod a \Rightarrow (i-j)(b_1 s_1 + b_2 s_2) \equiv 0 \mod a \Rightarrow i - j \equiv 0 \mod a \Rightarrow i = j$. Therefore we have that, for any $t \in \mathbb{Z}_b$, $\{(tb_1 + is_2 \mod a, tb_2 - is_1 \mod a)\}_{i \in \mathbb{Z}_a} \subset S_t$ and $|S_t| \geq a$. Due to that $|\mathbb{Z}_a^2| = a^2$, we have that $t^2 \leq \Sigma_{t \in \mathbb{Z}_a} |S_t| \leq |\mathbb{Z}_a^2| = a^2$. Therefore, for any $t \in \mathbb{Z}_a$, $|S_t| = a$.

Now, for any $t \in a$, we have that $\Pr[xs_1 + ys_2 \bmod a = t | x, y \leftarrow [A, B]] = \Sigma_{(x,y) \in S_t} \Pr[xs_1 + ys_2 \bmod a = t | x, y \leftarrow [A, B]] \in [a(\frac{1}{a} - \frac{1}{B-A})^2, a(\frac{1}{a} + \frac{1}{B-A})^2]$. Therefore we have that $\mathsf{Dist}(\{x \leftarrow [a]\}, \{xs_1 + ys_2 \bmod a | x, y \leftarrow [A, B]\}) = \Sigma_{t \in [a]} |\Pr[x = t | x \leftarrow [a]] - \Pr[xs_1 + ys_2 \bmod a = t | x, y \leftarrow [A, B]]| \leq \frac{2a}{B-A} + (\frac{a}{B-A})^2$. If $B - A > a$, then we have that $\mathsf{Dist}(\{x \leftarrow [a]\}, \{xs_1 + ys_2 \bmod a | x, y \leftarrow [A, B]\}) \leq \frac{2a}{B-A} + (\frac{a}{B-A})^2 \leq \frac{3a}{B-A}$; if $B - A < a$, then we have that $\mathsf{Dist}(\{x \leftarrow [a]\}, \{xs_1 + ys_2 \bmod a | x, y \leftarrow [A, B]\}) \leq 1 \leq \frac{3a}{B-A}$. The lemma is conclude. □

Lemma. 3 *For any multiplicative group $\mathbb{G}$ and group elements $\mathsf{g}, \mathsf{h} \in \mathbb{G}$, if there exists coprime integers a, p satisfying that $\mathsf{g}^a = \mathsf{h}^p$, then one can easily compute h' satisfying that $\mathsf{g} = \mathsf{h}'^p$ from a, p, g and h.*

Proof. Due to that $\gcd(a, p) = 1$, one could easily compute integers t_1, t_2 such that $t_1 a + t_2 p = 1$. Then we have that $\mathsf{g}^a = \mathsf{h}^p \Rightarrow \mathsf{g}^{at_1} = \mathsf{h}^{pt_1} \Rightarrow \mathsf{g}^{at_1 + pt_2} = \mathsf{h}^{pt_1} \mathsf{g}^{pt_2} \Rightarrow \mathsf{g} = (\mathsf{h}^{t_1} \mathsf{g}^{t_2})^p$. Then, set $\mathsf{h}' = \mathsf{h}^{t_1} \mathsf{g}^{t_2}$ and we have that $\mathsf{g} = \mathsf{h}'^p$. □

B The proof of Theorem.2

Proof. **Completeness** is obvious.

Soundness follows that: Suppose there exists an adversary $\mathcal{A}$ breaking the soundness property, which means that, upon inputting a random CRS δ, $\mathcal{A}$ can output $(\mathsf{C}, x, v, v', \pi, \pi')$ such that with a noticeable probability, $v \neq v'$ (without loss of generality, we assume that $v = 1$ and $v' = \perp$) and $\mathsf{Verify}(\delta, \mathsf{C}, x, v, \pi) = \mathsf{Verify}(\delta, \mathsf{C}, x, v', \pi') = 1$.

Let $p = \mathcal{H}_{prime}(x)$. Then, from the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{DL}$, one can extract (a, t) such that $|t| \leq 2^\lambda$ and $(\mathsf{g}^p)^a = \mathsf{C}^t$. Because $\gcd(p, t) = 1$, one can find integers α, β such that $\alpha p + \beta t = 1$, and then $(\mathsf{g}^{a\beta}\mathsf{C}^\alpha)^p = \mathsf{C}$. From the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{2DL}$, one can extract (a', b', t') such that $|t'| \leq 2^\lambda$ and $(\mathsf{g}^p)^{a'}\mathsf{C}^{b'} = \mathsf{g}^{t'}$. We therefore have $(\mathsf{g}^{a'}(\mathsf{g}^{a\beta}\mathsf{C}^\alpha)^{b'})^p = \mathsf{g}^{t'}$. Setting $\mathsf{c} = \mathsf{g}^{a'}(\mathsf{g}^{a\beta}\mathsf{C}^\alpha)^{b'}$, then $\mathsf{c}^p = \mathsf{g}^{t'}$. Again, because $\gcd(p, t') = 1$, one can find α', β' such that $\alpha' p + \beta' t' = 1$, and therefore $(\mathsf{c}^{\beta'}\mathsf{g}^{\alpha'})^p = \mathsf{g}$ which breaks the strong RSA assumption and concludes the soundness property. (Note that the strong RSA assumption holds in the generic group model [DF02].)

Zero-knowledge property follows the zero-knowledge property of $\mathsf{NI\text{-}ZK}_{DL}$ and $\mathsf{NI\text{-}ZK}_{2DL}$. Upon inputting a random CRS δ, the simulator Sim samples $r \leftarrow [2^\lambda B]$ and outputs the commitment $\mathsf{C} = \mathsf{g}^r$. To simulate the proof, Sim directly runs the simulator of $\mathsf{NI\text{-}ZK}_{DL}$ and $\mathsf{NI\text{-}ZK}_{2DL}$. From Lemma.1, the distribution of simulated commitment $\{\mathsf{g}^r | r \leftarrow [2^\lambda B]\}$ is statistically indistinguishable from the uniform distribution over group $\langle \mathsf{g} \rangle$. In addition, for any set $S = \{x_1, \cdots, x_m\}$ and $(p_1, \cdots, p_m) = (\mathcal{H}_{prime}(x_1), \cdots, \mathcal{H}_{prime}(x_m))$, the distribution of honest commitment $\{\mathsf{g}^{r\Pi_{i\in[m]}p_i} | r \leftarrow [2^\lambda B]\}$ is also statistically indistinguishable from the uniform distribution over group $\langle g \rangle$ (note that $\gcd(\Pi_{i\in[m]}p_i, Ord(\mathsf{g})) = 1$, and therefore $\mathsf{g}^{\Pi_{i\in[m]}p_i}$ is still a generator of the group $\langle \mathsf{g} \rangle$). As a result, the distributions of the simulated and honest commitments are statistically indistinguishable. Together with the statistical zero-knowledge property of $\mathsf{NI\text{-}ZK}_{DL}$ and $\mathsf{NI\text{-}ZK}_{2DL}$, no adversary can tell the simulator apart from the honest committer and prover, which concludes the proof.

C The proof of Lemma.10 and Lemma.11

In this section, we prove the security of protocols $\mathsf{NI\text{-}ZK}_\cup$ and $\mathsf{NI\text{-}ZK}_\setminus$.

Proof of Lemma.10:

The proof of Lemma.10 follows the same strategy of Lemma.9.

Because $S_1 \cup S_2 = U, S_1 \cap S_2 = I, J_1 = U \setminus S_1$ and $J_2 = U \setminus S_2$, we have $I \cup J_1 = S_2, I \cap J_1 = \emptyset$. This means that $H_{prime}(I) \cup H_{prime}(J_1) = H_{prime}(S_2)$ and $H_{prime}(I) \cap H_{prime}(J_1) = \emptyset$ (otherwise, the collision-resistant property of H_{prime} is broken). Therefore, it is true that $(\Pi_{p_i \in H_{prime}(I)} p_i) \cdot (\Pi_{p_i \in H_{prime}(J_1)} p_i) = \Pi_{p_i \in H_{prime}(S_2)} p_i$, which means that $(\mathsf{C}_I, \mathsf{C}_{J_1}, \mathsf{C}_{S_2}, 2^\lambda B) \in \mathcal{R}_{DDH-type}$. For the same reason, because $S_1 \cup J_1 = U, S_1 \cap J_1 = \emptyset$ and $S_2 \cup J_2 = U, S_2 \cap J_2 = \emptyset$,

$(\mathsf{C}_{S_1}, \mathsf{C}_{J_1}, \mathsf{C}_U, 2^\lambda B)$ and $(\mathsf{C}_{S_2}, \mathsf{C}_{J_2}, \mathsf{C}_U, 2^\lambda B)$ are also DDH-type tuples, which concludes the **completeness**.

The simulator of the **statistically zero-knowledge** property only needs to generate $\mathsf{C}_{J_1} = \mathsf{g}^{r_1}$, $\mathsf{C}_{J_2} = \mathsf{g}^{r_2}$ and $\mathsf{C}_I = \mathsf{g}^{r_3}$ by sampling $r_1, r_2, r_3 \leftarrow [2^\lambda B]$, and generate π_1, π_2, π_3 using the simulator of $\mathsf{NI\text{-}ZK}_{DDH-type}$. Then the statistically zero-knowledge property follows from the fact that the distributions of $\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, \mathsf{C}_I$ generated by the simulator are statistically indistinguishable from those of the honest prover and the statistically zero-knowledge property of $\mathsf{NI\text{-}ZK}_{DDH-type}$.

The proof of the special purpose **knowledge soundness** is as follows.

From the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{DDH-like}$, E_1 can extract $w_1 = (x, y, a_1, a_2, a_3, c_1, c_2, c_3)$ such that $|a_1|, |a_2|, |a_3| \leq 2^{3\lambda}B^2$, $|c_1|, |c_2| \leq 2^{4\lambda}B^2$, $|c_3| \leq 2^{6\lambda}B^3$, and $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x}, \mathsf{C}_{J_1}^{c_2} = \mathsf{g}^{a_2 y}, \mathsf{C}_{S_2}^{c_3} = \mathsf{g}^{a_3 xy}$; $w_2 = (x', y', a_1', a_2', a_3', c_1', c_2', c_3')$ such that $|a_1'|, |a_2'|, |a_3'| \leq 2^{3\lambda}B^2$, $|c_1'|, |c_2'| \leq 2^{4\lambda}B^2$, $|c_3'| \leq 2^{6\lambda}B^3$,and $\mathsf{C}_{S_1}^{c_1'} = \mathsf{g}^{a_1' x'}, \mathsf{C}_{J_1}^{c_2'} = \mathsf{g}^{a_2' y'}, \mathsf{C}_U^{c_3'} = \mathsf{g}^{a_3' x' y'}$; and $w_3 = (x'', y'', a_1'', a_2'', a_3'', c_1'', c_2'', c_3'')$ such that $|a_1''|, |a_2''|, |a_3''| \leq 2^{3\lambda}B^2$, $|c_1''|, |c_2''| \leq 2^{4\lambda}B^2$, $|c_3''| \leq 2^{6\lambda}B^3$,and $\mathsf{C}_{S_2}^{c_1''} = \mathsf{g}^{a_1'' x''}, \mathsf{C}_{J_2}^{c_2''} = \mathsf{g}^{a_2'' y''}, \mathsf{C}_U^{c_3''} = \mathsf{g}^{a_3'' x'' y''}$. Here, E_1 outputs $w = (\mathsf{C}_{J_1}, \mathsf{C}_{J_2}, \mathsf{C}_U, w_1, w_2, w_3)$.

For the construction of E_2, suppose that the input (g_a, p) satisfies $p \geq 2^{6\lambda}B^3$ and $\mathsf{C}_U = \mathsf{g}_a^p$. Because $\mathsf{C}_U^{c_3'} = \mathsf{g}^{a_3' x' y'}$, we have $\mathsf{g}_a^{c_3' p} = \mathsf{g}^{a_3' x' y'}$. We claim that $p | a_3' x' y'$ otherwise from Lemma.3 one can easily find a p-root of g and break the strong RSA assumption. Because $p > a_1$, we know that $p | x'$ or $p | y'$. In the case of $p | x'$, denote $x_p = x'/p \in \mathbb{Z}$. Then $\mathsf{C}_{S_1}^{c_1'} = \mathsf{g}^{a_1' x'} = (\mathsf{g}^{a_1' x_p})^p$. As $\gcd(p, c_1') = 1$, E_2 can easily compute g_b such that $\mathsf{C}_{S_1} = \mathsf{g}_b^p$. In the case of $p | y'$, E_2 can similarly compute g_c such that $\mathsf{C}_{J_1} = \mathsf{g}_c^p$. Again, from $\mathsf{C}_{J_1}^{c_2} = \mathsf{g}^{a_2 y}$, we have $\mathsf{g}_c^{c_2 p} = \mathsf{g}^{a_2 y}$. We claim that $p | a_2 y$ otherwise from Lemma.3 one can easily find a p-root of g and break the strong RSA assumption. Because $p > a_1$, we know that $p | y$. Denoting $y_p = y/p$, we have $\mathsf{C}_{S_2}^{c_3} = \mathsf{g}^{a_3 xy} = (\mathsf{g}^{a_3 x y_p})^p$. As $\gcd(p, c_3) = 1$, E_2 can easily compute g_d such that $\mathsf{C}_{S_2} = \mathsf{g}_d^p$.

Suppose that the input $a, b, p \in \mathbb{Z}$ satisfies p as a prime larger than $2^{6\lambda}B^3$ and $\mathsf{C}_U^a \cdot \mathsf{g}^{bp} = \mathsf{g}$. Because $\mathsf{C}_U^{c_3'} = \mathsf{g}^{a_3' x' y'}$, we have $\gcd(p, x') = 1$, otherwise one can find a p-root of g (from Lemma.3, one can find h such that $\mathsf{h}^p = \mathsf{C}_U$, and therefore we have $(\mathsf{h}^a \cdot \mathsf{g}^b)^p = \mathsf{g}$) and break the strong RSA assumption. Then, from $\mathsf{C}_{S_1}^{c_1'} = \mathsf{g}^{a_1' x'}$ and $p > a_1'$, we have $\gcd(p, a_1' x') = 1$, and E_2 can easily find integers α, β such that $\alpha p + \beta a_1' x' = 1$, and then $\mathsf{C}_{S_1}^{c_1' \beta} \mathsf{g}^{\alpha p} = \mathsf{g}$. Setting $a' = \beta c_1'$ and $b' = \alpha$, then $\mathsf{C}_{S_1}^{a'} \cdot \mathsf{g}^{b' p} = \mathsf{g}$. In the same strategy, E_2 can compute a'', b'' such that $\mathsf{C}_{S_2}^{a''} \cdot \mathsf{g}^{b'' p} = \mathsf{g}$, which concludes the proof. □

Proof of Lemma.11:

The proof of Lemma.11 follows the same strategy of Lemma.9.

Because $S_1 \backslash S_2 = D, S_1 \cap S_2 = I$ and $J = S_2 \backslash S_1$, we have $I \cup D = S_1$, $I \cap D = \emptyset$. This means that $H_{prime}(I) \cup H_{prime}(D) = H_{prime}(S_1)$ and $H_{prime}(I) \cap H_{prime}(D) = \emptyset$ (otherwise, the collision-resistant property of H_{prime} is broken). Therefore, it is true that $(\Pi_{p_i \in H_{prime}(I)} p_i) \cdot (\Pi_{p_i \in H_{prime}(D)} p_i) =$

$\Pi_{p_i \in H_{prime}(S_1)} p_i$, which means that $(\mathsf{C}_I, \mathsf{C}_D, \mathsf{C}_{S_1}, 2^\lambda B) \in \mathcal{R}_{DDH-type}$. For the same reason, because $I \cup J = S_1, I \cap J = \emptyset$, $(\mathsf{C}_I, \mathsf{C}_J, \mathsf{C}_{S_1}, 2^\lambda B)$ is also a DDH-type tuple. Similarly, because $D \cap S_2 = \emptyset$, we have $(\mathsf{C}_D, \mathsf{C}_{S_2}, 2^\lambda B) \in \mathcal{R}_{prime}$, which concludes the **completeness**.

The simulator of the **statistically zero-knowledge** property only needs to generate $\mathsf{C}_I = \mathsf{g}^{r_1}$ and $\mathsf{C}_J = \mathsf{g}^{r_2}$ by sampling $r_1, r_2 \leftarrow [2^\lambda B]$, and generate π_1, π_2, π_3 using the simulator of $\mathsf{NI\text{-}ZK}_{DDH-type}$ and $\mathsf{NI\text{-}ZK}_{prime}$. Then the statistically zero-knowledge property follows from the fact that the distributions of $\mathsf{C}_I, \mathsf{C}_J$ generated by the simulator are statistically indistinguishable from those of the honest prover and the statistically zero-knowledge property of $\mathsf{NI\text{-}ZK}_{DDH-type}$ and $\mathsf{NI\text{-}ZK}_{prime}$.

The proof of the special purpose **knowledge soundness** is as follows.

From the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{DDH-like}$, E_1 can extract $w_1 = (x, y, a_1, a_2, a_3, c_1, c_2, c_3)$ such that $|a_1|, |a_2|, |a_3| \leq 2^{3\lambda} B^2$, $|c_1|, |c_2| \leq 2^{4\lambda} B^2$, $|c_3| \leq 2^{6\lambda} B^3$, and $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x}, \mathsf{C}_D^{c_2} = \mathsf{g}^{a_2 y}, \mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy}$; and $w_2 = (x', y', a_1', a_2', a_3', c_1', c_2', c_3')$ such that $|a_1'|, |a_2'|, |a_3'| \leq 2^{3\lambda} B^2$, $|c_1'|, |c_2'| \leq 2^{4\lambda} B^2$, $|c_3'| \leq 2^{6\lambda} B^3$,and $\mathsf{C}_I^{c_1'} = \mathsf{g}^{a_1' x'}, \mathsf{C}_J^{c_2'} = \mathsf{g}^{a_2' y'}, \mathsf{C}_{S_2}^{c_3'} = \mathsf{g}^{a_3' x' y'}$. From the special purpose knowledge soundness of $\mathsf{NI\text{-}ZK}_{prime}$, E_1 can extract $w_3 = (t_1, t_2, c)$ such that $c \leq 2^{3\lambda} B^2$ and $\mathsf{C}_D^{t_1} \mathsf{C}_{S_2}^{t_2} = \mathsf{g}^c$. Here, E_1 outputs $w = (\mathsf{C}_D, \mathsf{C}_{S_2}, w_1, w_2, w_3)$.

For the construction of E_2 suppose that the input (g_a, p) satisfies $p \geq 2^{6\lambda} B^3$ and $\mathsf{C}_D = \mathsf{g}_a^p$. Because $\mathsf{C}_D^{c_2} = \mathsf{g}^{a_2 y}$, we have $\mathsf{g}_a^{c_2 p} = \mathsf{g}^{a_2 y}$. We claim that $p | a_2 y$ otherwise from Lemma.3 one can easily find a p-root of g and break the strong RSA assumption. Because $p > a_2$, we know that $p | y$. Denote $y_p = y/p \in \mathbb{Z}$. Then $\mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy} = (\mathsf{g}^{a_3 x_p y})^p$. As $\gcd(p, c_3) = 1$, E_2 can easily compute g_b such that $\mathsf{C}_{S_1} = \mathsf{g}_b^p$. At the same time, because $\mathsf{C}_D^{t_1} \mathsf{C}_{S_2}^{t_2} = \mathsf{g}^c$, we have $\mathsf{g}_a^{t_1 p} \mathsf{C}_{S_2}^{t_2} = \mathsf{g}^c$. Because $\mathsf{C}_{S_2}^{c_3'} = \mathsf{g}^{a_3' x' y'}$, we have $\gcd(p, a_3' x' y') = 1$ (Otherwise, $p | a_3' x' y'$ and one have $(\mathsf{g}_a^{t_1} \mathsf{g}^{a_3' x' y' t_2 / p})^p = \mathsf{g}^c$. From Lemma.3, one can find a p-root of g directly and break the strong RSA assumption.). E_2 can efficiently find $\alpha, \beta \in \mathbb{Z}$ such that $\alpha p + \beta a_3' x' y' = 1$, and then $\mathsf{C}_{S_2}^{\beta c_3'} \mathsf{g}^{\alpha p} = \mathsf{g}$. Setting $a' = \beta c_3', b' = \alpha$, then $\mathsf{C}_{S_2}^{a'} \mathsf{g}^{b' p} = \mathsf{g}$.

Suppose that the input $a, b, p \in \mathbb{Z}$ satisfies p as a prime larger than $2^{6\lambda} B^3$ and $\mathsf{C}_D^a \cdot \mathsf{g}^{bp} = \mathsf{g}$. Because $\mathsf{C}_D^{c_2} = \mathsf{g}^{a_2 y}$, we have $\gcd(p, y) = 1$, otherwise one can find a p-root of g (from Lemma.3, one can find h such that $\mathsf{h}^p = \mathsf{C}_D$, and therefore we have $(\mathsf{h}^a \cdot \mathsf{g}^b)^p = \mathsf{g}$) and break the strong RSA assumption. There are two cases, $p \nmid x$ or $p | x$. In the first case that $p \nmid x$, from $\mathsf{C}_{S_1}^{c_3} = \mathsf{g}^{a_3 xy}$ and $\gcd(p, a_3 xy) = 1$, E_2 can efficiently compute $\alpha, \beta \in \mathbb{Z}$ such that $\alpha p + \beta a_3 xy$, and then $\mathsf{C}_{S_1}^{\beta c_3} \mathsf{g}^{\alpha p} = \mathsf{g}$. Setting $a' = \beta c_3', b' = \alpha$, then $\mathsf{C}_{S_2}^{a'} \mathsf{g}^{b' p} = \mathsf{g}$. In the second case that $p | x$, denoting $x_p = x/p$, from $\mathsf{C}_I^{c_1} = \mathsf{g}^{a_1 x} = (\mathsf{g}^{a_1 x_p})^p$ and Lemma.3, E_2 can efficiently compute g_a such that $\mathsf{C}_I = \mathsf{g}_a^p$. From $\mathsf{C}_I^{c_1'} = \mathsf{g}^{a_1' x'}$, $\mathsf{g}_a^{c_1' p} = \mathsf{g}^{a_1' x'}$. We have $p | a_1' x'$ otherwise from Lemma.3 one can find a p-root of g directly and break the strong RSA assumption. Because p is a prime larger than a_1', we have $p | x'$. Denote $x_p' = x'/p$. From $\mathsf{C}_{S_2}^{c_3'} = \mathsf{g}^{a_3' x' y'} = (\mathsf{g}^{a_3' x_p' y'})^p$ and Lemma.3, E_2 can efficiently compute g_b such that $\mathsf{C}_{S_2} = \mathsf{g}_b^p$, which concludes the proof. $\square$

D Constant-Size Zero-Knowledge Elementary Databases

In this section, we present a standard ZK-EDB scheme achieving constant commitment and proof size from groups of unknown orders.

Standard zero-knowledge elementary database scheme

$\mathsf{Setup}(1^\lambda)$: On input the security parameter 1^λ, Setup generates the description of an unknown-order group $\mathbb{G} \leftarrow GGen(\lambda)$ and a random group element $\mathsf{g} \leftarrow \mathbb{G}$. Suppose B is the upper bound of $\mathbb{G}$ (i.e., $B \geq |\mathbb{G}|$). Sample the description of a hash function $\mathcal{H}_{prime}$ that on input a string, outputs a random prime larger than $B_{prime} > 2^\lambda B$. Output CRS $\delta = (\mathbb{G}, \mathsf{g}, B, \mathcal{H}_{prime})$.

$\mathsf{Commit}(\delta, D)$: On input the set $D = \{(x_1, v_1), \cdots, (x_m, v_m)\}$, Commit hashes them into large primes, i.e., for $i \in [m]$, $p_i \leftarrow \mathcal{H}_{prime}(x_i)$. Denote by $t = \Sigma_{i\in[m]} v_i \cdot \Pi_{j\neq i} p_j \cdot (\Pi_{j\neq i} p_j^{-1} \mod p_i)$ such that $t \mod p_i = v_i$. Sample $r, r' \leftarrow [2^\lambda B]$, and output the commitment $C = (\mathsf{c}_1, \mathsf{c}_2) = (\mathsf{g}^{r\Pi_{i\in[m]} p_i}, \mathsf{g}^{t + r'\Pi_{i\in[m]} p_i})$ and the open information $\tau = (r, r', p_1, \cdots, p_m, D)$.

$\mathsf{Prove}(\delta, (C, \tau), x, D(x))$: Parse the input τ as $(r, r', p_1, \cdots, p_m, D)$. If $D(x) = v \neq \perp$, which means that $p = \mathcal{H}_{prime}(x) \in \{p_1, \cdots, p_m\}$, Prove outputs the proof $\pi = (\mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^p, \mathsf{c}_1; r\Pi_{i\in[m]} p_i / p), \mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^p, \mathsf{c}_2 \cdot \mathsf{g}^{-v}; (t + r\Pi_{i\in[m]} p_i - v)/p))$. If $D(x) = \perp$, which means that $p = \mathcal{H}_{prime}(x) \notin \{p_1, \cdots, p_m\}$ and $\gcd(p, r\Pi_{i\in[m]} p_i) = 1$, Prove finds a, b such that $ap + br\Pi_{i\in[m]} p_i = 1$ and outputs $\pi = (\mathsf{ZK}_{2DL}(\mathsf{g}^p, \mathsf{c}_1, \mathsf{g}; a, b))$.

$\mathsf{Verify}(\delta, C, x, D(x), \pi)$: If $D(x) = v \neq \perp$, check whether π consists of two valid $\mathsf{NI\text{-}ZK}_{DL}$ proofs for statement $(\mathsf{g}^p, \mathsf{c}_1)$ and $(\mathsf{g}^p, \mathsf{c}_2 \cdot \mathsf{g}^{-v})$. If $D(x) = \perp$, check whether π is a valid ZK_{2DL} proof for statement $(\mathsf{g}^p, \mathsf{c}_2, \mathsf{g}) \in \mathcal{R}_{2DL}$. Output 1 iff. the check pass.

Fig. 13: Protocol ZK-EDB

Theorem 5. *The protocol constructed in Fig.13 is a secure ZK-EDB scheme in the generic group model and random oracle model.*

proof sketch. The proof of the above theorem is similar to that of Theorem.2. The **completeness** is oblivious and the zero-knowledge property follows the zero-knowledge property of ZK_{DL} and ZK_{2DL} and the fact that the distribution of c_1, c_2 is statistically indistinguishable from the uniform distribution over group $\langle g \rangle$.

Because c_1 is actually a ZKS for set $Sup(D)$, from the soundness of ZKS scheme, no adversary can prove that $x \in Sup(D)$ and $x \notin Sup(D)$ simultaneously. Now, suppose there exists an adversary that can simultaneously prove $(x, v) \in D$ and $(x, v') \in D$, and $v \neq v'$. It then means that the adversary can generate two valid proofs ZK_{DL} for statements $(\mathsf{g}^p, \mathsf{c}_2 \cdot \mathsf{g}^{-v}), (\mathsf{g}^p, \mathsf{c}_2 \cdot \mathsf{g}^{-v'})$, where $p = \mathcal{H}_{prime}(x)$. From the soundness of ZK_{DL}, one can extract $(a, b), (a', b')$ such that $b, b' \leq 2^\lambda$, $\mathsf{g}^{pa} = (\mathsf{c}_2 \cdot \mathsf{g}^{-v})^b$ and $\mathsf{g}^{pa'} = (\mathsf{c}_2 \cdot \mathsf{g}^{-v'})^{b'}$. We then have $(\mathsf{g}^{ab'-a'b})^p = \mathsf{g}^{-vbb'+v'bb'}$. Furthermore, because $\gcd(p, -vbb' + v'bb')$, one can easily find a p-root of g from above equality, which breaks the strong RSA assumption and concludes the proof.

Remark 3. One can use the batch technique put forward in [BBF19] to batch the (non-)membership proofs. We show how to batch the ZKS proof in Remark.1.Because the above ZK-EDB scheme consists of a ZKS scheme and an "encoding" scheme showing the values associated with the keys, we only need to batch the proof for $(x, v) \in D$. To prove that $(x'_1, v'_1), \cdots, (x'_t, v'_t) \in D$, the prover hashes the keys into primes $p'_1, \cdots p'_t$ by $\mathcal{H}_{prime}$ and generates the proof $\pi \leftarrow \mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^{\Pi_{i\in[t]}p'_i}, \mathsf{C}; r\Pi_{i\in[m]}p_i/\Pi_{i\in[t]}p'_i), \mathsf{NI\text{-}ZK}_{DL}(\mathsf{g}^p, \mathsf{c}_2 \cdot \mathsf{g}^{-\tilde{v}}; (t + r\Pi_{i\in[m]}p_i - \tilde{v})/\Pi_{i\in[t]}p'_i))$ where $\tilde{v}$ is the least integer that satisfies $\tilde{v} \equiv v'_i \mod \Pi_{i\in[t]}p'_i)$ for each $i \in [t]$.

Proof size analysis of function queriable ZK-EDB. As shown in Fig.12, the proof for query f consists of a set operation proof of the ZKS, $|D(f)|$ membership proofs of the ZK-EDB and $2l|D(f)|$ (non-)membership proofs of the ZKS. When using the ZKS described in Section.4, the size of the set operation proof is linear to the size of $\mathcal{Q}$ (therefore the size of f) and l (Remark.2). In addition, as we show in (Remark.1), we can batch the $2l|D(f)|$ (non-)membership proofs of the ZKS into $2l$ proofs, the size of which is linear to l. Using the ZK-EDB constructed above, on can batch the proofs into constant group elements. Therefore the proof size of the function queriable ZK-EDB scheme is only linearly to the size of f and l (the size of the elements of D).