Amortized NISC over $\mathbb{Z}_{2^k}$ from RMFE

Fuchun Lin, Chaoping Xing, Yizhou Yao, and Chen Yuan

Shanghai Jiao Tong University
{linfuchun,xingcp,yaoyizhou0620,chen_yuan}@sjtu.edu.cn

Abstract. Reversed multiplication friendly embedding (RMFE) amortization has been playing an active role in the state-of-the-art constructions of MPC protocols over rings (in particular, the ring $\mathbb{Z}_{2^k}$). As far as we know, this powerful technique has NOT been able to find applications in the crown jewel of two-party computation, the non-interactive secure computation (NISC), where the requirement of the protocol being non-interactive constitutes a formidable technical bottle-neck. We initiate such a study focusing on statistical NISC protocols in the VOLE-hybrid model. Our study begins with making the *decomposable affine randomized encoding (DARE)* based semi-honest NISC protocol compatible with RMFE techniques, which together with known techniques for forcing a malicious sender Sam to honestly follow DARE already yield a secure amortized protocol, assuming both parties follow RMFE encoding. Achieving statistical security in the full malicious setting is much more challenging, as applying known techniques for enforcing compliance with RMFE incurs interaction. To solve this problem, we put forward a new notion dubbed non-malleable RMFE (NM-RMFE), which is a randomized RMFE such that, once one party deviates from the encoding specification, the randomness injected by the other party will randomize the output, preventing information from being leaked. NM-RMFE simultaneously forces both parties to follow RMFE encoding, offering a desired *non-interactive* solution to amortizing NISC. We believe that NM-RMFE is on its own an important primitive that has applications in secure computation and beyond, interactive and non-interactive alike. With an asymptotically good instantiation of our NM-RMFE, we obtain the first *statistical* reusable NISC protocols in the VOLE-hybrid model with *constant communication overhead* for arithmetic branching programs over $\mathbb{Z}_{2^k}$.
As side contributions, we consider computational security and present two concretely efficient NISC constructions in the random oracle model from conventional RMFEs.

1 Introduction

Non-interactive secure computation (NISC) [19] is referring, in particular, to a two-message secure *two-party computation* (2-PC), where the receiver Rachel publishes a message encrypting her private input x and a sender Sam, at any time, can use Rachel's message to complete a secure computation of $f(x, y)$, where y is his private input, by sending a single message to Rachel, which contains no information about y beyond $f(x, y)$. The importance of NISC is vividly illustrated by application scenarios such as profile matching in a dating website or DNA data comparing in an algorithm that tells whether two persons are related. In these application scenarios, both Sam's and Rachel's inputs contain sensitive personal information and are to be kept private, hence a secure 2-PC protocol should be implemented to complete the tasks. However, conventional 2-PC protocols require interaction, which means that Sam and Rachel need to be online at the same time and possibly exchange messages in multiple rounds. The synchronization, for one thing, and intolerance to communication latency, for another, put solutions involving *interactive* 2-PC protocols out of consideration. On the other hand, NISC (especially those allow Rachel's message to be reused by multiple senders, called *reusable NISC*, or rNISC for short) enables "public-key" variants of *secure computation* in the fashion that a public-key encryption scheme enables *secure transmission* of messages among strangers.

Without efficiency concerns, the problem can be solved in a simple two-step approach: one begins with any two-message 2-PC protocol secure against *semi-honest* parties, e.g. Yao's garbled circuit (GC) [25] or fully homomorphic encryption (FHE) [16], and then have both parties include a non-interactive zero-knowledge (NIZK) proof showing that their respective messages are honestly prepared. The caveat of this simple approach is that the statements to be proved involve cryptographic operations on secrets, which is in general inefficient.

NISC from oblivious transfer. In order to build NISC protocols for general functions in the oblivious transfer (OT)-hybrid model, Ishai et. al. [19] started with the semi-honest GC protocol. They used a statistical NISC for $\mathbf{NC}^0$ circuits to prove that Sam participates in the GC protocol honestly, avoiding the inefficient non-black-box use of NIZK and only making a black-box use of a pseudo-random generator (PRG). For $\mathbf{NC}^0$ circuits, there is an efficient statistical semi-honest two-message 2-PC in the OT-hybrid model using the so-called *decomposable affine randomized encoding* (DARE) [18,2]. The DARE allows to transform a circuit evaluation into parallel calls to an OT functionality, which in fact leaves no room for a malicious NISC receiver (as OT receiver) to cheat. And the desired statistical NISC for $\mathbf{NC}^0$ circuits can be obtained by applying the so-called *certified* OT [19] mechanism that allows Rachel to verify that Sam's inputs to these parallel OTs are honestly prepared. Though the asymptotic efficiency of the above protocol is rather appealing, it contains several ingredients that could incur large hidden constant in the concrete efficiency estimation. The followup work [1] devised a clever way to squash the interactive cut-and-choose to a single round, and obtained a concretely efficient NISC construction. With more sophisticated manipulations, this cut-and-choose approach was extended to yield amortized NISC protocols that allow to simultaneously evaluate multiple instances of the same circuit in order to reduce the cost [21]. This amortization technique seems to be very task-specific and not likely to be applied elsewhere.

Reusable NISC. An impossibility result concerning statistical *reusable* NISC in the OT-hybrid model was shown in [10], casting such protocols in the setting of parallel calls to a string OT, where Sam provides the input pair and Rachel provides the choice bit. The main observation is that such protocols satisfy that

Rachel's message can be separated into bits where each bit is interacting with only a small part of Sam's message. This allows a malicious Sam to apply the so-called *selective failure attack*. For instance, Sam honestly follows the protocol specification, except that, in one call to string OT, he replaces one of the two input strings by a uniform string and is caught cheating only when Rachel selects the tampered string, which occurs with probability 1/2. The authors of [10] then proposed a countermeasure through replacing OT with *oblivious linear function evaluation* (OLE), and constructed efficient statistical rNISC protocols in the OLE-hybrid model for branching programs over finite fields, which has high-level resemblance to the statistical NISC for $\mathbf{NC}^0$ circuits in [19]. The OLE functionality over a ring $\mathcal{R}$ allows Sam to input $a, b \in \mathcal{R}$ and Rachel to input $\alpha \in \mathcal{R}$, after which the functionality outputs $a \cdot \alpha + b$ to Rachel. Intuitively, replacing OT with OLE has the advantage that, the *selective failure attack* succeeds with probability related to the ring size (e.g. $1/|\mathcal{R}|$, if $\mathcal{R}$ is a finite field), which may be negligible by setting the ring to be sufficiently large. For implementing the rNISC protocols, a two-message reusable OLE protocol under the Paillier assumption was also presented in [10]. Informally, a reusable OLE protocol has the property that it is difficult for Sam to construct a partially correct message: any answer message Sam provides to Rachel is either accepted or rejected except with a negligible probability.

The followup work [13] improved the state-of-the-art of statistical rNISC through a new statistical proof system for circuit satisfiability called line-point zero-knowledge (LPZK) by a single Vector-OLE (VOLE) invocation. Another optimization comes from the implementation aspect through an efficient *pseudo-random correlation generator* (PCG) [5] construction for VOLE (similar to OT extension) based on a variant of learning parity with noise (LPN) [3] assumption. Instantiated with the reusable VOLE construction in [10] and the two-round PCG construction in [6], the resulting VOLE protocol has good concrete efficiency.

A very recent work [17] bypassed the impossibility result of [10], through making a black-box use of a secure two-message OT protocol. The authors of [17] followed the framework of [19], instead of extending the VOLE-hybrid framework, and showed a compiler that constructs an rNISC in the *random oracle model* for any Boolean function f, via a black-box use of any non-reusable NISC protocol that computes a related function f'. The non-reusable NISC protocol for f' was then instantiated with the construction of [19]. The main drawbacks are that their compiler is not statistical and incurs at least quadratic communication overhead in the security parameter [1].

rNISC over integer rings. Focusing on statistical security and concerning recent progress on rNISC for branching programs over fields, it is natural to ask whether these schemes can be *efficiently* adapted to work over arbitrary rings (in particular, the ring $\mathbb{Z}_{2^k}$, as the models of computation in real-life programming and the computer architectures are formulated as operations over $\mathbb{Z}_{2^{32}}$ or $\mathbb{Z}_{2^{64}}$).

[1] We remark that for some simple function f, e.g. branching programs, it seems that the non-reusable NISC can be instantiated with a lightweight NISC protocol, and the overhead is then optimized to linear in the security parameter.

More specifically, we are interested in finding out if it is possible to have a statistical rNISC protocol over the ring $\mathbb{Z}_{2^k}$ that matches the benchmarks of statistical rNISC protocols over large fields.

Given a branching program over a sufficiently large field, the statistical rNISC protocols for branching programs in [10,13] have constant communication overhead, compared to the semi-honest NISC (from DARE). We note that naively taking the above rNISC protocols over a field and replacing the field with a ring $\mathbb{Z}_{2^k}$ will ruin the security, due to the fact that the ring $\mathbb{Z}_{2^k}$ contains too many zero divisors (one half). This results in that the soundness error will keep a constant, no matter how large the ring $\mathbb{Z}_{2^k}$ is.

Similar problems have been under scrutiny in the study of arithmetic circuit MPC protocols throughout the last decade, yielding a plethora of important results. We highlight an extremely successful technique, the *reversed multiplication friendly embedding* (RMFE) [7,12]. Informally, an RMFE includes two maps ϕ, ψ, which allows to efficiently transform computations over a small field $\mathbb{F}_p$ (ring $\mathbb{Z}_{2^k}$) into its extension field $\mathbb{F}_{p^d}$ (extension ring $\mathtt{GR}(2^k, d)^2$) through ϕ, and the results over $\mathbb{F}_p$ can be efficiently recovered from the result of computations over the large field $\mathbb{F}_{p^d}$ (Galois ring $\mathtt{GR}(2^k, d)$) through ψ. Here we remark that the naive embedding (e.g. $\mathbb{F}_p \hookrightarrow \mathbb{F}_{p^d}$) is a special case of RMFE ($\mathbb{F}_p^m \hookrightarrow \mathbb{F}_{p^d}$). On the one hand, the RMFE technique provides amortization benefits (compared to naive embedding). On the other hand, non-trivial efforts should be made to force parties to follow the RMFE encoding honestly. As far as we know, RMFEs have been applied into honest majority MPC [7,9,12], dishonest majority MPC [15], VOLE-based ZK [20], and zk-SNARKs [8,4], etc. However, RMFEs have not been applied in the NISC setting yet.

1.1 Our Contributions

We put forward a new and novel RMFE technique that strengthens RMFEs, called *non-malleable RMFE* (NM-RMFE). We initiate the study of NISC over $\mathbb{Z}_{2^k}$ as well. With our NM-RMFE technique, we give the first asymptotically efficient statistical rNISC/VOLE for arithmetic branching programs over $\mathbb{Z}_{2^k}$. We also explore computational approaches to realize more concretely efficient NISC constructions for arithmetic branching programs over $\mathbb{Z}_{2^k}$ from RMFEs.

(1) The NM-RMFE is essentially a randomized variant of RMFE such that, when used in amortizing a secure 2-PC protocol, the randomness injected by the honest party prevents information about his/her private input from being leaked to the malicious party who cheats by providing an element not in the image of the map ϕ. NM-RMFE offers a conceptually simpler (removing the proofs) and, more importantly, *non-interactive* solution to forcing correct RMFE encoding: Rachel can directly implement a check mechanism in NM-RMFE to abort a cheating Sam, while simultaneously an honest Sam's input is not leaked to Rachel when Rachel is cheating. We believe that NM-RMFE is of independent

[2] see definition in Section 2.

interest in amortizing secure computation, interactive and non-interactive alike. We give an NM-RMFE construction (in Section 3.2) with the following features.

Theorem (informal) *There exists a family of $(m, d; D)$-NM-RMFE's from $\mathbb{Z}_{2^k}^m$ to $\mathtt{GR}(2^k, d)$, supporting multiplication for $D - 1$ times, with d/m asymptotically close to a constant.*

For instance, we construct a family of $(m, d; 2)$-NM-RMFEs over $\mathbb{Z}_{2^k}$ with $\frac{d}{m} \to 29.13, m \to \infty$ and a family of $(m, d; 3)$-NM-RMFEs over $\mathbb{Z}_{2^k}$ with $\frac{d}{m} \to 80.15, m \to \infty$.

(2) Regarding NISC protocols over $\mathbb{Z}_{2^k}$, we have the following informal theorem (induced by Theorem 4).

Theorem (informal) *There exists a statistical rNISC protocol computing branching programs over $\mathbb{Z}_{2^k}$ with communication overhead close to a constant.*

The above theorem indicates that the amortized efficiency of our construction asymptotically matches the state-of-the-art statistical rNISC protocol over large fields [13]. Though our exposition highlights the most useful special cases of arithmetic circuits over $\mathbb{Z}_{2^k}$, all protocols straightforwardly extend to $\mathbb{F}_{p^k}$ and $\mathbb{Z}_{p^k}$ for arbitrary prime p. Before this work, statistical rNISC over small fields [10,13] had to pay overhead at least linear in the security parameter (which are realized by rNISC over large fields). More importantly, there was no efficient constructions for $\mathbb{Z}_{2^k}$. Our results bridge the gaps left behind by the difference between computation domains in an amortized sense (asymptotic nature).

(3) As side contributions, we present a maliciously secure rNISC construction for computing branching programs over $\mathbb{Z}_{2^k}$ from a random oracle aided cut-and-choose, through making a black-box use of any two-message reusable VOLE protocol over $\mathtt{GR}(2^k, d)$ (inspired by the approach of [17]). We also present a highly efficient maliciously secure NISC construction for computing branching programs over $\mathbb{Z}_{2^k}$ in the OT-hybrid model.

1.2 Technical Overview

The challenge for constructing NM-RMFE lies in the fact that the notion itself demands the coexistence of two conflicting properties: multiplication friendliness (malleability for valid multiplicands) and non-malleability (against one invalid multiplicand). There was no cryptographic primitive of this flavor in the literature, as far as we know. One must turn to known constructions for each property separately for inspirations and hope that they can be combined. The RMFE concatenation technique plays an important role in constructing binary RMFE by concatenating two RMFE's. On the other hand, the Fujisaki-Okamoto (FO) transform (widely used in e.g. post-quantum cryptography NIST submissions) uses two encryption schemes, one encrypting the payload while the other one encrypting the random key of the first to enable a consistency check via checking whether the ciphertext of the second encryption scheme is valid. We import the core idea of the FO transformation into RMFE concatenation by injecting randomness into the first RMFE and use the its encoding relation for FO style "cipher text" check.

For rNISC/VOLE constructions, we begin with recalling the construction of rNISC/VOLE for branching programs in [13]. The semi-honest protocol is an execution of t parallel VOLE's over a large enough Galois field $\mathbb{F}$, where t is the number of components in Rachel's input $x \in \mathbb{F}^t$. Rachel's inputs to the VOLE's are simply the t components of x. Sam's inputs to the VOLE's are generated using a DARE scheme that, given y, the branching program $f(\cdot,\cdot)$ and Sam's private randomness, produces t pairs of vectors (each pair of such vectors define an *affine line*, hence the name DARE). Note that the semi-honest protocol already leaves a malicious Rachel no room to cheat. The malicious protocol only needs to make sure that Sam's t input lines to the VOLE's in the semi-honest protocol are indeed the result of running the DARE scheme using $f(\cdot,\cdot)$, some secret y and some secret randomness. If one adds a new VOLE to the semi-honest protocol and let Sam describe the DARE scheme specification as an arithmetic circuit C to invoke the LPZK proof system using his t input lines as witness, then the only room for Sam to cheat in this intermediate protocol is to fake the consistency of the t lines and LPZK witness encoded in the new VOLE instance. The above consistency check problem boils down to a mechanism called *VOLE with equality constraint (eVOLE)* that allows Sam to prove equivalence of an arbitrary component in the two vectors defining one line and some component in the two vectors defining the other line. To complete the malicious rNISC/VOLE protocol, one more instance of VOLE is then added where Rachel's input is a uniform point $\beta \in \mathbb{F}$ (input randomizer for eVOLE construction), and eVOLE is invoked to prove consistency of Sam's input lines between this copy of VOLE (serving as a bridge) and all other copies of VOLE's.

We are now in good position to describe our constructions for rNISC/VOLE over $\mathbb{Z}_{2^k}$. Our exposition begins with an observation that the Galois ring $\mathtt{GR}(2^k, d)$ behaves very similarly to the Galois field $\mathbb{F}_{2^d}$ with respect to constructing building blocks LPZK and eVOLE in the recalled malicious protocol. This suggests that we could view the inputs of Sam and Rachel as consisting of elements in $\mathtt{GR}(2^k, d)$ and try to design a rNISC/VOLE over $\mathtt{GR}(2^k, d)$. But this idea alone does not give us the desired efficiency, as the choice of the extension degree d depends on the security parameter (we need 2^d to be roughly the size of $\mathbb{F}$ in the recalled protocol). To circumvent this caveat, we embed multiple elements of $\mathbb{Z}_{2^k}$ into a single element of $\mathtt{GR}(2^k, d)$ and make sure that computation (in particular, multiplication) is still "preserved" under this embedding. If we could make the RMFE techniques work with the recalled rNISC/VOLE framework, the cost of operating over a large ring $\mathtt{GR}(2^k, d)$ will be amortized by executing $\Omega(d)$ copies of computation over $\mathbb{Z}_{2^k}$ and we are done.

The first challenge comes from the DARE scheme in the semi-honest protocol. Unlike the conventional masking approach to privacy (adding a one-time-pad to the sensitive value before computing on it and remove the pad afterwards), the DARE scheme hides information through multiplying the sensitive matrix by two random structured matrices from the left and the right, respectively (effectively hiding the sensitive matrix among the set of such matrices that have the same determinant, hence destroying other information about the matrix than its determinant). This subtle difference already causes big troubles in the semi-honest model when naively compiling the DARE scheme with RMFE. Recall that an RMFE over $\mathbb{Z}_{2^k}$ includes two $\mathbb{Z}_{2^k}$-linear maps, the embedding map ϕ and the decoding map ψ. Note that ϕ is not surjective (being an embedding) and the embedding only preserves multiplication for $\mathtt{GR}(2^k, d)$ elements that lie in the image of ϕ. Therefore, one needs to carefully analyze the effect of this fact on the correctness and privacy of the DARE scheme. If we were to sample entries of the structured random matrices over the entire $\mathtt{GR}(2^k, d)$, we could encounter multiplication by a $\mathtt{GR}(2^k, d)$ element that does not lie in the image of ϕ, which would damage the correctness of the DARE scheme. From now on, assume we sample entries of the structured random matrices over the image of ϕ only. The DARE scheme involves computing matrix multiplication for two times (for correctness we can use Degree-3 RMFEs [14]). We next analyze whether the privacy of the DARE scheme is affected by RMFE. We remark that the product of two elements in the image of ϕ may no longer remain in the image of ϕ, which may reveal more information than we expect. We solve this by masking with a random element in the kernel of ψ.

The second challenge comes from making malicious Sam follow the RMFE encoding in our semi-honest protocol and making malicious Rachel correctly encode her input using RMFE, simultaneously. Jumping ahead, note that once the correct RMFE encoding is guaranteed, the rest of the security proof against a malicious adversary follows straightforwardly using analogy to the recalled protocol over fields (we include a self-contained exposition of the building blocks LPZK and eVOLE over $\mathtt{GR}(2^k, d)$ in Section 4.2 for completeness). This second challenge is a huge bottle-neck because none of the known RMFE techniques come close to suggesting a workable idea. The standard RMFE techniques for constructing interactive secure computation protocols do have a (V)OLE-based variant [20], where elements in $\mathtt{GR}(2^k, d)$ are restricted in image of ϕ (see the exposition in Section 5.1, where we do use it in our two side contributions). The difficulties lie in removing the interaction that is liberally in use and seemingly inherent. For one thing, sacrifice is used to generate correlated randomness that enables the re-embedding VOLE functionality. This can be made non-interactive at the cost of using a random oracle, hence settling for computational security. More seriously, the above process of correlated randomness generation is only capable of allowing one party (VOLE sender) to prove correct RMFE encoding to the other party (VOLE receiver). This means that we would not be able to prove correct RMFE encoding for Sam and Rachel simultaneously without interaction. We put forward the notion of NM-RMFE and propose a statistical instantiation of NM-RMFE that solves this bottle-neck problem. In a high level, strengthening RMFE to NM-RMFE allows for "extraction" in the simulation, which means that the simulation will go through no matter how the adversary deviates from the NM-RMFE encoding. Combining all pieces together, we obtain a statistical rNISC/VOLE for computing branching programs over $\mathbb{Z}_{2^k}$ with asymptotic efficiency as the rNISC/VOLE over large fields.

Computational NISC. Finally, we explore standard computationally secure techniques for forcing both parties to follow the RMFE encoding honestly. For the malicious sender side, we augment the above Galois ring analogue *certified* VOLE by substituting the VOLE-hybrid model with the re-embedding VOLE-hybrid model following the idea of [20]. For the malicious receiver side, we define a variant of VOLE over $\mathtt{GR}(2^k, d)$, where the receiver's inputs are restricted in the image of ϕ, and provide two instantiations. The former instantiation (inspired by [17]) uses the random oracle to realize a non-interactive cut-and-choose such that Rachel "proves" to Sam her input is in the image of ϕ. The latter construction comes from an observation that for a correlated OT-based VOLE construction, since the image of ϕ is actually a linear space over $\mathbb{Z}_{2^k}$, the bits that Rachel sends to correlated OTs one-to-one correspond to an element in the image of ϕ as long as the number of correlated OTs is restricted to the size of the image of ϕ.

2 Preliminaries

Notations. In this paper, bold letters (e.g. $\boldsymbol{a}, \boldsymbol{b}$) are used to denote vectors. We use x_i to denote the i_{th}-component of the vector $\boldsymbol{x}$ (similarly $x_{i,j}$ for the j_{th}-component of $\boldsymbol{x}_i$). We use $[a, b]$ (or $[a, b+1)$ sometimes) to denote the set of integers in the range from a to b. If $a = 1$, it is simplified by $[b]$. We also use $\boldsymbol{x}|_J$ to denote the set $\{x_i \mid i \in J\}$. We use $x \overset{\$}{\leftarrow} \mathcal{R}$ to denote that x is uniformly sampled from a ring $\mathcal{R}$ and denote the uniform distribution over $\mathcal{R}$ by $\mathrm{U}_{\mathcal{R}}$. For a map $\phi : \mathcal{R}_1 \to \mathcal{R}_2$, we naturally extend it to be defined over vector space $\mathcal{R}_1^n$ and matrix space $\mathcal{R}_1^{m \times n}$. Let $\mathrm{Im}(\phi)$ denote the set $\{\phi(x) \mid x \in \mathcal{R}_1\}$ and $\mathrm{Ker}(\phi)$ denote the set $\{x \in \mathcal{R}_1 \mid \phi(x) = 0\}$. For a commitment scheme, we use the notation $[\![\alpha]\!]$ to denote the commitment of α. For two distributions $\mathcal{D}_1, \mathcal{D}_2$, we use the notation $\mathcal{D}_1 \overset{s}{\approx} \mathcal{D}_2$ to denote that they are statistically close.

Galois Rings. Let p be a prime, and $k, d \geq 1$ be integers. Let $f(X) \in \mathbb{Z}_{p^k}[X]$ be a monic polynomial of degree d such that $\overline{f(X)} := f(X) \mod p$ is irreducible over $\mathbb{F}_p$. A Galois ring over $\mathbb{Z}_{p^k}$ of degree d denoted by $\mathtt{GR}(p^k, d)$ is a ring extension $\mathbb{Z}_{p^k}[X]/(f(X))$ of $\mathbb{Z}_{p^k}$. We refer to the textbook [24] for a friendly exposition. Same as the special case of Galois fields, there is a bound on the number of roots for a nonzero polynomial over $\mathtt{GR}(p^k, d)$.

Lemma 1 ([24]). *A nonzero degree-r polynomial over $\mathtt{GR}(p^k, d)$ has at most $rp^{(k-1)d}$ roots.*

Lemma 1 immediately gives that for any nonzero degree-r polynomial $f(x)$ over $\mathtt{GR}(p^k, d)$, we have that

$$\Pr\Big[f(\alpha) = 0 \;\Big|\; \alpha \overset{\$}{\leftarrow} \mathtt{GR}(p^k, d)\Big] \leq rp^{-d}.$$

In particular, we have that $1/p^d$ fraction of elements are zero divisors in $\mathtt{GR}(p^k, d)$.

Degree-D RMFE. The reverse Multiplicative Friendly Embedding (RMFE for short) was first introduced in [7], which packs multiple multiplications over a field $\mathbb{F}_q$ to one multiplication over its extension $\mathbb{F}_{q^d}$. It was further shown in [12] that RMFEs over Galois fields ($\mathbb{F}_{p^r}^m \to \mathbb{F}_{p^{rd}}$) induce RMFEs over Galois rings ($\mathtt{GR}(p^k, r)^m \to \mathtt{GR}(p^k, rd)$). Degree-$D$ RMFE [14] is a natural generalization of RMFE, which supports multiplication for upto $D - 1$ times.

Definition 1. *Let p be a prime, $k, r, m, d, D \geq 1$ be integers. A pair (ϕ, ψ) is called an $(m, d; D)$-RMFE over $\mathtt{GR}(p^k, r)$, if $\phi : \mathtt{GR}(p^k, r)^m \to \mathtt{GR}(p^k, rd)$ and $\psi : \mathtt{GR}(p^k, rd) \to \mathtt{GR}(p^k, r)^m$ are two $\mathtt{GR}(p^k, r)$-linear maps such that*

$$\psi(\phi(\boldsymbol{x}_1) \cdot \phi(\boldsymbol{x}_2) \cdots \phi(\boldsymbol{x}_D)) = \boldsymbol{x}_1 * \boldsymbol{x}_2 * \cdots * \boldsymbol{x}_D$$

for all $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_D \in \mathtt{GR}(p^k, r)^m$, where $*$ denotes the entry-wise multiplication.

Standard RMFEs are essentially degree-2 RMFEs. Degree-D RMFEs have the following properties, which are generalized from the degree-2 case.

Lemma 2 ([20]). *Let (ϕ, ψ) be an $(m, d; D)$-RMFE over Galois ring $\mathtt{GR}(p^k, r)$. We have that $\mathtt{GR}(p^k, rd)$ is the direct sum of $\mathrm{Ker}(\psi), \phi(\mathbf{1})^{D-1} \cdot \mathrm{Im}(\phi)$, where $\mathbf{1}$ denotes the vector of all 1's. That is $\mathtt{GR}(p^k, rd) = \mathrm{Ker}(\psi) \oplus (\phi(\mathbf{1})^{D-1} \cdot \mathrm{Im}(\phi))$.*

As shown in [14], there always exists an $(m, d; D)$-RMFE over Galois ring $\mathtt{GR}(p^k, r)$ with $\phi(\mathbf{1}) = 1$. Thus, we always assume $\phi(\mathbf{1}) = 1$ for the rest of this paper. Then, the above lemma indicates that ψ introduces a bijection when restricted on $\mathrm{Im}(\phi)$. We have the following lemma that indicates the asymptotic behavior of degree-D RMFEs.

Lemma 3 ([14]). *There exists a family of $(m, d; D)$-RMFE over $\mathbb{Z}_{2^k}$ for all $k \geq 1$ with $m \to \infty$ and $\frac{d}{m} \to \frac{1+2D}{3}(D + \frac{D(3+1/(2^D-1))}{2^{D+1}-1})$.*

For instance, when $m \to \infty$, there exists a family of $(m, d; 2)$-RMFEs over $\mathbb{Z}_{2^k}$ with $\frac{d}{m} \to 4.92$. and a family of $(m, d; 3)$-RMFEs over $\mathbb{Z}_{2^k}$ with $\frac{d}{m} \to 8.47$.
VOLE. The (random) vector oblivious linear function evaluation (VOLE) is a two-party primitive that allows two parties $P_\mathcal{S}, P_\mathcal{R}$ to obtain random correlated values. In more detail, the sender $P_\mathcal{S}$ obtains two random vectors $\boldsymbol{a}, \boldsymbol{b}$, while the receiver $P_\mathcal{R}$ obtains a random scalar α and a random vector $\boldsymbol{v}$ such that $\boldsymbol{v} = \boldsymbol{a} \cdot \alpha + \boldsymbol{b}$ holds. We formalize the ideal VOLE functionality over arbitrary ring $\mathcal{R}$ in Figure 1. We also use the chosen-input variant of VOLE in this paper, where $(\boldsymbol{a}, \boldsymbol{b}), \alpha$ are provided by the sender and the receiver, respectively. The above VOLE correlation can be viewed as a linear homomorphic Message Authentication Code (MAC) that authenticates $\boldsymbol{a}$ using the MAC key α, denoted by $[\boldsymbol{a}]_\alpha$.
Non-Interactive Secure Computation. We follow the VOLE-based reusable Non-interactive Secure Computation (rNISC) definition in [13]. In a high level, the sender $P_\mathcal{S}$ encodes its input as multiple lines ($P_\mathcal{S}$'s VOLE inputs) and the receiver $P_\mathcal{R}$ encodes its input as multiple points ($P_\mathcal{R}$'s VOLE inputs), one for each line. In reusable security, a malicious sender $\mathcal{A}$ can learn whether the receiver rejects its possibly illegal messages after each execution. We give the formal definition of rNISC as follows.

Functionality $\mathcal{F}^{\mathcal{R}}_{\text{VOLE}}$

Parameterized by a ring $\mathcal{R}$, length parameters $l_1, ..., l_n \in \mathbb{N}$.

Setup phase: Upon receiving $(sid; \texttt{initialize})$ from $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$, sample $\alpha \overset{\$}{\leftarrow} \mathcal{R}$ and store $(sid; \alpha)$, and ignore any further inputs from $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$ with the same session identifier sid. Send α to $P_{\mathcal{R}}$.

Send phases: Upon receiving $(sid; \texttt{send}; l_i)$ from $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$, verify that there are stored values $(sid; \alpha)$; else, ignore that message. Sample $\boldsymbol{a}, \boldsymbol{b} \overset{\$}{\leftarrow} \mathcal{R}^{l_i}$, and store $(sid; \boldsymbol{a}, \boldsymbol{b}; l_i)$, and ignore any further inputs from $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$ with the same session identifier sid. Send $(\boldsymbol{a}, \boldsymbol{b})$ to $P_{\mathcal{S}}$ and $\boldsymbol{v} := \boldsymbol{a} \cdot \alpha + \boldsymbol{b}$ to $P_{\mathcal{R}}$.

Fig. 1: Ideal functionality for random VOLE over $\mathcal{R}$.

Definition 2 (rNISC). *An VOLE-based reusable non-interactive secure computation (NISC) protocol for an arithmetic function $f : \mathcal{R}^{n_1} \times \mathcal{R}^{n_2} \rightarrow \mathcal{R}^t$ consists of a triple of algorithms $(R1, S, R2)$ defined as follows:*

- *$R1(\mathcal{R}, \boldsymbol{x})$ is a* PPT *algorithm that, given an input $\boldsymbol{x} \in \mathcal{R}^{n_1}$, outputs points $(\alpha_1, ..., \alpha_{n'}) \in \mathcal{R}^{n'}$ and auxiliary information* $\texttt{aux}$*.*
- *$S(\mathcal{R}, \boldsymbol{y})$ is a* PPT *algorithm that, given an input $\boldsymbol{y} \in \mathcal{R}^{n_2}$, outputs n' pairs of vectors $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathcal{R}^{l_i}$, each specifying an affine line $\boldsymbol{v}_i(\alpha) := \boldsymbol{a}_i \cdot \alpha + \boldsymbol{b}_i$.*
- *$R2(\mathcal{R}, (\boldsymbol{v}_1, ..., \boldsymbol{v}_{n'}), \texttt{aux})$ is a polynomial-time algorithm, such that given n' evaluations $\boldsymbol{v}_i \in \mathcal{R}^{l_i}$ and auxiliary information* $\texttt{aux}$*, outputs either $\boldsymbol{z} \in \mathcal{R}^t$ or $\perp$.*

We say the algorithms $(R1, S, R2)$ has reusable malicious security, if the following security requirements hold:

- ***Completeness.*** *As long as $R2$ takes inputs $\boldsymbol{v}_i = \boldsymbol{v}_i(\alpha_i)$, for $i \in [n']$, where $\boldsymbol{v}_i(\alpha)$ and α_i are given by S and $R1$, respectively, we have that $R2$ outputs $\boldsymbol{z} = f(\boldsymbol{x}, \boldsymbol{y})$.*
- ***Reusable ε-security against malicious sender.*** *There exists a polynomial-time extractor* Ext *such that given n' lines $\boldsymbol{v}_i^*(t) := \boldsymbol{a}_i^* \cdot \alpha + \boldsymbol{b}_i^*$ with vectors $\boldsymbol{a}_i^*, \boldsymbol{b}_i^* \in \mathcal{R}^{l_i}$, outputs $\boldsymbol{y}^* \in \mathcal{R}^{n_2}$ or $\perp$ with the following holds: for every honest receiver's input $\boldsymbol{x} \in \mathcal{R}^{n_1}$, the receiver's output $\boldsymbol{z} := R2(\mathcal{R}, (\boldsymbol{v}_1^*, ..., \boldsymbol{v}_{n'}^*), \texttt{aux})$ is equal to $f(\boldsymbol{x}, \boldsymbol{y}^*)$ except with $\leq \varepsilon$ probability over the receiver's randomness. The random-input variant of the above definition is also used in this paper, where the probability is over both the receiver's randomness and an $\boldsymbol{x}$ sampled from $\mathcal{R}^{n_1}$ uniformly at random.*
- ***Statistical security against malicious receiver.*** *There exist a polynomial-time extractor algorithm* Ext *and* PPT *simulator algorithm* Sim *such that, given points $\alpha_1^*, ..., \alpha_{n'}^* \in \mathcal{R}$,* Ext *outputs effective $\boldsymbol{x}^* \in \mathcal{R}^{n_1}$ with the following holds: for every honest sender's input $\boldsymbol{y} \in \mathcal{R}^{n_2}$, the output distribution of $\mathsf{Sim}(\mathcal{R}, f(\boldsymbol{x}^*, \boldsymbol{y}))$ is statistically close to $\{(\boldsymbol{v}_1(\alpha_1^*), ..., \boldsymbol{v}_{n'}(\alpha_{n'}^*)) \mid (\boldsymbol{v}_1(\alpha), ..., \boldsymbol{v}_{n'}(\alpha)) \leftarrow S(\mathcal{R}, \boldsymbol{y})\}$.*

Branching Program. In this paper, we mainly consider arithmetic functions that can be represented by branching programs [18].

Definition 3 (Branching Program over $\mathcal{R}$). *A branching program (BP) over $\mathcal{R}$ is defined by a quadruple* $\mathrm{BP} = (G, \varphi, v, t)$*, where* $G = (V, E)$ *is a directed acyclic graph,* φ *is an edge labeling function assigning each edge a degree-1 polynomial in a single input variable* x_i*, and* v, t *are two special vertices. The size of BP is the number of vertices in G. Each input assignment* $\boldsymbol{x} = (x_1, ..., x_n) \in \mathcal{R}^n$ *induces an assignment* $G_{\boldsymbol{x}}$ *of a value from* $\mathcal{R}$ *to each* $e \in E$*. The output* $\mathrm{BP}(\boldsymbol{x})$ *is defined as the sum of the weights of all directed paths from* v *to* t *in* $G_{\boldsymbol{x}}$*, where the weight of a path is the product of the values of its edges.*

Let $\mathrm{BP} = (G, \varphi, v, t)$ be a BP of size $s + 1$ over $\mathcal{R}$, computing a function $f : \mathcal{R}^n \to \mathcal{R}$. Fix some topological ordering of the vertices of G, where the source vertex v is labeled 1 and the terminal vertex t is labeled $s + 1$. For any input $\boldsymbol{x}$, let $A_{\boldsymbol{x}}$ be the $(s + 1) \times (s + 1)$ matrix over $\mathcal{R}$ whose (i, j) entry contains the value assigned by φ to the edge (i, j) (or 0 if there is no such edge). Define $L(\boldsymbol{x})$ as the submatrix of $A_{\boldsymbol{x}} - I$ obtained by deleting column v and row t (i.e. the first column and the last row). Note that each entry of $L(\boldsymbol{x})$ has degree (at most) 1 in the inputs $\boldsymbol{x}$; moreover, $L(\boldsymbol{x})$ contains the constant -1 in each entry of its second diagonal (the one below the main diagonal) and the constant 0 below this diagonal. We have the fact that $f(\boldsymbol{x}) = det(L(\boldsymbol{x}))$, and we say $L(\boldsymbol{x})$ is induced by a BP that computes f.

We briefly introduce the so-called "Decomposable Affine Randomized Encoding" (DARE) for branching programs [18,2]. We begin by a simple randomization lemma.

Lemma 4 ([18]). *Let* $\mathcal{H}$ *be a set of square matrices over* $\mathcal{R}$*, and* $\mathcal{G}_1, \mathcal{G}_2$ *be multiplicative groups of matrices of the same dimension as* $\mathcal{H}$*. Denote by '*$\sim$*' the equivalence relation on* $\mathcal{H}$ *defined by:* $H \sim H'$ *iff there exists* $G_1 \in \mathcal{G}_1, G_2 \in \mathcal{G}_2$ *such that* $H = G_1 H' G_2$*. Let* R_1, R_2 *be uniformly and independently distributed matrices from* $\mathcal{G}_1, \mathcal{G}_2$*, respectively. Then, for any* H, H' *such that* $H \sim H'$*, the random variables* $R_1 H R_2$ *and* $R_1 H' R_2$ *are identically distributed.*

The above lemma can be instantiated with the following matrix sets:

- $\mathcal{H}^s$ consists of all $s \times s$ matrices over $\mathbb{Z}_{2^k}$ with -1's in the second diagonal (the diagonal below the main diagonal), and 0's below the second diagonal.
- $\mathcal{G}_1^s$ consists of all $s \times s$ matrices over $\mathbb{Z}_{2^k}$ with 1's on the main diagonal and 0's below the main diagonal.
- $\mathcal{G}_2^s$ consists of all $s \times s$ matrices over $\mathbb{Z}_{2^k}$ with 1's on the main diagonal and 0's in all of the remaining entries except those of the rightmost column.

Let $L(\boldsymbol{x})$ be a matrix induced by a size $(s + 1)$ BP over $\mathbb{Z}_{2^k}$ computing $f : \mathbb{Z}_{2^k}^n \to \mathbb{Z}_{2^k}$. We have the following corollary.

Corollary 1. *Let* R_1, R_2 *be uniformly and independently distributed matrices from* $\mathcal{G}_1^s, \mathcal{G}_2^s$*, respectively. We have that* $R_1 L(\boldsymbol{x}) R_2$ *reveals nothing about* $L(\boldsymbol{x})$ *but* $det(L(\boldsymbol{x}))$*.*

Essentially, $R_1 L(\boldsymbol{x}) R_2$ in Corollary 1 is a randomized encoding of $L(\boldsymbol{x})$, and the above procedure is referred as DARE.

3 Non-Malleable RMFE

Before we show how to construct NISC/VOLE over $\mathbb{Z}_{2^k}$, we first introduce our main innovation separately, as we believe it is of independent interest. We start with introducing the conception of NM-RMFE, followed by a construction.

3.1 Non-Malleable RMFE

To better illustrate the benefits of upgrading RMFEs to NM-RMFEs, let us first consider a simple NISC/VOLE task as a warm-up, where $P_{\mathcal{S}}$ has inputs $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathbb{Z}_{2^k}^l$, $P_{\mathcal{R}}$ has inputs α_i, and $P_{\mathcal{R}}$ wants to obtain $\boldsymbol{v}_i := \boldsymbol{a}_i \cdot \alpha_i + \boldsymbol{b}_i$, for $i \in [m]$, i.e. the task for parallel VOLE over $\mathbb{Z}_{2^k}$. This can be done with the help of an $(m, d; 2)$-RMFE (ϕ, ψ) over $\mathbb{Z}_{2^k}$ and a VOLE functionality $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$. We show what would go wrong if RMFE encodings are not honestly computed through this example.

In more detail, $P_{\mathcal{S}}$ picks $\boldsymbol{r} \overset{\$}{\leftarrow} \text{Ker}(\psi)^l$ and sends $\phi(\boldsymbol{a}_1, ..., \boldsymbol{a}_m)$, $\phi(\boldsymbol{b}_1, ..., \boldsymbol{b}_m) + \boldsymbol{r}$ to $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$, while $P_{\mathcal{R}}$ sends $\phi(\alpha_1, ..., \alpha_m)$. Finally, $P_{\mathcal{R}}$ receives $\boldsymbol{v}$ from $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$, and outputs $(\boldsymbol{v}_1, ..., \boldsymbol{v}_m) := \psi(\boldsymbol{v})$. We remark that the mask $\boldsymbol{r}$ is necessary and sufficient for the privacy of $P_{\mathcal{S}}$'s private inputs. In fact, we only want $P_{\mathcal{R}}$ to obtain $\psi(\boldsymbol{v})$, but $P_{\mathcal{R}}$ actually receives $\boldsymbol{v}$. The potential leakage then is prevented by masking with $\boldsymbol{r}$, by Lemma 2.

The above protocol achieves semi-honest security, but not malicious security. The main obstacle is that, for example, if a malicious receiver takes an input $Y \notin \text{Im}(\phi)$, the simulator cannot "extract" a $\boldsymbol{y'}$ from Y (the simulation will go through in the semi-honest model, by setting $\boldsymbol{y'} := \psi(Y)$). Existing works [9,15,20] have developed methods to solve this issue, by letting the adversary to prove that $Y \in \text{Im}(\phi)$. However, these approaches are either not statistical or interactive. Instead, we solve the issue statistically by putting forward the notion of Non-Malleable RMFE, which conceptually allows $Y \notin \text{Im}(\phi)$. For the sake of generality, we define Degree-D Non-Malleable RMFE as follows:

Definition 4 (Degree-D NM-RMFE). *Let $\mathtt{GR}(p^k, r)$ be a Galois ring and κ be the statistical security parameter. A pair of maps (ϕ, ψ) is called an $(m, d; D)$-NM-RMFE over $\mathtt{GR}(p^k, r)$, if it has the following properties:*

1. *$\phi : \mathtt{GR}(p^k, r)^m \times \{0,1\}^{\mathcal{O}(\kappa)} \to \mathtt{GR}(p^k, rd)$, $\psi : \mathtt{GR}(p^k, rd) \to \mathtt{GR}(p^k, r)^m \cup \{\perp\}$ are $\mathtt{GR}(p^k, r)$-linear maps[3], satisfying*

$$\psi(\phi(\boldsymbol{x}_1, r_1) \cdot \phi(\boldsymbol{x}_2, r_2) \cdots \phi(\boldsymbol{x}_D, r_D)) = \boldsymbol{x}_1 * \boldsymbol{x}_2 * \cdots * \boldsymbol{x}_D,$$

for any $\boldsymbol{x}_1, ..., \boldsymbol{x}_D \in \mathtt{GR}(p^k, r)^m$ and $r_1, ..., r_D \overset{\$}{\leftarrow} \{0,1\}^\kappa$.

[3] More precisely, ϕ is $\mathtt{GR}(p^k, r)$-linear on $\mathtt{GR}(p^k, r)^m$.

2. *if $Y \notin \mathrm{Im}(\phi)$, there exists a constant $\boldsymbol{y} \in \mathtt{GR}(p^k, r)^m$, such that for any $\boldsymbol{x}_1, ..., \boldsymbol{x}_{D-1} \in \mathtt{GR}(p^k, r)^m$, we have*

$$\psi(\phi(\boldsymbol{x}_1) \cdots \phi(\boldsymbol{x}_{D-1}) \cdot Y) = \boldsymbol{x}_1 * \cdots * \boldsymbol{x}_{D-1} * \boldsymbol{y} + \boldsymbol{\delta},$$

where $\boldsymbol{\delta} \sim \mathcal{D}_{\boldsymbol{x},Y} \overset{s}{\approx} \mathcal{D}_Y$ and $\mathcal{D}_Y$ is a PPT*-sampleable distribution over $\mathtt{GR}(p^k, r)^m \cup \{\bot\}$ determined only by Y. We use the convention that for any $\boldsymbol{z} \in \mathtt{GR}(p^k, r)^m$, $\boldsymbol{z} + \bot = \bot$ to make ψ well-defined.*

Note that the above definition includes Degree-D NM-RMFE over Galois fields, as $\mathtt{GR}(p^k, r)$ is a field when $k = 1$. According to property 1, we can specify the distribution $\mathcal{D}_Y$ for $Y \in \mathrm{Im}(\phi)$, such that $\boldsymbol{\delta} \leftarrow \mathcal{D}_Y$, $\Pr[\boldsymbol{\delta} = \boldsymbol{0}] = 1$. We remark that in a high level, NM-RMFE allows for "extraction". Using an $(m, d; 2)$-NM-RMFE (ϕ, ψ) over $\mathbb{Z}_{2^k}$ instead of $(m, d; 2)$-RMFEs over $\mathbb{Z}_{2^k}$, we immediately obtain a reusable malicious secure VOLE scheme over $\mathbb{Z}_{2^k}$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model without any additional cryptographic primitives (see Figure 2). We have the following theorem.

Protocol $\mathit{\Pi}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$

Parameterized by $\mathtt{GR}(2^k, d)$, length l. Suppose $P_{\mathcal{R}}$ has her private inputs $\boldsymbol{\alpha} \in \mathbb{Z}_{2^k}^m$, $P_{\mathcal{S}}$ has his private inputs $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathbb{Z}_{2^k}^l$, for $i \in [m]$. Let (ϕ, ψ) be an $(m, d; 2)$-NM-RMFE over $\mathbb{Z}_{2^k}$.

1. The receiver $P_{\mathcal{R}}$: Compute $\mathit{\Delta} := \phi(\boldsymbol{\alpha})$. Send $\mathit{\Delta}$ to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$.
2. The sender $P_{\mathcal{S}}$: Compute $\boldsymbol{A} := \phi(\boldsymbol{a}_1, ..., \boldsymbol{a}_m)$ and $\boldsymbol{B} := \phi(\boldsymbol{b}_1, ..., \boldsymbol{b}_m)$. Sample $\boldsymbol{C} \overset{\$}{\leftarrow} \mathrm{Ker}(\psi)^l$. Send $\boldsymbol{A}, \boldsymbol{B} + \boldsymbol{C}$ to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$.
3. The receiver $P_{\mathcal{R}}$: Upon receiving $\boldsymbol{Z}$ from $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$. Compute $(\boldsymbol{z}_1, ..., \boldsymbol{z}_m) := \psi(\boldsymbol{Z})$. Output $\boldsymbol{z}_1, ..., \boldsymbol{z}_l$.

Fig. 2: A reusable malicious secure VOLE construction over $\mathbb{Z}_{2^k}$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model

Theorem 1. *The protocol $\mathit{\Pi}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$ realizes $\mathcal{F}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$ with reusable malicious security in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model.*

Proof. We first consider the situation that $P_{\mathcal{S}}$ is corrupted and then turn to the situation that $P_{\mathcal{R}}$ is corrupted. Messages with a hat are from the simulator and messages with a prime are from the adversary.

If $P_{\mathcal{S}}$ is corrupted. When the simulator $Sim_{\mathcal{S}}$ extracts the messages $\boldsymbol{A}', \boldsymbol{B}' + \boldsymbol{C}'$ sent to the ideal functionality $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ by the adversary $\mathcal{A}$, he computes

$\hat{\boldsymbol{a}}_1, ..., \hat{\boldsymbol{a}}_m \in \mathbb{Z}_{2^k}^l$ *such that for any* $\boldsymbol{\alpha} \in \mathbb{Z}_{2^k}^m$, $\psi(\boldsymbol{A'} \cdot \phi(\boldsymbol{\alpha})) = (\hat{\boldsymbol{a}}_1 \cdot \alpha_1, ..., \hat{\boldsymbol{a}}_m \cdot \alpha_m) + \boldsymbol{\delta}$, *where the i-th row of* $\boldsymbol{\delta}$ *satisfies the distribution* $\mathcal{D}_{A'_i}^{\mathrm{T}}$*, and picks* $(\hat{\boldsymbol{b}}_1, ..., \hat{\boldsymbol{b}}_m) \leftarrow \psi(\boldsymbol{B'} + \boldsymbol{C'}) + \mathcal{D}_{\boldsymbol{A'}}$[4]*.* $S_{\mathcal{S}}$ *sends* $\hat{\boldsymbol{a}}_i$, $\hat{\boldsymbol{b}}_i$ *for* $i \in [m]$ *to the ideal functionality* $\mathcal{F}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$*. The indistinguishability comes from that* $\psi(\boldsymbol{A'} \cdot \Delta + \boldsymbol{B'} + \boldsymbol{C'})$ *in the real world and* $(\hat{\boldsymbol{a}}_1 \cdot \alpha_1 + \hat{\boldsymbol{b}}_1, ..., \hat{\boldsymbol{a}}_m \cdot \alpha_1 + \hat{\boldsymbol{b}}_m)$ *in the ideal world are statistically-close by the definition of* $(m, d; 2)$*-NM-RMFE.*

If $P_{\mathcal{R}}$ *is corrupted. When* $Sim_{\mathcal{R}}$ *extracts the message* Δ' *sent to the ideal functionality* $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ *by the adversary* $\mathcal{A}$*, he computes a* $\hat{\boldsymbol{\alpha}}$ *such that for any* $\boldsymbol{a}_1, ..., \boldsymbol{a}_m \in \mathbb{Z}_{2^k}^l$, $\psi(\phi(\boldsymbol{a}_1, ..., \boldsymbol{a}_m) \cdot \Delta') = (\boldsymbol{a}_1 \cdot \hat{\alpha}_1, ..., \boldsymbol{a}_m \cdot \hat{\alpha}_m) + \boldsymbol{\delta}$*, where* $\boldsymbol{\delta}$ *satisfies the distribution* $(\mathcal{D}_{\Delta'}^{\mathrm{T}})^l$*.* $Sim_{\mathcal{R}}$ *sends* $\hat{\boldsymbol{\alpha}}$ *to the ideal functionality* $\mathcal{F}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$*. Upon receiving* $\boldsymbol{z}_1, ..., \boldsymbol{z}_m \in \mathbb{Z}_{2^k}^l$ *from* $\mathcal{F}_{\mathrm{VOLE}}^{\mathbb{Z}_{2^k}}$*,* $Sim_{\mathcal{R}}$ *picks* $(\hat{\boldsymbol{d}}_1, ..., \hat{\boldsymbol{d}}_m) \leftarrow (\mathcal{D}_{\Delta'}^{\mathrm{T}})^l$, $\hat{\boldsymbol{C}} \xleftarrow{\$} \mathrm{Ker}(\psi)^l$*, and computes* $\hat{\boldsymbol{Z}} := \phi(\boldsymbol{z}_1 + \hat{\boldsymbol{d}}_1, ..., \boldsymbol{z}_m + \hat{\boldsymbol{d}}_m) + \hat{\boldsymbol{C}}$*. Then* $Sim_{\mathcal{R}}$ *sends* $\hat{\boldsymbol{Z}}$ *to* $\mathcal{A}$*. The adversary* $\mathcal{A}$ *receives* $\boldsymbol{Z} = \phi(\boldsymbol{a}_1, ..., \boldsymbol{a}_m) \cdot \Delta' + \phi(\boldsymbol{b}_1, ..., \boldsymbol{b}_m) + \boldsymbol{C}$ *in the real world, where* $\psi(\boldsymbol{Z})$ *and* $\psi(\hat{\boldsymbol{Z}})$ *are statistically-close by the definition of* $(m, d; 2)$*-NM-RMFE. Further, as* $\boldsymbol{Z}$*'s projection on* $\mathrm{Ker}(\psi)$ *is perfectly masked by* $\boldsymbol{C}$*,* $\mathcal{A}$ *can not distinguish* $\boldsymbol{Z}$ *and* $\hat{\boldsymbol{Z}}$ *as well. Thus, we conclude the proof.* □

3.2 Constructing NM-RMFE

In this section, we present an asymptotically good instantiation, that realizes a slightly weaker variant of NM-RMFE, where the Property 2 in Definition 4 holds for any $\boldsymbol{x}_1, ..., \boldsymbol{x}_D \in (\mathtt{GR}(p^k, r)^*)^m$. We argue that this weaker variant is as good as the standard one when applied in our NISC/VOLE protocol later in Section 4.

For convenience and w.l.o.g., we construct NM-RMFE over Galois fields. In a high level, our construction consists of two layers of RMFEs, one is a standard RMFE, and the other is a so-called Extended RMFE. We define degree-D Extended RMFE as follows:

Definition 5 (Degree-D Extended RMFE). *Let* $\mathbb{F}_q$ *be a finite field of* q *elements,* $n > d > m \geq 1$ *and* $D \geq 1$ *be integers. A pair of maps* (ϕ, ψ) *is called an* $(m, d, n; D)_q$*-Extended RMFE if* $\phi : \mathbb{F}_q^m \times \mathbb{F}_{q^d} \to \mathbb{F}_{q^n}$ *and* $\psi : \mathbb{F}_{q^n} \to \mathbb{F}_q^m \times \mathbb{F}_{q^d}$ *are two* $\mathbb{F}_q$*-linear maps satisfying*

$$\psi(\phi(\boldsymbol{x_1}, y_1) \cdot \phi(\boldsymbol{x_2}, y_2) \cdots \phi(\boldsymbol{x_D}, y_D)) = (\boldsymbol{x_1} * \boldsymbol{x_2} * \cdots * \boldsymbol{x}_D, y_1 y_2 \cdots y_D),$$

for any $\boldsymbol{x}_i \in \mathbb{F}_q^m$*,* $\boldsymbol{y}_i \in \mathbb{F}_{q^d}$*,* $i \in [D]$*.*

The degree-D Extended RMFE is a natural extension of degree-D RMFEs, and the construction is straightforward. Thus it is omitted here.

Let (ϕ_1, ψ_1) be an $(m+k, d; D)_q$-RMFE, and (ϕ_2, ψ_2) be an $(m+k, d, n; D)_q$-extended RMFE. We construct an $(m, n; D)_q$-NM-RMFE (ϕ, ψ) as follows.

- $\phi : \mathbb{F}_q^m \to \mathbb{F}_{q^n}$ is an $\mathbb{F}_q$-linear map, such that $\phi : \boldsymbol{x} \mapsto \phi_2(\boldsymbol{x}, \boldsymbol{r}, \phi_1(\boldsymbol{x}, \boldsymbol{r}))$, where $\boldsymbol{r} \xleftarrow{\$} \mathbb{F}_q^k$.

[4] We define $\mathcal{D}_{\boldsymbol{A'}} := (\mathcal{D}_{A'_1}, ..., \mathcal{D}_{A'_l})^{\mathrm{T}}$.

– For a $Y \in \mathbb{F}_{q^n}$, let $(\boldsymbol{y}, \boldsymbol{s}, e) := \psi_2(Y)$, where $\boldsymbol{y} \in \mathbb{F}_q^m$, $\boldsymbol{s} \in \mathbb{F}_q^k$ and $e \in \mathbb{F}_{q^d}$. $\psi : \mathbb{F}_{q^n} \to \mathbb{F}_q^m$ is defined as follows:

$$\psi(Y) = \begin{cases} \boldsymbol{y}, \; if \;\; \psi_1(e) = (\boldsymbol{y}, \boldsymbol{s}) \;, \\ \perp, \; otherwise. \end{cases}$$

W.l.o.g. and for simplicity, we take $D = 2$, and assume $\phi_1(\mathbf{1}) = 1$ and $\phi_2(\mathbf{1}, 1) = 1$. Let $\mathcal{V}_\perp$ denote the set $\{\phi_2(\mathbf{0}, \mathbf{0}, \phi_1(\boldsymbol{x}, \boldsymbol{r})) \mid \boldsymbol{x} \in \mathbb{F}_q^m, \boldsymbol{r} \in \mathbb{F}_q^k\}$. We have the following observations (which can be naturally extended to $D > 2$ cases).

Proposition 1. *Let (ϕ, ψ) be defined as above, there exists q^{n-d} solutions for $Y \in \mathbb{F}_{q^n}$, such that $\psi(Y) \neq \perp$.*

Proof. Assume there exist $\boldsymbol{z} \in \mathbb{F}_q^m, \boldsymbol{t} \in \mathbb{F}_q^k$ satisfying $\psi_2(\phi(Y) = (\boldsymbol{z}, \boldsymbol{t}, \phi_1(\boldsymbol{z}, \boldsymbol{t}))$. Then, we have that

$$Y \in \psi_2^{-1}(\{(\boldsymbol{z}, \boldsymbol{t}, \phi_1(\boldsymbol{z}, \boldsymbol{t})) \mid \boldsymbol{z} \in \mathbb{F}_q^m, \boldsymbol{t} \in \mathbb{F}_q^k\}).$$

By Lemma 2, we have that $\psi_2^{-1}(\{(\boldsymbol{z}, \boldsymbol{t}, \phi_1(\boldsymbol{z}, \boldsymbol{t})) \mid \boldsymbol{z} \in \mathbb{F}_q^m, \boldsymbol{t} \in \mathbb{F}_q^k\}) = \mathrm{Ker}(\psi_2) \oplus \mathrm{Im}(\phi)$. Since $|\mathrm{Ker}(\psi_2)| = \frac{q^n}{q^{m+k} \cdot q^d}$ and $|\mathrm{Im}(\phi)| = q^{m+k}$, there are q^{n-d} solutions for Y such that $\psi(Y) \neq \perp$. □

Proposition 2. *Let (ϕ, ψ) be defined as above, then $\mathbb{F}_{q^n} = \mathrm{Im}(\phi) \oplus \mathcal{V}_\perp \oplus \mathrm{Ker}(\psi)$.*

Proof. We show that for any $Y \in \mathbb{F}_{q^n}$, Y can be uniquely written as the additions of the projections on the above sets, respectively. Define $\tau_1 := \phi_1 \circ \psi_1$. Assume $(\boldsymbol{y}, \boldsymbol{s}, e) := \psi_2(Y)$. Let $A := \phi_2(\boldsymbol{y}, \boldsymbol{s}, \phi_1(\boldsymbol{y}, \boldsymbol{s}))$ and $B := \phi_2(\mathbf{0}, \mathbf{0}, \tau_1(e) - \phi_1(\boldsymbol{y}, \boldsymbol{s}))$. By definition, we have that $A \in \mathrm{Im}(\phi)$ and $B \in \mathcal{V}_\perp$. It can be verified that $\psi_2(Y - A - B) = (\mathbf{0}, \mathbf{0}, e - \tau_1(e))$. Thus, $(Y - A - B) \in \mathrm{Ker}(\psi)$ and we complete the proof. □

As the adversary can carefully select a $Y \notin \mathrm{Im}(\phi)$, we need to find the distribution $\mathcal{D}_Y$ for each Y. From now on we consider the specific polynomial-based construction of RMFE[5], which allows us to provide an explicit description of $\mathcal{D}_Y$ [6].

Let $\alpha_1, \alpha_2, ..., \alpha_m$ and $\beta_1, \beta_2, ..., \beta_k$ be $m + k$ pair-wise distinct elements in $\mathbb{F}_q$. There exists a unique polynomial $f \in \mathbb{F}_q[x]$ with $\deg(f) \leq m + k - 1$, such that $f(\alpha_i) = x_i, i \in [m]$ and $f(\beta_j) = r_j, j \in [k]$. Let $d = 2(m + k) - 1$ and $n \geq 2(m + k + d) - 1$. There exist a degree-d irreducible polynomial $p \in \mathbb{F}_q[x]$ and a degree-n irreducible polynomial $g \in \mathbb{F}_q[x]$ such that $\mathbb{F}_{q^d} \cong \mathbb{F}_q[x]/(p)$ and $\mathbb{F}_{q^n} \cong \mathbb{F}_q[x]/(g)$. Therefore, elements in the extension field can be viewed as polynomials. In particular, we pick $g(x) = x^n + ax + b$, where $a, b \in \mathbb{F}_q$ (such irreducible $g(x)$ exists, if q is a prime and $(n, d) = 1$, see [22]). Let Y be a polynomial over $\mathbb{F}_q$ with degree $\leq n - 1$.

[5] In the RMFE literature [7,12,14], known RMFEs are constructed from algebraic geometry curves. Polynomials are essentially genus-0 curves and in most cases, RMFEs constructed from genus-0 curves have a lower ratio $\frac{d}{m}$.

[6] In fact, similar results can be obtained for general constructions of RMFE, also for RMFEs over Galois rings.

- For $\boldsymbol{x} \in \mathbb{F}_q^m$, ϕ is defined as

$$\phi : \boldsymbol{x} \mapsto f,\ where\ f \xleftarrow{\$} \mathbb{F}_q[x]_{\leq m+k-1},\ satisfying\ f(\alpha_i) = x_i, i \in [m].$$

- For $f \in \mathbb{F}_q[x]_{\leq n-1}$, let $\hat{f} := f \mod p$. ψ is defined as

$$\psi : f \mapsto \begin{cases} (f(\alpha_1), ..., f(\alpha_m)),\ if\ \hat{f}(\alpha_i) = f(\alpha_i), \hat{f}(\beta_j) = f(\beta_j), i \in [m], j \in [k], \\ \perp,\ otherwise. \end{cases}$$

First we give the following lemma.

Lemma 5. *Given a $\boldsymbol{x} \in \mathbb{F}_q^m$, $\alpha_1, ..., \alpha_m$ and $\beta_1, ..., \beta_k$ are $m+k$ pair-wise distinct elements in $\mathbb{F}_q$, then there are q^i solutions of $\boldsymbol{r} \in \mathbb{F}_q^k$, such that $\boldsymbol{x}, \boldsymbol{r}$ interpolate a polynomial f with degree $\leq m-1+i$, $i = 0, 1, ..., k$.*

Proof. Since the evaluation map $\sigma : f \mapsto (f(\alpha_1), ..., f(\alpha_m), f(\beta_1), ..., f(\beta_k))$ induces a bijection from $\mathbb{F}_q[x]_{\leq m+k-1} := \{f \in \mathbb{F}_q[x] \mid \deg(f) \leq m+k-1\}$ to $\mathbb{F}_q^m \times \mathbb{F}_q^k$, there are at most q^{m+i} solutions of $\boldsymbol{x}, \boldsymbol{r}$ such that $\boldsymbol{x}, \boldsymbol{r}$ interpolate a polynomial f with degree $\leq m-1+i$, $i = 0, 1, ..., k$. On the other hand, for a given $\boldsymbol{x} \in \mathbb{F}_q^m$, let the first i positions of $\boldsymbol{r}$ be random, thus $\boldsymbol{x}$ along with $\boldsymbol{r}|_{[i]}$ interpolate a polynomial f with degree $\leq m-1+i$. Set the remaining $k-i$ positions of $\boldsymbol{r}$ lie on f. Thus, there are at least q^i solutions of $\boldsymbol{r} \in \mathbb{F}_q^k$ for a given $\boldsymbol{x}$. Combining together, we conclude the proof. □

We have the following theorem.

Theorem 2. *Let ϕ, ψ be defined above. For any $Y \in \mathbb{F}_{q^n}$, $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$ and sufficiently large k, there exists a distribution $\mathcal{D}_Y$ such that $\mathcal{D}_{\boldsymbol{x},Y} \overset{s}{\approx} \mathcal{D}_Y$.*

Proof. Let us consider the degree of Y. We remark that the result holds for any $\boldsymbol{x} \in \mathbb{F}_q^m$ if not pointed out explicitly.

i. If $\deg(Y) \leq m+k-1$. We have that

$$\begin{aligned} \psi(\phi(\boldsymbol{x}) \cdot Y) &= \psi(f \cdot Y) = (f \cdot Y(\alpha_1), ..., f \cdot Y(\alpha_m)) \\ &= (f(\alpha_1), ..., f(\alpha_m)) * (Y(\alpha_1), ..., Y(\alpha_m)) \\ &= \boldsymbol{x} * \psi(Y), \end{aligned}$$

as $\deg(f \cdot Y) \leq d-1$. So in this condition, $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}_Y : \Pr[\boldsymbol{\delta} = \boldsymbol{0}] = 1$.

ii. If $m+k-1 < \deg(Y) \leq m+2k-1$. Since $\deg(\phi(\boldsymbol{x} \cdot Y)) \leq d+k-1$, $\psi(\phi(\boldsymbol{x}) \cdot Y) = \perp$ if $\deg(\phi(\boldsymbol{x}) \cdot Y) \geq d$. On the other hand, the equation $\psi(\phi(\boldsymbol{x}) \cdot Y) = \boldsymbol{x} \star \psi(Y)$ holds if $\deg(\phi(\boldsymbol{x}) \cdot Y) \leq d-1$. By Lemma 5 , we have that for all possible values of $\boldsymbol{x}$,

$$\mathcal{D}_{\boldsymbol{x},Y} : \begin{cases} \Pr[\boldsymbol{\delta} = \boldsymbol{0}] = 1/q^{\deg(Y)-m-k+1}, \\ \Pr[\boldsymbol{\delta} = \perp] = 1 - 1/q^{\deg(Y)-m-k+1}. \end{cases}$$

So in this condition, $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}_Y$.

iii. *If* $m + 2k - 1 < \deg(Y) \leq 2(m + k - 1)$. *Similarly, since* $\deg(\phi(\boldsymbol{x} \cdot Y)) \leq d + m + k - 2$, *we have that* $\psi(\phi(\boldsymbol{x}) \cdot Y) = \perp$ *if* $\deg(\phi(\boldsymbol{x}) \cdot Y) \geq d$, *and* $\psi(\phi(\boldsymbol{x}) \cdot Y) = \boldsymbol{x} \star \psi(Y)$ *holds if* $\deg(\phi(\boldsymbol{x}) \cdot Y) \leq d - 1$. *There are only* $q^{2(m+k)-1-\deg(Y)}$ *possible values of* $\boldsymbol{x}$ *such that* $\boldsymbol{x}$ *interpolates a polynomial* f *with degree* $\leq 2(m+k-1) - \deg(Y)$, *as there are exactly* $q^{2(m+k)-1-\deg(Y)}$ *choices of such* f. *For these* $\boldsymbol{x}$, *let* $\boldsymbol{r}$ *lie on* f, *and we have that*

$$\mathcal{D}_{\boldsymbol{x},Y} : \begin{cases} \Pr[\boldsymbol{\delta} = \boldsymbol{0}] = 1/q^k, \\ \Pr[\boldsymbol{\delta} = \perp] = 1 - 1/q^k. \end{cases}$$

For the remaining $q^m - q^{2(m+k)-1-\deg(Y)}$ *possible values of* $\boldsymbol{x}$, *there are no solutions for* $r \in \mathbb{F}_q^k$, *and we have* $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}$. *So in this condition,* $\mathcal{D}_{\boldsymbol{x},Y} \overset{s}{\approx} \mathcal{D}$ *for sufficient large* k.

iv. *If* $2(m + k) - 1 \leq \deg(Y) \leq n - m - k$. *Let* $\hat{Y} := Y \mod p$, *and we can find* $r \in \mathbb{F}_q[x]$ *satisfying* $Y = \hat{Y} + p \cdot r$. *Let* $a(x) := \prod_{i=1}^{m}(x - \alpha_i)$, *and* $b(x) := \prod_{i=1}^{k}(x - \beta_i)$. *Since* $\deg(\phi(\boldsymbol{x} \cdot Y)) \leq n - 1$, *we have that if* $\deg(\hat{Y}) \leq m + k - 1$, $a(x) \mid r(x)$, *and* $r(\beta_j) = 0$ *for* $j \in J \subseteq [k]$,

$$\mathcal{D}_{\boldsymbol{x},Y} : \begin{cases} \Pr[\boldsymbol{\delta} = \boldsymbol{0}] = 1/q^{k-|J|}, \\ \Pr[\boldsymbol{\delta} = \perp] = 1 - 1/q^{k-|J|}. \end{cases}$$

However, we remark that if $\deg(\hat{Y}) \leq m + k - 1$, $b(x) \mid r(x)$ *and* $a(x) \nmid r(x)$, *we have that* $\psi(\phi(\boldsymbol{0}) \cdot Y) = \boldsymbol{0}$ *but* $\psi(\phi(\boldsymbol{x}) \cdot Y) = \perp$, *for* $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$. *Thus for the remaining choices of* Y, $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}$ *holds for* $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$. *So, in this condition,* $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}_Y$ *for* $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$.

v. *If* $\deg(Y) \geq n - m - k + 1$. *If* $\deg(\phi(\boldsymbol{x}) \cdot Y) \leq n - 1$, *the discussion is similar to the previous one, and we have that* $\mathcal{D}_{\boldsymbol{x},Y} = \mathcal{D}_Y$ *for* $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$. *If* $\deg(\phi(\boldsymbol{x}) \cdot Y) \geq n$, $\deg(\phi(\boldsymbol{x}) \cdot Y \mod g)$ *will exceed* $d - 1$ *and lead to* $\perp$ *overwhelmingly, as we take* $g(x)$ *of the form* $g(x) = x^n + ax + b$, $a, b \in \mathbb{F}_q$. *So in this condition,* $\mathcal{D}_{\boldsymbol{x},Y} \overset{s}{\approx} \mathcal{D}_Y$ *for* $\boldsymbol{x} \in (\mathbb{F}_q^*)^m$.

From above discussions, we conclude the proof. □

As the above NM-RMFE construction contains two layers of RMFEs, we remark that the asymptotic behavior of NM-RMFE is not as good as RMFE (though still constant). For instance, by Lemma 3, there exists a family of $(m, d; 2)$-NM-RMFEs over $\mathbb{F}_2(\mathbb{Z}_{2^{32}}, \mathbb{Z}_{2^{64}})$ with $m \to \infty$ and $\frac{d}{m} \to 29.13$ and a family of $(m, d; 3)$-NM-RMFEs over $\mathbb{F}_2(\mathbb{Z}_{2^{32}}, \mathbb{Z}_{2^{64}})$ with $m \to \infty$ and $\frac{d}{m} \to 80.15$. For the concrete efficiency of $(m, d; 3)$-NM-RMFEs over $\mathbb{Z}_{2^{32}}$, according to the results in [14], there exists a $(3m, 7(3m + 4); 3)$-RMFE over $\mathbb{Z}_{2^{32}}$ for any $m \leq 150$. We obtain that the NM-RMFE ratio $\frac{d}{m+k}$ is 56 approximately, where k is related to the statistical security parameter κ. Assume $\kappa = 80$, by setting $m = 150$, the ratio d/m is 87.3. Note that for a given κ, the ratio d/m is close to a constant (56 in this case), as long as m is relatively large compared to k.

4 Amortized rNISC/VOLE

In this section, we first show how to construct a semi-honest secure NISC/VOLE for computing branching programs over $\mathbb{Z}_{2^k}$. Then we show how to obtain a reusable malicious secure one.

4.1 Amortized NISC/VOLE with semi-honest security

Let $L(\boldsymbol{x})$ be a matrix induced by a size $(s+1)$ BP over $\mathbb{Z}_{2^k}$ computing $f : \mathbb{Z}_{2^k}^n \to \mathbb{Z}_{2^k}$, i.e. $det(L(\boldsymbol{x})) = f(\boldsymbol{x})$. Suppose $\boldsymbol{x}_1, ..., \boldsymbol{x}_m \in \mathbb{Z}_{2^k}^n$. We consider the case of computing $f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_m)$ in parallel[7], where we can amortize the cost by using RMFEs. In a high level, we present a variant of DARE for BP over $\texttt{GR}(2^k, d)$, which, in effect, computes parallel DAREs for BP over $\mathbb{Z}_{2^k}$ (i.e. reveals nothing but $f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_m)$).

Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$, and $\tau \coloneqq \phi \circ \psi$. We generalize these maps to perform on matrices in a natural way. As ϕ is a $\mathbb{Z}_{2^k}$-linear map, it can be observed that $\phi(L(\boldsymbol{x}_1), ..., L(\boldsymbol{x}_m)) = L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m))$. We define following matrix sets over $\texttt{GR}(2^k, d)$:

$$\hat{\mathcal{H}}^s \coloneqq \{\phi(H_1, ..., H_m) \mid H_i \in \mathcal{H}^s, i \in [m]\},$$
$$\hat{\mathcal{G}}_1^s \coloneqq \{\phi(G_1, ..., G_m) \mid G_i \in \mathcal{G}_1^s, i \in [m]\},$$
$$\hat{\mathcal{G}}_2^s \coloneqq \{\phi(G_1, ..., G_m) \mid G_i \in \mathcal{G}_2^s, i \in [m]\},$$

where $\mathcal{H}^s, \mathcal{G}_1^s, \mathcal{G}_2^s$ are defined in Corollary 1.

We observe that the encoding of $L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m))$ (i.e. $R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m)) R_2$, where R_1, R_2 are sampled uniformly at random from $\hat{\mathcal{G}}_1^s, \hat{\mathcal{G}}_2^s$, respectively) reveals not only $det(L(\boldsymbol{x}_1)), ..., det(L(\boldsymbol{x}_m))$ but also $det(L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m)))$, which we do not desire. To solve this issue, we mask the encoding by random values over $\mathrm{Ker}(\psi)$. Therefore, we define a matrix group $\mathcal{I}^s$ for this purpose.

Definition 6. *Let $\mathcal{I}^s$ be the set of all $s \times s$ matrices over* $\mathrm{Ker}(\psi)$ *with* 0*'s below the main diagonal.*

We have the following proposition, which indicates that parallel DAREs over $\mathbb{Z}_{2^k}$ can be implemented at one time via RMFE.

Proposition 3. *Let R_1, R_2, R_3 be uniformly and independently distributed matrices from $\hat{\mathcal{G}}_1^s, \hat{\mathcal{G}}_2^s, \mathcal{I}^s$, respectively. We have that $M \coloneqq R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m)) R_2 + R_3$ reveals no information about $L(\boldsymbol{x}_1), ..., L(\boldsymbol{x}_m)$ but $det(L(\boldsymbol{x}_1)), ..., det(L(\boldsymbol{x}_m))$.*

Proof. The map $\psi : \texttt{GR}(2^k, d) \to \mathbb{Z}_{2^k}^m$ induces m $\mathbb{Z}_{2^k}$-linear maps $\psi_i : \texttt{GR}(2^k, d) \to \mathbb{Z}_{2^k}$, for $i \in [m]$, i.e. $(\psi_1, ..., \psi_m) \coloneqq \psi$. By the definition of $(m, d; 3)$-RMFEs, we have that $\psi_i(R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m)) R_2 + R_3) = \psi_i(R_1) L(\boldsymbol{x}_i) \psi_i(R_2)$, for $i \in [m]$. Since R_1, R_2 are uniformly and independently distributed from $\hat{\mathcal{G}}_1^s, \hat{\mathcal{G}}_2^s$, and ψ

[7] It remains interesting and open whether a branching program can be transformed into copies of a sub branching program.

conditioned on $\mathrm{Im}(\phi)$ *is a bijection,* $\psi_i(R_1), \psi_i(R_2)$ *are uniformly and independently distributed from* $\mathcal{G}_1^s, \mathcal{G}_2^s$ *for* $i \in [m]$*, respectively. Thus, by Corollary 1,* $\psi_i(R_1)L(\boldsymbol{x}_i)\psi_i(R_2)$ *reveals nothing about* $L(\boldsymbol{x}_i)$ *but* $det(L(\boldsymbol{x}_i))$*, for* $i \in [m]$*. Finally, we claim that* $R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m))R_2 + R_3$ *reveals no more information than* $\psi(R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m))R_2 + R_3)$*. Since, by Lemma 2, we have that* $\mathtt{GR}(2^k, d) = \mathrm{Im}(\phi) \oplus \mathrm{Ker}(\psi)$ *(assuming* $\phi(\mathbf{1}) = 1$*) and* $R_1 L(\phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m))R_2$*'s projection on* $\mathrm{Ker}(\psi)$ *is perfectly masked by* R_3*. This completes the proof.* □

Given the above proposition, now we proceed to construct our semi-honest NISC protocol over $\mathbb{Z}_{2^k}$. We consider a slightly more general framework with t branching programs BP_i of size (s_i+1) over $\mathbb{Z}_{2^k}$, computing $f_i : \mathbb{Z}_{2^k}^{n_1} \times \mathbb{Z}_{2^k}^{n_2} \to \mathbb{Z}_{2^k}$, for $i \in [t]$. Let $f(\boldsymbol{x}, \boldsymbol{y})$ be a two-party sender-receiver functionality, taking inputs $\boldsymbol{x} \in \mathbb{Z}_{2^k}^{n_1}, \boldsymbol{y} \in \mathbb{Z}_{2^k}^{n_2}$ from the receiver $P_{\mathcal{R}}$ and the sender $P_{\mathcal{S}}$, respectively, and sends $f(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}_{2^k}^t$ to $P_{\mathcal{R}}$, where $f := (f_1, ..., f_t)$. Let $L_i(\boldsymbol{x}, \boldsymbol{y})$ be the $s_i \times s_i$ matrix induced by BP_i, for $i \in [t]$. Suppose $f(\boldsymbol{x}, \boldsymbol{y})$ will be invoked m times, with inputs $(\boldsymbol{x}_1, \boldsymbol{y}_1), ..., (\boldsymbol{x}_m, \boldsymbol{y}_m) \in \mathbb{Z}_{2^k}^{n_1} \times \mathbb{Z}_{2^k}^{n_2}$, respectively. We present the amortized NISC protocol in Figure 3 and we have the following theorem.

Protocol $\mathit{\Pi}_{\mathrm{NISC}}$

The function $f : \mathbb{Z}_{2^k}^{n_1} \times \mathbb{Z}_{2^k}^{n_2} \to \mathbb{Z}_{2^k}^t$ is described as above. Suppose $P_{\mathcal{R}}$ has input $\boldsymbol{x}_1, ..., \boldsymbol{x}_m \in \mathbb{Z}_{2^k}^{n_1}$, and $P_{\mathcal{S}}$ has input $\boldsymbol{y}_1, ..., \boldsymbol{y}_m \in \mathbb{Z}_{2^k}^{n_2}$. Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$.

1. The receiver $P_{\mathcal{R}}$ computes $\boldsymbol{X} := \phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m)$. For $j \in [n_1]$, $P_{\mathcal{R}}$ sends $(j; X_j)$ to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$.
2. The sender $P_{\mathcal{S}}$ computes $\boldsymbol{Y} := \phi(\boldsymbol{y}_1, ..., \boldsymbol{y}_m)$. For $i \in [t]$, $P_{\mathcal{S}}$ computes $M_i(\cdot) := R_{1,i} L_i(\cdot, \boldsymbol{Y}) R_{2,i} + R_{3,i}$, where $R_{1,i} \xleftarrow{\$} \hat{\mathcal{G}}_1^{s_i}, R_{2,i} \xleftarrow{\$} \hat{\mathcal{G}}_2^{s_i}, R_{3,i} \xleftarrow{\$} \mathcal{I}^{s_i}$. Since each entry of $M_i(\boldsymbol{X})$ is a linear polynomial on variables $X_1, ..., X_{n_1}$, $P_{\mathcal{S}}$ sends messages to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ according to $M_i(\cdot)$, for $i \in [t]$.
3. For $i \in [t]$, $P_{\mathcal{R}}$ obtains $M_i(\boldsymbol{X})$ from $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$, and then computes $(L_i(\boldsymbol{x}_1, \boldsymbol{y}_1), ..., L_i(\boldsymbol{x}_m, \boldsymbol{y}_m)) := \psi(M_i(\boldsymbol{X}))$. For $j \in [m]$, $P_{\mathcal{R}}$ computes and outputs $f(\boldsymbol{x}_j, \boldsymbol{y}_j) := (det(L_1(\boldsymbol{x}_j, \boldsymbol{y}_j)), ..., det(L_t(\boldsymbol{x}_j, \boldsymbol{y}_j)))$.

Fig. 3: Protocol for semi-honest NISC over $\mathbb{Z}_{2^k}$ in the (chosen-input) $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model.

Theorem 3 (Semi-honest NISC/VOLE over $\mathbb{Z}_{2^k}$**).** *Protocol* $\mathit{\Pi}_{\mathrm{NISC}}$ *realizes a two-party sender-receiver functionality that computes* $f : \mathbb{Z}_{2^k}^{n_1} \times \mathbb{Z}_{2^k}^{n_2} \to \mathbb{Z}_{2^k}^t$ *with semi-honest security in the* $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$*-hybrid model. In particular,* $\mathit{\Pi}_{\mathrm{NISC}}$ *invokes* n_1 *instances of VOLE, and the length of the* j*-th VOLE instance is* $S_j := \sum_{i \in D(j)} \binom{s_i}{2}$*, where* $D(j)$ *is the set of output entries that depend on* X_j*.*

Proof. *If $P_{\mathcal{R}}$ is corrupted, the simulator $Sim_{\mathcal{R}}$ receives $X_1, ..., X_{n_1}$ from the adversary $\mathcal{A}$. Then, $Sim_{\mathcal{R}}$ computes $(\boldsymbol{x}_1, ..., \boldsymbol{x}_m) := \psi(\boldsymbol{X})$, and sends them to the ideal functionality that computes f. For $j \in [m]$, $Sim_{\mathcal{R}}$ receives $\boldsymbol{z}_j \in \mathbb{Z}_{2^k}^t$ from the ideal functionality, and for $i \in [t]$, $Sim_{\mathcal{R}}$ samples random matrix $L_{j,i}$ over $\mathbb{Z}_{2^k}$ with -1's in the second diagonal and 0's below the second diagonal such that $det(L_{j,i}) = z_{j,i}$. For $i \in [t]$, $Sim_{\mathcal{R}}$ samples $R_{3,i} \xleftarrow{\$} \mathcal{I}^{s_i}$ and computes $M_i := \phi(L_{1,i}, ..., L_{m,i}) + R_{3,i}$. Finally, $Sim_{\mathcal{R}}$ delivers $M_1, ..., M_t$ to $\mathcal{A}$ emulated as VOLE outputs (n_1 instances of VOLE, with total length $\sum_{j=1}^{n_1} S_j$). We remark that this procedure can be done without the knowledge of $\boldsymbol{Y}$, since the function f is public. The correctness is easy to verify and the indistinguishability is directly obtained by Proposition 3.*

If $P_{\mathcal{S}}$ is corrupted, the simulator $Sim_{\mathcal{S}}$ receives VOLE inputs from the adversary $\mathcal{A}$, which conveys the matrices $M_1(\cdot), ..., M_t(\cdot)$ over $\mathtt{GR}(2^k, d)$. For $i \in [t]$, $Sim_{\mathcal{S}}$ computes $(M_{1,i}(\cdot), ..., M_{m,i}(\cdot)) := \psi(M_i(\cdot))$. Recall that for each $M_{j,i}(\cdot)$, there exist $R_{1,j,i} \in \mathcal{G}_1^{s_i}$ and $R_{2,j,i} \in \mathcal{G}_2^{s_i}$ such that $R_{1,j,i} L_i(\cdot, \boldsymbol{y}_j) R_{2,j,i} = M_{j,i}(\cdot)$. It can be observed that each entry of $R_{1,j,i}, R_{2,j,i}$ can be computed from the VOLE messages. (This depends crucially on the structure of $R_{1,j,i}, L_i(\cdot, \boldsymbol{y}_j), R_{2,j,i}$; we refer the reader to [18] for more details.) Since $R_{1,j,i}, R_{2,j,i}$ are invertible, $S_{\mathcal{S}}$ can extract $\boldsymbol{y}_j$ for all $j \in [m]$. Finally, $Sim_{\mathcal{S}}$ sends $\boldsymbol{y}_1, ..., \boldsymbol{y}_m$ to the ideal functionality that computes f. The indistinguishability is obtained by the correct extraction of $\boldsymbol{y}_1, ..., \boldsymbol{y}_m$. This completes the proof. □

4.2 Amortized rNISC/VOLE with malicious security

We first consider an intermediate security model where both the malicious sender and the malicious receiver follow the RMFE part specifications [8] (e.g., computes $X := \phi(\boldsymbol{x}_1, ..., \boldsymbol{x}_m), Y := \phi(\boldsymbol{y}_1, ..., \boldsymbol{y}_m)$) and we only need to enforce the malicious sender's compliance with the DARE part specifications (e.g., sends messages to the VOLE functionality according to $M_i(\cdot)$). Then, we consider the full malicious security model and show how to construct maliciously secure rNISC/VOLE for computing branching programs over $\mathbb{Z}_{2^k}$.

We generalize the certified VOLE (cVOLE) method (for Galois fields) [13] to Galois ring analogue as the first step. The cVOLE is a special case of NISC/VOLE, where the sender's inputs sent to multiple instances of VOLE need to satisfy some arithmetic constraints (formulated by a circuit $\mathcal{C}$), which allows for forcing the malicious sender to follow the (Π_{NISC}) protocol specifications honestly (see Figure 4). Similar to its Galois field counterpart [13], constructing a cVOLE protocol over Galois rings involves two main ingredients, a certified VOLE with equality constraint (eVOLE for short) over Galois rings and a statistical NIZK protocol for proving circuit satisfiability over Galois rings.

eVOLE. The eVOLE is a weak variant of cVOLE, which only restricts some given positions of the sender's inputs to multiple instances of VOLE to being equal (rather than satisfying a general arithmetic constraint). We formalize the

[8] In fact, the malicious receiver can only cheat by deviating from the RMFE encoding. Namely, we assume a semi-honest receiver.

Functionality $\mathcal{F}_{\text{cVOLE}}^{\texttt{GR}(2^k,d)}$

Parameterized by a Galois ring $\texttt{GR}(2^k, d)$, a sequence of n positive integers $l_1, ..., l_n$, for $i \in [n]$, and an arithmetic circuit $\mathcal{C}$ over $\texttt{GR}(2^k, d)$ on $q \leq 2\sum_{i=1}^{n} l_i$ inputs. Suppose $P_\mathcal{S}$ has input $\boldsymbol{a}_i, \boldsymbol{b}_i \in \texttt{GR}(2^k, d)^{l_i}$, and $P_\mathcal{R}$ has input $\alpha_i \in \texttt{GR}(2^k, d)$, for $i \in [n]$.

1. Receive $(\alpha_1, ..., \alpha_n)$ from $P_\mathcal{R}$, and $(\boldsymbol{a}_1, ..., \boldsymbol{a}_n, \boldsymbol{b}_1, ..., \boldsymbol{b}_n)$ from $P_\mathcal{S}$.
2. Verify that $(\boldsymbol{a}_1, ..., \boldsymbol{a}_n, \boldsymbol{b}_1, ..., \boldsymbol{b}_n)$ is a satisfying assignment for circuit $\mathcal{C}$. If the check fails, send $\perp$ to $P_\mathcal{R}$. Otherwise, compute $\boldsymbol{v}_i := \boldsymbol{a}_i \cdot \alpha_i + \boldsymbol{b}_i$ for $i \in [n]$ and send $(\boldsymbol{v}_1, ..., \boldsymbol{v}_n)$ to $P_\mathcal{R}$. If $P_\mathcal{S}$ is corrupted, and receive `aborting` from $\mathcal{S}$, send $\perp$ to $P_\mathcal{R}$.

Fig. 4: Certified VOLE with a general arithmetic constraint

Functionality $\mathcal{F}_{\text{eVOLE}}^{\texttt{GR}(2^k,d)}$

$\mathcal{F}_{\text{eVOLE}}^{\texttt{GR}(2^k,d)}$ extends the (chosen-input) VOLE functionality $\mathcal{F}_{\text{VOLE}}^{\texttt{GR}(2^k,d)}$. **Setup phase**, **Send phases**, and **Deliver phases** are identical to those in $\mathcal{F}_{\text{VOLE}}^{\texttt{GR}(2^k,d)}$, respectively. Parameterized by a Galois ring $\texttt{GR}(2^k, d)$, length parameters $l_1, ..., l_n \in \mathbb{N}$.

Verify phases:

1. Upon receiving $(sid_1, sid_2; \texttt{Verify}\dagger; l_{i_1}, l_{i_2}, j_1, j_2)$ from $P_\mathcal{R}$, where $i_1, i_2 \in [n], j_1 \in [l_{i_1}], j_2 \in [l_{i_2}]$ and sid_1, sid_2 are two session identifiers, verify that there are stored inputs $(sid_1; \boldsymbol{a}_1, \boldsymbol{b}_1; l_{i_1})$ and $(sid_2; \boldsymbol{a}_2, \boldsymbol{b}_2; l_{i_2})$ from $P_\mathcal{S}$; else ignore the message. Then, verify that $a_{1,j_1} = a_{2,j_2}$. If the check fails, send $\perp$ to $P_\mathcal{R}$.
2. Upon receiving $(sid_1, sid_2; \texttt{Verify}\ddagger; l_{i_1}, l_{i_2}, j_1, j_2)$ from $P_\mathcal{R}$, where $i_1, i_2 \in [n], j_1 \in [l_{i_1}], j_2 \in [l_{i_2}]$ and sid_1, sid_2 are two session identifiers, verify that there are stored inputs $(sid_1; \boldsymbol{a}_1, \boldsymbol{b}_1; l_{i_1})$ and $(sid_2; \boldsymbol{a}_2, \boldsymbol{b}_2; l_{i_2})$ from $P_\mathcal{S}$; else ignore the message. Then, verify that $b_{1,j_1} = a_{2,j_2}$. If the check fails, send $\perp$ to $P_\mathcal{R}$.

Fig. 5: Distributional certified VOLE with equality constraints.

eVOLE functionality in Figure 5. The eVOLE construction[9] (presented in Figure 6) shares similarity with the eVOLE construction for Galois fields [13], and is built upon random VOLE. We address the main difference and sketch how to construct eVOLE from chosen-input VOLE for simplicity (the reduction of chosen-input VOLE to random VOLE is straightforward).

Protocol $\Pi_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$

Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, length parameters $l_1, l_2 \in \mathbb{N}$. $P_{\mathcal{S}}$ has inputs $\boldsymbol{a}_t, \boldsymbol{b}_t \in \mathtt{GR}(2^k,d)^{l_t}$, $t \in [2]$. $P_{\mathcal{R}}$ has (random) inputs $\alpha_1, \alpha_2 \in \mathtt{GR}(2^k,d)$.

1. The sender $P_{\mathcal{S}}$ and the receiver $P_{\mathcal{R}}$ invoke the **Setup phase** of $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ with $P_{\mathcal{R}}$'s inputs (α_1, α_2).
2. For $t \in [2]$, $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$ invoke the **Send phases** of $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ with inputs $(t; l_t + 1)$. The sender $P_{\mathcal{S}}$ receives $\hat{\boldsymbol{a}}_t, \hat{\boldsymbol{b}}_t \in \mathtt{GR}(2^k,d)^{l_t+1}$, while $P_{\mathcal{R}}$ receives $\hat{\boldsymbol{v}}_t \in \mathtt{GR}(2^k,d)^{l_t+1}$, such that $\hat{\boldsymbol{v}}_t = \hat{\boldsymbol{a}}_t \cdot \alpha_t + \hat{\boldsymbol{b}}_t$.
3. For $t \in [2]$, $P_{\mathcal{S}}$ sends $\boldsymbol{u}_t := \boldsymbol{a}_t - \hat{\boldsymbol{a}}_t|_{[l_t]}$, $\boldsymbol{w}_t := \boldsymbol{b}_t - \hat{\boldsymbol{b}}_t|_{[l_t]}$ to $P_{\mathcal{R}}$.

Verify†: On input i, j.

(i) The sender $P_{\mathcal{S}}$ sends $u_{1,l_1+1} := b_{2,j} - \hat{a}_{1,l_2+1}$, $u_{2,l_1+1} := b_{1,i} - \hat{a}_{2,l_2+1}$ and $\hat{b}_{1,l_1+1} - \hat{b}_{2,l_2+1}$ to $P_{\mathcal{R}}$.
(ii) For $t \in [2]$, $P_{\mathcal{R}}$ computes $\boldsymbol{v}_t := \hat{\boldsymbol{v}}_t + \boldsymbol{u}_t \cdot \alpha_t + (\boldsymbol{w}_t \parallel 0)$, where $\boldsymbol{v}_t|_{[l_t]} = \boldsymbol{a}_t \cdot \alpha_t + \boldsymbol{b}_t$. Note that $v_{1,l_1+1} = b_{2,j} \cdot \alpha_1 + \hat{b}_{1,l_1+1}$, $v_{2,l_2+1} = b_{1,i} \cdot \alpha_2 + \hat{b}_{2,l_2+1}$.
(iii) The receiver $P_{\mathcal{R}}$ checks that $\alpha_2 \cdot v_{1,i} - \alpha_1 \cdot v_{2,j} + v_{1,l_1+1} - v_{2,l_2+1} = \hat{b}_{1,l_1+1} - \hat{b}_{2,l_2+1}$. If the check fails, $P_{\mathcal{R}}$ aborts.

Verify‡: On input i, j.

(i) The sender $P_{\mathcal{S}}$ sends $w_{1,l_1+1} := b_{2,j} - \hat{b}_{1,l_1+1}$, $u_{2,l_2+1} := a_{1,i} - \hat{a}_{2,l_2+1}$ and $\hat{b}_{2,l_2+1} - \hat{a}_{1,l_1+1}$ to $P_{\mathcal{R}}$.
(ii) The receiver $P_{\mathcal{R}}$ computes $\boldsymbol{v}_1 := \hat{\boldsymbol{v}}_1 + (\boldsymbol{u}_1 \parallel 0) \cdot \alpha_1 + \boldsymbol{w}_1$ and $\boldsymbol{v}_2 := \hat{\boldsymbol{v}}_2 + \boldsymbol{u}_2 \cdot \alpha_2 + (\boldsymbol{w}_2 \parallel 0)$, where $\boldsymbol{v}_t|_{[l_t]} = \boldsymbol{a}_t \cdot \alpha_t + \boldsymbol{b}_t$, for $t \in [2]$. Note that $v_{1,l_1+1} = \hat{a}_{1,l_1+1} \cdot \alpha_1 + b_{2,j}$ and $v_{2,l_2+1} = a_{1,i} \cdot \alpha_2 + \hat{b}_{2,l_2+1}$.
(iii) The receiver $P_{\mathcal{R}}$ checks that $v_{2,j} - v_{1,l_1+1} - v_{1,i} \cdot \alpha_2 + v_{2,l_2+1} \cdot \alpha_1 = \alpha_1 \cdot (\hat{b}_{2,l_2+1} - \hat{a}_{1,l_1+1})$. If the check fails, $P_{\mathcal{R}}$ aborts.

Fig. 6: Protocol for eVOLE over $\mathtt{GR}(2^k,d)$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model.

Suppose $\boldsymbol{a}_1, \boldsymbol{b}_1 \in \mathtt{GR}(2^k,d)^{l_1}$, $\boldsymbol{a}_2, \boldsymbol{b}_2 \in \mathtt{GR}(2^k,d)^{l_2}$ are $P_{\mathcal{S}}$'s inputs and α_1, α_2 are $P_{\mathcal{R}}$'s inputs to two VOLE instances, respectively. For proving the equality constraint of some $a_{1,i} = a_{2,j}$, where $i \in [l_1], j \in [l_2]$, we apply a Galois ring

[9] We remark that for convenience, the construction proves one equality constraint, which can be naturally extended to prove an arbitrary number of equality constraints.

analogue of the check mechanism [13]. By $P_{\mathcal{S}}$ setting $a_{1,l_1+1} := b_{2,j}$, $a_{2,l_2+1} := b_{1,i}$, $b_{1,l_1+1}, b_{2,l_2+1} \overset{\$}{\leftarrow} \mathtt{GR}(2^k, d)$ and sending $b_{1,l_1+1} - b_{2,l_2+1}$ to $P_{\mathcal{R}}$, we have that

$$\alpha_2 \cdot v_{1,i} - \alpha_1 \cdot v_{2,j} + v_{1,l_1+1} - v_{2,l_2+1} = b_{1,l_1+1} - b_{2,l_2+1} \tag{1}$$

holds if $a_{1,i} = a_{2,j}$. If $a_{1,i} \neq a_{2,j}$ and α_1, α_2 are uniformly and independently distributed, by Lemma 1, Equation (1) holds with probability at most $1/2^{d-1}$.

For proving the equality constraint of some $b_{1,i} = a_{2,j}$, we cannot reduce it to the above case like [13] [10]. We use another equation for the check. By $P_{\mathcal{S}}$ setting $b_{1,l_1+1} := b_{2,j}$, $a_{2,l_2+1} := a_{1,i}$, $b_{2,l_2+1}, a_{1,l_1+1} \overset{\$}{\leftarrow} \mathtt{GR}(2^k, d)$ and sending $b_{2,l_2+1} - a_{1,l_1+1}$ to $P_{\mathcal{R}}$, we have that

$$v_{2,j} - v_{1,l_1+1} - v_{1,i} \cdot \alpha_2 + v_{2,l_2+1} \cdot \alpha_1 = \alpha_1 \cdot (b_{2,l_2+1} - a_{1,l_1+1}) \tag{2}$$

holds if $b_{1,i} = a_{2,j}$. Similarly, by Lemma 1, Equation (2) holds with probability at most $1/2^{d-1}$ if $b_{1,i} \neq a_{2,j}$ and α_1, α_2 are uniformly and independently distributed.

We have the following proposition (the proof is deferred to Appendix B.2).

Proposition 4. *$\Pi_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model.*

NIZK. The authors of [13] introduced a simple kind of information-theoretic proof system for proving circuit satisfiability called *line point zero knowledge* (LPZK). Informally, in an LPZK proof, the prover $\mathcal{P}$ generates from the witness w and the circuit $\mathcal{C}$ an affine line $\boldsymbol{v}(x) := \boldsymbol{a} \cdot x + \boldsymbol{b}$ over a field $\mathbb{F}_q$. The verifier $\mathcal{V}$ queries a single point α and obtains the evaluation $\boldsymbol{v}(\alpha)$, then $\mathcal{V}$ decides whether to accept the proof or reject. The LPZK proof system is statistical in the VOLE-hybrid model and can be realized by a single invocation of VOLE. We naturally extend the LPZK-NIZK construction for fields of [13] to Galois rings, by simply replacing the field $\mathbb{F}_q$ with a Galois ring $\mathtt{GR}(2^k, d)$ (see $\Pi_{\mathrm{NIZK}}^{q,t}$ in Figure 15). We remark that the soundness error is decreased from $\mathcal{O}(1/q)$ to $\mathcal{O}\big(1/2^d\big)$, which can be negligible in the security parameter for a sufficiently large d. The construction communicates 3 elements over $\mathtt{GR}(2^k, d)$ per multiplication gate and is "free" for addition gates. We have the following proposition with the detailed proof in Appendix B.1.

Proposition 5. *Protocol $\Pi_{\mathrm{NIZK}}^{q,t}$ realizes $\mathcal{F}_{\mathrm{ZK}}$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model with soundness error $1/2^{d-1}$ and statistical security.*

cVOLE. We next provide a high-level overview of the construction of cVOLE from eVOLE and LPZK-NIZK. The eVOLE is used to move the sender's inputs to multiple VOLE instances (n_1 lines, on fixed points $\alpha_1, ..., \alpha_{n_1}$) to another VOLE instance (another line, on a random point γ), where a LPZK-NIZK over Galois rings can be performed.

When substituting parallel VOLE instances in the semi-honest protocol Π_{NISC} with the above cVOLE construction, the eVOLE requirement should be satisfied

[10] In more detail, we cannot reduce $v_{1,i} = a_{1,i} \cdot \alpha_1 + b_{1,i}$ to $v_{1,i} \cdot \alpha_1^{-1} = b_{1,i} \cdot \alpha_1^{-1} + a_{1,i}$, as $1/2^d$ fraction of elements in $\mathtt{GR}(2^k, d)$ are zero divisors.

(i.e. $P_{\mathcal{R}}$'s inputs need to be uniformly and independently distributed). However, $P_{\mathcal{R}}$ has fixed inputs in the NISC setting. To solve this, $n_1 + 2$ VOLE instances with $P_{\mathcal{R}}$'s corresponding inputs $(\alpha_1 + \beta, ..., \alpha_{n_1} + \beta, \beta, \gamma)$ are required, where $\alpha_1, ..., \alpha_n \in \mathrm{Im}(\phi)$ and $\beta, \gamma \xleftarrow{\$} \mathtt{GR}(2^k, d)$, and the equality constraints are proven between VOLE instances corresponding to $(\alpha_i + \beta, \gamma)$ and (β, γ). The cVOLE protocol is presented in Figure 7, and we have the following corollary with a deferred proof in Appendix B.2.

Corollary 2. *$\mathit{\Pi}_{\mathrm{cVOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\mathrm{cVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $(\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}, \mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)})$-hybrid model.*

Protocol $\mathit{\Pi}_{\mathrm{cVOLE}}^{\mathtt{GR}(2^k,d)}$

Parameterized by a Galois ring $\mathtt{GR}(2^k, d)$, a sequence of n positive integers $l_1, ..., l_n$, and an arithmetic circuit $\mathcal{C}$ over $\mathtt{GR}(2^k, d)$ on $q_a + q_b = q \leq 2\sum_{i=1}^{n} l_i$ inputs with t multiplication gates. Let $L_1 = 0$ and for $i = 2, 3, ..., n+1$, let $L_i = l_1 + ... + l_{i-1}$. Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$. The receiver $P_{\mathcal{R}}$ has inputs $\alpha_i \in \mathtt{GR}(2^k, d)$ and the sender $P_{\mathcal{S}}$ has inputs $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathtt{GR}(2^k, d)^{l_i}$, for $i \in [n]$. Suppose $\mathcal{C}$ takes q_a inputs from $\boldsymbol{a}$ entries and q_b inputs from $\boldsymbol{b}$ entries.

1. The two parties invoke the **Setup phase** of $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$ with $P_{\mathcal{R}}$'s inputs $(\alpha_1 + \beta, ..., \alpha_n + \beta, \beta, \gamma)$, where $\beta, \gamma \xleftarrow{\$} \mathtt{GR}(2^k, d)$.
2. For $i \in [n]$, $P_{\mathcal{S}}$ picks $\boldsymbol{e}_i \xleftarrow{\$} \mathtt{GR}(2^k, d)^{l_i}$ and sends $(\boldsymbol{a}_i, \boldsymbol{b}_i + \boldsymbol{e}_i)$ with session id i to $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$. For the $(n+1)$-st instance of VOLE, $P_{\mathcal{S}}$ computes $\boldsymbol{a}_{n+1} := \boldsymbol{a}_1 \parallel ... \parallel \boldsymbol{a}_n, \boldsymbol{b}_{n+1} := \boldsymbol{e}_1 \parallel ... \parallel \boldsymbol{e}_n$ and sends $(n+1; \boldsymbol{a}_{n+1}, \boldsymbol{b}_{n+1}; L_{n+1})$ to $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$. For the $(n+2)$-nd instance of VOLE, if $a_{i,j}$ is the k-th input from $\boldsymbol{a}$ entries to circuit $\mathcal{C}$, set $a_{n+2,k} := a_{i,j}$; else if $b_{i,j}$ is the k-th input from $\boldsymbol{b}$ entries to circuit $\mathcal{C}$, set $a_{n+2,q_a+k} := b_{i,j}$ and $a_{n+2,q+k} := b_{n+1,L_i+j}$. Additionally, $P_{\mathcal{S}}$ picks $\boldsymbol{b}_{n+2} \xleftarrow{\$} \mathtt{GR}(2^k, d)^{q+q_b}$. Then, $P_{\mathcal{S}}$ sends $(n+2; \boldsymbol{a}_{n+2}, \boldsymbol{b}_{n+2}; q+q_b)$ to $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$. The receiver $P_{\mathcal{R}}$ receives $\boldsymbol{v}_1, ..., \boldsymbol{v}_{n+2}$ from $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$.
3. By invoking the **Verify phases** of $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$, $P_{\mathcal{R}}$ verifies that
   (i) $a_{i,j} = a_{n+2,k}$ and $a_{n+1,L_i+j} = a_{n+2,k}$, if $a_{i,j}$ is the k-th input from $\boldsymbol{a}$ entries to circuit $\mathcal{C}$, for $k \in [q_a]$.
   (ii) $b_{i,j} = a_{n+2,q_a+k}$ and $b_{n+1,L_i+j} = a_{n+2,q+k}$, if $b_{i,j}$ is the k-th input from $\boldsymbol{b}$ entries to circuit $\mathcal{C}$, for $k \in [q_b]$. Recompute v_{n+2,q_a+k} by subtracting $v_{n+2,q+k}$.
4. Invoke the subprotocol $\mathit{\Pi}_{\mathrm{NIZK}}^{q,t}$ with inputs $\{[a_{n+2,i}]_\gamma\}_{i\in[q]}$ to verify that $\{[a_{n+2,i}]_\gamma\}_{i\in[q]}$ is a satisfying assignment for $\mathcal{C}$. If any of above verifications fails, $P_{\mathcal{R}}$ aborts.

Fig. 7: Protocol for Certified VOLE with a general arithmetic constraint in the $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model

In particular, if instantiating $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$ with $\Pi_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$, we obtain a cVOLE/VOLE construction, which essentially admits a NISC/VOLE protocol with reusable malicious security for branching programs over $\mathtt{GR}(2^k, d)$. If we further assume both parties follow RMFE encoding honestly, the protocol can securely compute branching programs over $\mathbb{Z}_{2^k}$.

Putting all pieces together. Recall that NM-RMFE allows for "extraction" when the adversary does not follow the NM-RMFE encoding honestly. The final step is upgrading standard RMFE to NM-RMFE and we do not need to assume both parties follow NM-RMFE encoding honestly. For NISC tasks that compute BPs over $\mathbb{Z}_{2^k}$, using the cVOLE technique[11] and substituting Degree-3 RMFEs by an $(m, d; 3)$-NM-RMFE (ϕ, ψ) in Π_{NISC}, we have the following theorem.

Theorem 4 (rNISC/VOLE from NM-RMFE). *Suppose $f : \mathbb{Z}_{2^k}^{n_1} \times \mathbb{Z}_{2^k}^{n_2} \to \mathbb{Z}_{2^k}^{t}$ is a sender-receiver functionality whose i-th output can be computed by an arithmetic branching program over $\mathbb{Z}_{2^k}$ of size $s_i + 1$ that depends on d_i inputs. Let (ϕ, ψ) be an $(m, d; 3)$-NM-RMFE over $\mathbb{Z}_{2^k}$ and κ be the statistical security parameter. Then f admits an rNISC/VOLE protocol with the following features:*

- *The protocol takes $n_1 + 2$ parallel VOLE instances over $\mathtt{GR}(2^k, d)$, and outputs m executions of f.*
- *The protocol is secure against a malicious sender and a malicious receiver.*
- *Assume the branching program admits a verification circuit $\mathcal{C}$ that takes q_a inputs from $\boldsymbol{a}$ entries, q_b inputs from $\boldsymbol{b}$ entries. The circuit $\mathcal{C}$ has $S := \sum_{i=1}^{t}(d_i\binom{s_i}{2} + s_i^3)$ multiplication gate. The total length of VOLE instances is $2S + 6q_a + 7q_b + \sum_{i=1}^{t} d_i\binom{s_i}{2}$, and $3S + 1 + 8q_a + 9q_b + 2\sum_{i=1}^{t} d_i\binom{s_i}{2}$ elements over $\mathtt{GR}(2^k, d)$ are communicated.*
- *The simulation error is $\varepsilon = \mathcal{O}\big(1/2^d + 1/2^\kappa\big)$.*

Proof. The construction is obtained by replacing RMFE with NM-RMFE and VOLE with cVOLE in Π_{NISC}. Similar to that in Theorem 1, NM-RMFE allows for simulating the cheating behavior of not following NM-RMFE encoding, thus the resulting NISC protocol has reusable malicious security in the $\mathcal{F}_{\mathrm{cVOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model. With a statistical secure instantiation of $\mathcal{F}_{\mathrm{cVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model, the resulting NISC protocol has reusable malicious security in the $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model. The simulation error is computed by the union bound of the soundness of cVOLE and that of NM-RMFE. For the communication complexity, recall that in cVOLE we require two additional entries of VOLE for an $a_{i,j}$ entry that is an input to $\mathcal{C}$, and three additional entries of VOLE for an $b_{i,j}$ entry that is an input to $\mathcal{C}$. Thus, we can obtain the above results. □

Recall that by Theorem 2, we realize a slightly weaker variant of NM-RMFE, where $\boldsymbol{x} \in (\mathbb{Z}_{2^k}^*)^m$. We argue that this imperfect construction is still sufficient for building rNISC/VOLE. When instantiating NM-RMFE with our construction in the above rNISC/VOLE, intuitively the adversary $\mathcal{A}$ is allowed to query some

[11] The circuit $\mathcal{C}$ (specifies the arithmetic constraints in DARE) is of size $S := \sum_{j=1}^{n_1} S_j + \sum_{i=1}^{t} s_i^3$ according to naive matrix multiplication (S_j, s_i are defined as in Theorem 3).

positions of the honest party's inputs to VOLEs, and he learns whether they are all in $\mathbb{Z}_{2^k}^*$ (through observing the validity of VOLE outputs). More precisely, if the receiver is corrupted, recall that when we implement parallel DAREs, $P_{\mathcal{S}}$'s input $\boldsymbol{Y}$ will never be put into $\boldsymbol{a}$ entries, thus this attack can be avoided by instantiating Lemma 4 with $\bar{\mathcal{G}}_1^s \leq \mathcal{G}_1^s, \bar{\mathcal{G}}_2^s \leq \mathcal{G}_2^s$ such that their entries are even (i.e. zero divisors) except for the main diagonal. If the sender is corrupted, we let $P_{\mathcal{R}}$ sample the mask α from $\phi((\mathbb{Z}_{2^k}^*)^m)$[12], then applying this attack will always lead to $P_{\mathcal{R}}$ aborting.

5 Amortized Computationally Secure NISC

Our NM-RMFE approach admits rNISC/VOLE with good asymptotic efficiency and practical concrete efficiency for a relative large batch size m. To achieve better concrete efficiency (especially for small m), we consider weaker security models and explore computationally secure solutions to constructing (reusable) NISC protocols for BPs over $\mathbb{Z}_{2^k}$.

In this section, we first show how to force the sender to follow RMFE encoding efficiently. Then, we present two approaches to forcing the receiver to follow RMFE encoding honestly. The former approach is based on cut-and-choose and makes black-box use of any two-round reusable VOLE protocol. The latter OT-based approach is highly efficient but unfortunately not reusable. Combining all together, we obtain two NISC protocols, a concrete efficient reusable NISC construction with communication overhead $\mathcal{O}(\lambda)$, where λ is the computational security parameter, and a highly efficient NISC construction with communication overhead close to a constant.

5.1 Forcing the Sender to Follow RMFE Encoding

A naive solution is to augment the NIZK subprotocol in cVOLE to include a proof for correct RMFE encoding. However, this would lead to proving circuit satisfiability on a circuit of large size, which is inefficient. To this end, we use a more efficient technique, re-embedding VOLE (embVOLE) [20] that was originally designed for ZK protocols and allows to "prove" RMFE constraints before NIZK is applied. We slightly generalize the random embVOLE functionality to fit NISC settings (see $\mathcal{F}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ in Figure 8), which then allows the receiver $P_{\mathcal{R}}$ to query $\hat{\boldsymbol{a}}|_J$'s projection on $\mathrm{Ker}(\psi)$, for some $J \subseteq [l]$. For some $i \in J$, to obtain $[a_i]_\alpha$ from a random $[\hat{a}_i]_\alpha$, $P_{\mathcal{S}}$ is supposed to send $u_i := a_i - \hat{a}_i$ to $P_{\mathcal{R}}$, then $P_{\mathcal{R}}$ can verify whether $u_i = \tau(\hat{a}_i) - \hat{a}_i$ ($P_{\mathcal{R}}$ learns $\tau(\hat{a}_i) - \hat{a}_i$ from querying $\hat{a}_i$'s projection on $\mathrm{Ker}(\psi)$). If the check fails, $a_i \notin \mathrm{Im}(\phi)$ and $P_{\mathcal{R}}$ will abort, which forces $P_{\mathcal{S}}$'s inputs to satisfy RMFE constraints.

The random embVOLE protocol $\Pi_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ (slightly adapted from [20]) is presented in Figure 16, which can be made non-interactive by Fiat-Shamir heuristic. We remark that $\Pi_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ has communication overhead close to a constant, when the length l is relatively large.

[12] This would slightly affect the eVOLE soundness.

Functionality $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$

$\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$ extends the random VOLE functionality $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{VOLE}}$ (Figure 1). **Setup phase** and **Send phases** are identical to those in $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{VOLE}}$, respectively. Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, length parameters $l_1,...,l_n \in \mathbb{N}$. Let (ϕ,ψ) be an $(m,d;3)$-RMFE over $\mathbb{Z}_{2^k}$, and $\tau := \phi \circ \psi$.

Deliver phases: Upon receiving $(sid;\mathtt{Delivery};l_i;J)$ from the adversary where $i \in [n], J \subseteq [l_i]$ and sid is a session identifier, verify that there are stored values $(sid;\alpha)$ and $(sid;\boldsymbol{a},\boldsymbol{b};l_i)$; else ignore that message. Next, compute $\boldsymbol{\eta} := (\tau(\boldsymbol{a}) - \boldsymbol{a})|_J \in \mathrm{Ker}(\psi)^{|J|}$, and send $(sid;\boldsymbol{\eta};l_i;J)$ to $P_{\mathcal{R}}$, and ignore further messages $(sid;\mathtt{Delivery};l_i;J)$ from the adversary with the same session identifier sid.

Fig. 8: Ideal functionality for random re-embedding VOLE over $\mathtt{GR}(2^k,d)$.

We slightly modify the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{eVOLE}}$ functionality and the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{cVOLE}}$ functionality to include checking RMFE constraints (see the resulting functionalities $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ in Figure 17, and $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}}$ in Figure 19.). We present the $\ddot{e}$VOLE protocol in Figure 18, which is the same as $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\mathrm{eVOLE}}$, except that $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ is built upon $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$. We also present the $\ddot{c}$VOLE protocol in Figure 20, which is the same as $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\mathrm{cVOLE}}$, except that $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}}$ is built upon $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$.

We have the following lemmas (proofs are deferred to Appendix B.2).

Lemma 6. *$\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ realizes $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$-hybrid model with reusable malicious security.*

Lemma 7. *Instantiating $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ with $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$, we have that $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}}$ realizes $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}}$ in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$-hybrid model with reusable malicious security.*

5.2 Forcing the Receiver to follow RMFE Encoding

In this section, we consider the remaining issue of forcing the receiver to follow RMFE encoding. We remark again that the malicious receiver in the $\mathit{\Pi}_{\mathrm{NISC}}$ protocol can only cheat by deviating from RMFE encoding. Therefore, in Figure 9 we define a variant of VOLE, called ϕVOLE, where the receiver's inputs are restricted in the image of a RMFE map ϕ (this leaves no room for the malicious receiver to cheat when building $\mathit{\Pi}_{\mathrm{NISC}}$ upon it). In general, we construct computationally secure NISC protocols following the roadmap below,

$$\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}} \Longrightarrow \mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}} \Longrightarrow \mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}} \overset{+\mathit{\Pi}^{q,t}_{\mathrm{NIZK}}}{\Longrightarrow} \mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}} \Longrightarrow \mathrm{NISC}.$$

Recall that in $\ddot{c}$VOLE, there are $n+2$ VOLE instances, and the first $n+1$ VOLE instances correspond to $P_{\mathcal{R}}$'s inputs $\alpha_1+\beta,...,\alpha_n+\beta,\beta$, respectively. We remark that $\alpha_1+\beta, ..., \alpha_n+\beta, \beta$ will be restricted in $\mathrm{Im}(\phi)$, the $\ddot{e}$VOLE soundness is $1/2^d + 1/2^m$ rather than $1/2^{d-1}$. We present two ϕVOLE constructions with different features as follows.

Functionality $\mathcal{F}_{\phi\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$

Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$. Parameterized by a ring $\mathtt{GR}(2^k, d)$, length parameters $l_1, ..., l_n \in \mathbb{N}$ and the RMFE map ϕ.

Setup phase: Upon receiving input $(sid; \alpha)$ from $P_{\mathcal{R}}$ where $\alpha \in \mathrm{Im}(\phi)$ and sid is a session identifier, store $(sid; \alpha)$, send $(sid; \mathtt{initialized})$ to the adversary and ignore any further inputs from $P_{\mathcal{R}}$ with the same session identifier sid.
Send phases: Upon receiving input $(sid; \boldsymbol{a}; \boldsymbol{b}; l_i)$ from $P_{\mathcal{S}}$, where $(\boldsymbol{a}, \boldsymbol{b}; l_i) \in \mathtt{GR}(2^k, d)^{l_i} \times \mathtt{GR}(2^k, d)^{l_i} \times \mathbb{N}$ and sid is a session identifier, store $(sid; \boldsymbol{a}; \boldsymbol{b}; l_i)$, send $(sid; \mathtt{sent}; l_i)$ to the adversary, and ignore any further inputs from $P_{\mathcal{S}}$ with the same session identifier sid.
Deliver phases: Upon receiving a message $(sid; \mathtt{Delivery}; l_i)$ from the adversary where $l_i \in \mathbb{N}$ and sid is a session identifier, verify that there are stored inputs $(sid; \alpha)$ from $P_{\mathcal{R}}$ and $(sid; \boldsymbol{a}, \boldsymbol{b}; l_i)$ from $P_{\mathcal{S}}$; else ignore that message. Next, compute $\boldsymbol{v} := \boldsymbol{a} \cdot \alpha + \boldsymbol{b}$, send $(sid; \boldsymbol{v}; l_i)$ to $P_{\mathcal{R}}$, and ignore further messages $(sid; \mathtt{Delivery}; l_i)$ from the adversary with the same session identifier sid.

Fig. 9: Ideal functionality for chosen-input ϕVOLE over $\mathtt{GR}(2^k, d)$.

The ϕVOLE construction based on cut-and-choose. We are initially inspired by the approach of the concurrent work [17], where they bypassed the impossible result of OT-based rNISC [10] via making a black-box use of OT protocols with random oracles. Making a black-box use of OT (VOLE) protocols instead of assuming black-box access to an ideal OT (VOLE) functionality allows for "connecting" the inputs that the parties use to compute OT messages with the other cryptographic primitives, e.g. commitments. Let $(\mathit{\Pi}_{P_{\mathcal{R}},1}^{\mathtt{GR}(2^k,d)}, \mathit{\Pi}_{P_{\mathcal{S}},1}^{\mathtt{GR}(2^k,d)}, \mathit{\Pi}_{P_{\mathcal{R}},2}^{\mathtt{GR}(2^k,d)})$ be a two-message reusable VOLE protocol over $\mathtt{GR}(2^k, d)$, where $P_{\mathcal{R}}$ runs $\mathit{\Pi}_{P_{\mathcal{R}},1}^{\mathtt{GR}(2^k,d)}$ on her private input and random tape to obtain the first round message π_1, then $P_{\mathcal{S}}$ computes the second round message π_2 by running $\mathit{\Pi}_{P_{\mathcal{S}},1}^{\mathtt{GR}(2^k,d)}$ on π_1 and his private input, and finally $P_{\mathcal{R}}$ obtains the result by evaluating $\mathit{\Pi}_{P_{\mathcal{R}},2}^{\mathtt{GR}(2^k,d)}$ on π_2 and her random tapes. Recall that our goal here is to guarantee that $P_{\mathcal{R}}$'s inputs are restricted in the image of an RMFE map, and intuitively our high-level idea is cut-and-choose. The receiver $P_{\mathcal{R}}$ commits to inputs and random tapes used for generating her first VOLE messages (several copies), and reveals some of them according to queries to a random oracle. The sender $P_{\mathcal{S}}$ then can check whether $P_{\mathcal{R}}$'s inputs are valid and the VOLE messages are correctly computed. However, there is still a gap since $P_{\mathcal{R}}$'s inputs are private (for computing rNISC/VOLE tasks) and none of them could be revealed. We overcome this issue by observing that in cVOLE, $P_{\mathcal{R}}$'s inputs to multiple VOLE instances are masked with a random β (suppose $P_{\mathcal{R}}$ has inputs $\alpha_1, ..., \alpha_n$), thus one of these commitments $[\![\alpha_1 + \beta]\!], ..., [\![\alpha_n + \beta]\!], [\![\beta]\!]$ can be opened. Repeating the procedure for a sufficient number of times and $P_{\mathcal{S}}$ will believe that $P_{\mathcal{R}}$ behaves honestly with an overwhelming probability. The final problem is that a malicious $P_{\mathcal{R}}$ may not provide consistent inputs in different iterations. We show that if further assuming the commitment scheme $(\mathtt{Com}, \mathtt{Open})$ is linearly homomorphic over Galois rings, we can apply a random linear combination check on the committed inputs, where the random coefficients can be obtained by querying a random oracle as well. Since RMFEs over $\mathbb{Z}_{2^k}$ are $\mathbb{Z}_{2^k}$-linear maps, these random coefficients can be sampled from $\mathbb{Z}_{2^k}$. We present the desired protocol in Figure 10, which has $\mathcal{O}(\lambda)$ communication overhead due to cut-and-choose.

We have the following theorem (see proof in Appendix B.6).

Theorem 5. *Assuming a two-message reusable VOLE protocol over* $\mathtt{GR}(2^k, d)$, *and a linearly-homomorphic commitment scheme over* $\mathtt{GR}(2^k, d)$, $\Omega^{\mathtt{GR}(2^k,d)}_{\phi\text{VOLE}}$ *realizes* $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\phi\text{VOLE}}$ *in the random oracle model.*

Combining all pieces together, we obtain an rNISC protocol (for computing BPs over $\mathbb{Z}_{2^k}$) that makes black-box use of any two-message reusable VOLE protocol in the random oracle model. We note that, for constructing an rNISC computing a general function f, the work [17] provides a compiler that lifts a non-reusable (malicious secure) NISC protocol (computes a related function f') to a reusable one. We observe that their tool is strong, but quite heavy and expensive for general functions, while for some simple f, the efficiency can be significantly improved. To optimize the efficiency, we can use their rNISC compiler to obtain a reusable VOLE protocol over Galois rings from black-box use of OT [13], which is expected to have good concrete efficiency.

The ϕVOLE construction from OT. We start with an observation on RMFEs. Let (ϕ, ψ) be an $(m, d; D)$-RMFE over $\mathbb{Z}_{2^k}$. We have that $\mathtt{GR}(2^k, d)$ can be viewed as a linear space over $\mathbb{Z}_{2^k}$ with dimension d. As ϕ, ψ are $\mathbb{Z}_{2^k}$-linear maps, $\mathrm{Im}(\phi)$ can be viewed as a linear space over $\mathbb{Z}_{2^k}$ with dimension m, which is a subspace of $\mathtt{GR}(2^k, d)$ as well. Therefore, there exist a basis $\gamma_1, ..., \gamma_m \in \mathtt{GR}(2^k, d)$ such that

$$\phi : \mathbb{Z}_{2^k}^m \to \mathtt{GR}(2^k, d), \quad (a_1, ..., a_m) \mapsto a_1\gamma_1 + ... + a_m\gamma_m.$$

We call such $\gamma_1, ..., \gamma_m$ an RMFE-basis. Let $\boldsymbol{\alpha} \in \mathbb{Z}_{2^k}^m$, and $\boldsymbol{a}, \boldsymbol{b}_1, ..., \boldsymbol{b}_m \in \mathtt{GR}(2^k, d)^l$. Denote $\boldsymbol{a} \cdot \alpha_i + \boldsymbol{b}_i$ by $\boldsymbol{v}_i$, for $i \in [m]$. We have that

$$\begin{aligned}\sum_{i=1}^m \boldsymbol{v}_i \cdot \gamma_i = \sum_{i=1}^m (\boldsymbol{a} \cdot \alpha_i + \boldsymbol{b}_i) \cdot \gamma_i &= \boldsymbol{a} \cdot (\sum_{i=1}^m \alpha_i \gamma_i) + \sum_{i=1}^m \boldsymbol{b}_i \cdot \gamma_i \\ &= \boldsymbol{a} \cdot \phi(\alpha_1, ..., \alpha_m) + \sum_{i=1}^m \boldsymbol{b}_i \cdot \gamma_i.\end{aligned}$$

[13] VOLE is essentially a simple NISC task, therefore the inner protocol of the rNISC compiler can be an OT-based (non-reusable) VOLE protocol. The specific construction is beyond the scope of this work, thus omitted.

Protocol $\Omega^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$

Parameterized by a Galois ring $\mathtt{GR}(2^k, d)$, length parameter l, computational security parameter λ and cut-and-choose parameter $t = \mathcal{O}(\lambda)$. Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$. Let $(\Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}, \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{S}},1}, \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},2})$ be a two-message reusable VOLE protocol over $\mathtt{GR}(2^k, d)$. Let $H_1 : \{0,1\}^* \to (\{0,1\}^{s_t})^{(n+1)\times t}$ and $H_2 : \{0,1\}^* \to \mathbb{Z}^{n\times t}_{2^k}$ be two hash functions, where s_t is the number of bits needed to toss a biased coin (used for interpreting the hash values as matrices with a special form). Let $(\mathtt{Com}, \mathtt{Open})$ be a commitment scheme. Suppose $P_{\mathcal{R}}$ has input $\boldsymbol{\alpha} \in \mathrm{Im}(\phi)^n$, and $P_{\mathcal{S}}$ has input $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathtt{GR}(2^k, d)^l$ for $i = 0, 1, ..., n$.

1. **Round-1:**
   (a) For $j \in [t]$, $P_{\mathcal{R}}$ samples $\beta_j \xleftarrow{\$} \mathrm{Im}(\phi)$. For $i = 0, 1, ..., n$ and $j \in [t]$, $P_{\mathcal{R}}$ samples random tape $r_{i,j}$ to be used in the protocol $\Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}$.
   (b) For $i \in [n]$, $P_{\mathcal{R}}$ computes $[\![\alpha_i]\!] := \mathtt{Com}(\alpha_i)$. For $j \in [t]$, $P_{\mathcal{R}}$ computes $\pi_{0,j,1} := \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}(\beta_j; r_{0,j})$, $[\![\beta_j]\!] := \mathtt{Com}(\beta_j)$, and $[\![r_{0,j}]\!] := \mathtt{Com}(r_{0,j})$. For $i \in [n]$ and $j \in [t]$, $P_{\mathcal{R}}$ computes $\pi_{i,j,1} := \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}(\alpha_i + \beta_j; r_{i,j})$, $[\![\alpha_i + \beta_j]\!] := \mathtt{Com}(\alpha_i + \beta_j)$ and $[\![r_{i,j}]\!] := \mathtt{Com}(r_{i,j})$. Denote the sequence of values $(\{[\![\alpha_i]\!]\}_{i\in[n]}, \{\pi_{0,j,1}, [\![\beta_j]\!], [\![r_{0,j}]\!]\}_{j\in[t]}, \{\pi_{i,j,1}, [\![\alpha_i + \beta_j]\!], [\![r_{i,j}]\!]\}_{i\in[n],j\in[t]})$ by $\mathtt{msg}$.
   (c) The receiver $P_{\mathcal{R}}$ computes $S = H_1(\mathtt{msg})$, and interprets S as a $(n + 1) \times t$ matrix such that each column contains at most one 1 and zeros everywhere else (we require S has at least one column with all zeros). Then, $P_{\mathcal{R}}$ computes $\{\chi_{i,j}\}_{i\in[n],j\in[t]} := H_2(\mathtt{msg})$, where $\chi_{i,j} \in \mathbb{Z}_{2^k}$.
   (d) The receiver $P_{\mathcal{R}}$ computes $\mathtt{unv} := \mathtt{Open}(\sum_{i\in[n],j\in[t]} \chi_{i,j} \cdot ([\![\alpha_i + \beta_j]\!] - [\![\alpha_i]\!] - [\![\beta_j]\!]))$.
   (e) The receiver $P_{\mathcal{R}}$ sends $(\{\mathtt{msg}, \mathtt{unv}, \{\mathtt{Open}([\![\alpha_i + \beta_j]\!]), \mathtt{Open}([\![r_{i,j}]\!])\}_{S_{i,j}=1})$ as the first message.
2. **Round-2:**
   (a) The sender $P_{\mathcal{S}}$ recomputes S and $\{\chi_{i,j}\}_{i\in[n],j\in[t]}$ as in step-(c) of Round-1 and checks whether the openings are valid. In particular, $P_{\mathcal{S}}$ checks whether $\mathtt{unv}$ is an opening of $[\![0]\!]$.
   (b) For $i \in [n], j \in [t]$ such that $S_{i,j} = 1$, $P_{\mathcal{S}}$ checks whether $\alpha_i + \beta_j \in \mathrm{Im}(\phi)$. For $j \in [t]$ such that $S_{0,j} = 1$, $P_{\mathcal{S}}$ checks whether $\beta_j \in \mathrm{Im}(\phi)$.
   (c) For $i \in [n], j \in [t]$ such that $S_{i,j} = 1$, $P_{\mathcal{S}}$ checks whether $\pi_{i,j,1} = \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}(\alpha_i + \beta_j; r_{i,j})$. For $j \in [t]$ such that $S_{0,j} = 1$, $P_{\mathcal{S}}$ checks whether $\pi_{0,j,1} = \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},1}(\beta_j; r_{0,j})$.
   (d) If any of above checks fail, $P_{\mathcal{S}}$ aborts. Otherwise, $P_{\mathcal{S}}$ randomly picks j satisfying the j-th column of S are all 0's.
   (e) $P_{\mathcal{S}}$ computes $\pi_{i,2} := \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{S}},1}(\boldsymbol{a}_i, \boldsymbol{b}_i, \pi_{i,j,1})$ for $i = 0, 1, ..., n$. $P_{\mathcal{S}}$ sends $(\pi_{0,2}, ..., \pi_{n,2}, j)$ to $P_{\mathcal{R}}$.
3. **Output Computation:** The receiver $P_{\mathcal{R}}$ computes $\boldsymbol{v}_i := \Pi^{\mathtt{GR}(2^k,d)}_{P_{\mathcal{R}},2}(\pi_{i,2}, r_{i,j})$, for $i = 0, 1, ..., n$.

Fig. 10: Protocol for ϕVOLE making black-box use of VOLE over $\mathtt{GR}(2^k, d)$.

Denoting $\sum_{i=1}^{m} \boldsymbol{v}_i \cdot \gamma_i$ by $\boldsymbol{v}$ and $\sum_{i=1}^{m} \boldsymbol{b}_i \cdot \gamma_i$ by $\boldsymbol{b}$, we obtain an VOLE instance $\boldsymbol{v} = \boldsymbol{a} \cdot \phi(\boldsymbol{\alpha}) + \boldsymbol{b}$. From above discussions, we present the (chosen-input) reverse subring VOLE (rsVOLE) functionality over $\mathtt{GR}(2^k, d)$ in Figure 21, where the receiver $P_{\mathcal{R}}$'s inputs are over $\mathbb{Z}_{2^k}$. Then in Figure 11, we present a semi-honest secure ϕVOLE construction over $\mathtt{GR}(2^k, d)$ in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$-hybrid model. To enable perfect simulation, we further require that γ_1 is invertible in $\mathtt{GR}(2^k, d)$. Such RMFE-basis $\gamma_1, ..., \gamma_m \in \mathtt{GR}(2^k, d)$ always exists, for an $(m, d; D)$-RMFE (ϕ, ψ) with $\phi(\mathbf{1}) = 1$. Intuitively, if $\boldsymbol{b}_1$ is uniformly random, then $\boldsymbol{b}$ is uniformly random as well. We have the following theorem (see proof in Appendix B.6).

Theorem 6. *Protocol $\Pi^{\mathtt{GR}(2^k,d)}_{\phi\text{VOLE}}$ realizes $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\phi\text{VOLE}}$ with semi-honest security.*

It is straightforward that the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$ functionality can be instantiated from OTs in a standard way (similar to the way that SPD$\mathbb{Z}_{2^k}$[11] implements VOLE over $\mathbb{Z}_{2^k}$), yielding the required VOLE protocol with semi-honest security (mk OTs involved in total). We present the OT-based protocol $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$ in Figure 22 and its security proof in Theorem 13. For such OT-based constructions, the malicious security can be naturally obtained if upgrading OTs to Correlated OTs (COTs) (we can use the COT construction in [23]), as there is no room for a malicious $P_{\mathcal{R}}$ to cheat and a malicious $P_{\mathcal{S}}$ can only cheat by providing inconsistent inputs to OTs. This leads to an efficient malicious secure ϕVOLE construction (the cost is even cheaper than constructing standard VOLE from COT), and further an efficient malicious secure NISC construction for computing BPs over $\mathbb{Z}_{2^k}$.

Protocol $\Pi^{\mathtt{GR}(2^k,d)}_{\phi\text{VOLE}}$

Parameterized by a Galois ring $\mathtt{GR}(2^k, d)$, length parameter l. Let (ϕ, ψ) be an $(m, d; 3)$-RMFE over $\mathbb{Z}_{2^k}$, and $\gamma_1, ..., \gamma_m \in \mathtt{GR}(2^k, d)$ be an RMFE-basis such that γ_1 is invertible. Suppose $P_{\mathcal{R}}$ has input $\boldsymbol{\alpha} \in \mathbb{Z}_{2^k}^m$, and $P_{\mathcal{S}}$ has input $\boldsymbol{a}, \boldsymbol{b} \in \mathtt{GR}(2^k, d)^l$.

1. For $i = 1, ..., m$, $P_{\mathcal{R}}$ sends $(i; \alpha_i)$ to $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$.
2. For $i = 2, ..., m$, $P_{\mathcal{S}}$ picks random $\boldsymbol{b}_i \in \mathtt{GR}(2^k, d)^l$. $P_{\mathcal{S}}$ sets $\boldsymbol{b}_1 = \gamma_1^{-1} \cdot (\boldsymbol{b} - \sum_{i=2}^{m} \boldsymbol{b}_i \cdot \gamma_i)$. $P_{\mathcal{S}}$ sends $(i; \boldsymbol{a}, \boldsymbol{b}_i; l)$ to $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$, for $i \in [m]$.
3. Upon receiving $(i; \boldsymbol{v}_i; l)$ from $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$, where $\boldsymbol{v}_i = \boldsymbol{a} \cdot \alpha_i + \boldsymbol{b}_i$, for $i \in [m]$, $P_{\mathcal{R}}$ computes $\boldsymbol{v} := \sum_{i=1}^{m} \boldsymbol{v}_i \cdot \gamma_i$.

Fig. 11: Protocol for ϕVOLE over $\mathtt{GR}(2^k, d)$ in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\text{rsVOLE}}$-hybrid model.

6 Acknowledgements

The work was supported in part by the National Key Research and Development Project under Grant 2022YFA1004900, in part by the National Natural Science Foundation of China under Grant 12031011.

References

1. Afshar, A., Mohassel, P., Pinkas, B., Riva, B.: Non-interactive secure computation based on cut-and-choose. In: EUROCRYPT 2014. LNCS, vol. 8441, pp. 387–404. Springer (2014)
2. Applebaum, B., Ishai, Y., Kushilevitz, E.: How to garble arithmetic circuits. SIAM J. Comput. **43**(2), 905–929 (2014)
3. Blum, A., Furst, M., Kearns, M., Lipton, R.J.: Cryptographic primitives based on hard learning problems. In: Advances in Cryptology—CRYPTO'93: 13th Annual International Cryptology Conference Santa Barbara, California, USA August 22–26, 1993 Proceedings 13. LNCS, vol. 773, pp. 278–291. Springer (1993)
4. Bootle, J., Chiesa, A., Guan, Z., Liu, S.: Linear-time probabilistic proofs with sublinear verification for algebraic automata over every field. IACR Cryptol. ePrint Arch. p. 1056 (2022), `https://eprint.iacr.org/2022/1056`
5. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y.: Compressing vector OLE. In: CCS 2018. pp. 896–912. ACM (2018)
6. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Rindal, P., Scholl, P.: Efficient two-round OT extension and silent non-interactive secure computation. In: CCS 2019. pp. 291–308. ACM (2019)
7. Cascudo, I., Cramer, R., Xing, C., Yuan, C.: Amortized complexity of information-theoretically secure MPC revisited. In: CRYPTO 2018. LNCS, vol. 10993, pp. 395–426. Springer (2018)
8. Cascudo, I., Giunta, E.: On interactive oracle proofs for boolean R1CS statements. In: Financial Cryptography and Data Security - 26th International Conference, FC 2022, Grenada, May 2-6, 2022, Revised Selected Papers. LNCS, vol. 13411, pp. 230–247. Springer (2022)
9. Cascudo, I., Gundersen, J.S.: A secret-sharing based MPC protocol for boolean circuits with good amortized complexity. In: TCC 2020. LNCS, vol. 12551, pp. 652–682. Springer (2020)
10. Chase, M., Dodis, Y., Ishai, Y., Kraschewski, D., Liu, T., Ostrovsky, R., Vaikuntanathan, V.: Reusable non-interactive secure computation. In: CRYPTO 2019. LNCS, vol. 11694, pp. 462–488. Springer (2019)
11. Cramer, R., Damgård, I., Escudero, D., Scholl, P., Xing, C.: Spd$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for dishonest majority. In: CRYPTO 2018. LNCS, vol. 10992, pp. 769–798. Springer (2018)
12. Cramer, R., Rambaud, M., Xing, C.: Asymptotically-good arithmetic secret sharing over $\mathbb{Z}/p^\ell\mathbb{Z}$ with strong multiplication and its applications to efficient MPC. In: CRYPTO 2021. LNCS, vol. 12827, pp. 656–686. Springer (2021)
13. Dittmer, S., Ishai, Y., Ostrovsky, R.: Line-point zero knowledge and its applications. In: 2nd Conference on Information-Theoretic Cryptography, ITC 2021, July 23-26, 2021, Virtual Conference. LIPIcs, vol. 199, pp. 5:1–5:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021)

14. Escudero, D., Liu, H., Xing, C., Yuan, C.: Degree-d reverse multiplication-friendly embeddings: Constructions and applications. IACR Cryptol. ePrint Arch. p. 173 (2023), `https://eprint.iacr.org/2023/173`
15. Escudero, D., Xing, C., Yuan, C.: More efficient dishonest majority secure computation over $\mathbb{Z}_{2^k}$ via galois rings. In: CRYPTO 2022. LNCS, vol. 13507, pp. 383–412. Springer (2022)
16. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009. pp. 169–178. ACM (2009)
17. Ishai, Y., Khurana, D., Sahai, A., Srinivasan, A.: Black-box reusable NISC with random oracles. In: EUROCRYPT 2023. LNCS, vol. 14005, pp. 68–97. Springer (2023)
18. Ishai, Y., Kushilevitz, E.: Perfect constant-round secure computation via perfect randomizing polynomials. In: ICALP 2002. LNCS, vol. 2380, pp. 244–256. Springer (2002)
19. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Prabhakaran, M., Sahai, A.: Efficient non-interactive secure computation. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 406–425. Springer (2011)
20. Lin, F., Xing, C., Yao, Y.: More efficient zero-knowledge protocols over $\mathbb{Z}_{2^k}$ via galois rings. IACR Cryptol. ePrint Arch. p. 150 (2023), `https://eprint.iacr.org/2023/150`
21. Mohassel, P., Rosulek, M.: Non-interactive secure 2pc in the offline/online and batch settings. In: EUROCRYPT 2017. LNCS, vol. 10212, pp. 425–455 (2017)
22. Ree, R.: Proof of a conjecture of s. chowla. Journal of Number Theory **3**(2), 210–212 (1971)
23. Scholl, P.: Extending oblivious transfer with low communication via key-homomorphic prfs. In: PKC 2018. LNCS, vol. 10769, pp. 554–583. Springer (2018)
24. Wan, Z.X.: Lectures on finite fields and Galois rings. World Scientific Publishing Company (2003)
25. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: 27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986. pp. 162–167. IEEE Computer Society (1986)

Supplementary Material

A More Preliminaries

Commitment Scheme. Informally, a commitment scheme is a two party protocol consisting of two algorithms, `Commit` and `Open`. In the commit phase of the protocol, the sender $P_{\mathcal{S}}$ invokes `Commit` to commit some value m, obtaining $[\![m]\!] \leftarrow \texttt{Commit}(m)$ as the result. Then he sends $[\![m]\!]$ to the receiver $P_{\mathcal{R}}$. Later on in the unveil phase, $P_{\mathcal{S}}$ is required to send m along with the unveil information $\texttt{unv} \coloneqq \texttt{Open}([\![m]\!])$ to $P_{\mathcal{R}}$ such that $P_{\mathcal{R}}$ can check whether m is a valid opening of $[\![m]\!]$. There are two security properties that a commitment scheme should satisfy; 1.*Hiding*: $P_{\mathcal{R}}$ can not learn anything about m from $[\![m]\!]$ in the commit phase, and 2.*Binding*: $P_{\mathcal{S}}$ can not provide a $m' \neq m$ and a $\texttt{unv}'$ such that $P_{\mathcal{R}}$ accepts m' as a valid opening of $[\![m]\!]$ in the unveil phase. A linearly homomorphic commitment scheme over some ring $\mathcal{R}$ additionally satisfies the *linear homomorphic* property: for $n > 0$, $a_i, m_i \in \mathcal{R}$, $i \in [n]$, $P_{\mathcal{R}}$ can check whether $m = \sum_{i\in[n]} a_i m_i$ is a valid opening of $\sum_{i\in[n]} a_i [\![m_i]\!]$, as long as given $m, a_i, [\![m_i]\!], i \in [n]$ and $\texttt{unv} \coloneqq \texttt{Open}(\sum_{i\in[n]} a_i [\![m_i]\!])$.
Zero-Knowledge Proof. The zero-knowledge proof functionality ($\mathcal{F}_{\mathrm{ZK}}$, Figure 12) for circuit satisfiability allows the prover $\mathcal{P}$ to prove knowledge of a witness $\boldsymbol{w}$ for some public circuit $\mathcal{C}$, i.e. $\mathcal{C}(\boldsymbol{w}) = 1$ without revealing any additional information to the verifier $\mathcal{V}$.

Functionality $\mathcal{F}_{\mathrm{ZK}}$
Parameterized by a ring $\mathcal{R}$, a witness size parameter n.

Input phase: Upon receiving $(sid; \texttt{input}, \boldsymbol{w})$ from a prover $\mathcal{P}$ with $\boldsymbol{w} \in \mathcal{R}^n$ and $(sid; \texttt{input})$ from a verifier $\mathcal{V}$, store $(sid; \boldsymbol{w})$ and ignore any further inputs from $\mathcal{P}$ and $\mathcal{V}$ with the same session identifier sid.
Prove phase: Upon receiving $(sid; \texttt{prove}, \mathcal{C})$ from $\mathcal{P}$ and $(sid; \texttt{verify}, \mathcal{C})$ from $\mathcal{V}$, verify that there are stored values $(sid; \boldsymbol{w})$ and $\mathcal{C}$ is a circuit over $\mathcal{R}$ with input size n; else ignore that message. Send $(sid; \texttt{true})$ to $\mathcal{V}$ if $\mathcal{C}(\boldsymbol{w}) = 1$ and $(sid; \texttt{false})$ otherwise.

Fig. 12: Zero-knowledge functionality for circuit satisfiability.

Functionality $\mathcal{F}_{\mathrm{EQ}}$

Input phase: Upon receiving $(sid; V_\mathcal{P})$ from a party $\mathcal{P}$ and $(sid; V_\mathcal{V})$ from a party $\mathcal{V}$, where $V_\mathcal{P}$, $V_\mathcal{V}$ are two strings, store $(sid; V_\mathcal{P}, V_\mathcal{V})$ and ignore any further inputs from $\mathcal{P}$ and $\mathcal{V}$ with the same session identifier *sid*.

Send phase: Upon receiving $(sid; \texttt{send})$ from $\mathcal{P}$, send $(sid; V_\mathcal{P}, V_\mathcal{P} \overset{?}{=} V_\mathcal{V})$ to $\mathcal{V}$. Upon receiving $(sid; \texttt{send})$ from an honest $\mathcal{V}$, if $V_\mathcal{P} = V_\mathcal{V}$, send $(sid; \texttt{true})$ to $\mathcal{P}$, and if $V_\mathcal{P} \neq V_\mathcal{V}$, send $(sid; \texttt{abort})$ to $\mathcal{P}$. If receive $(sid; \texttt{continue})$ from a corrupted $\mathcal{V}$, send $(sid; V_\mathcal{P} \overset{?}{=} V_\mathcal{V})$ to $\mathcal{P}$. If receive $(sid; \texttt{abort})$ from a corrupted $\mathcal{V}$, send $(sid; \texttt{abort})$ to $\mathcal{P}$.

Fig. 13: Ideal functionality for equality tests.

Functionality $\mathcal{F}_{\mathrm{OT}}$

Parameterized by a ring $\mathcal{R}$.

Input phase: Upon receiving $(sid; \texttt{input}, m_0, m_1)$ from $P_\mathcal{S}$ and $(sid; \texttt{input}, b)$ from $P_\mathcal{S}$, where $m_0, m_1 \in \mathcal{R}$ and $b \in \{0, 1\}$, store $(sid; m_b)$ and ignore any further inputs from $P_\mathcal{S}$ and $P_\mathcal{R}$ with the same session identifier *sid*.

Send phase: Upon receiving $(sid; \texttt{send})$ from $P_\mathcal{S}$, verify that there are stored inputs $(sid; m_b)$; else ignore that message. Next, send $(sid; m_b)$ to $P_\mathcal{R}$.

Fig. 14: Ideal functionality for oblivious transfer.

B Deferred Proofs & Protocols

B.1 NIZK

We present the subprotocol $\Pi^{q,t}_{\text{NIZK}}$ for proving circuit satisfiability over $\texttt{GR}(2^k, d)$ in Figure 15. We remark that though the presentation of our protocol is different from [13], it is essentially an LPZK-NIZK in Galois ring analogue. On input $[\boldsymbol{w}]_\gamma$, the subprotocol $\Pi^{q,t}_{\text{NIZK}}$ communicates 3 Galois ring elements per multiplication gate.

Protocol $\Pi^{q,t}_{\text{NIZK}}$

The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ have agreed on a circuit $\mathcal{C}$ over $\texttt{GR}(2^k, d)$ with q inputs and t multiplication gates, and $\mathcal{P}$ holds a witness $\boldsymbol{w} \in \texttt{GR}(2^k, d)^q$ such that $\mathcal{C}(\boldsymbol{w}) = 1$.

1. **Offline:**
   (a) The two parties invoke the **Setup phase** of $\mathcal{F}^{\texttt{GR}(2^k,d)}_{\text{VOLE}}$ (where $\mathcal{P}$ acts as the sender, and $\mathcal{V}$ acts as the receiver), then $\mathcal{V}$ receives $\gamma \in \texttt{GR}(2^k, d)$.
   (b) The two parties invoke the **Send phases** of $\mathcal{F}^{\texttt{GR}(2^k,d)}_{\text{VOLE}}$, then $\mathcal{P}$ receives $\boldsymbol{\nu}, \boldsymbol{\pi} \in \texttt{GR}(2^k, d)^t, \boldsymbol{M} \in \texttt{GR}(2^k, d)^{2t}$, and $\mathcal{V}$ receives $\boldsymbol{K} \in \texttt{GR}(2^k, d)^{2t}$ such that $K_{\nu_i} = M_{\nu_i} + \nu_i \cdot \gamma$ and $K_{\pi_i} = M_{\pi_i} + \pi_i \cdot \gamma$ for $i \in [t]$. Namely, they obtain random MACs $[\boldsymbol{\nu}]_\gamma, [\boldsymbol{\pi}]_\gamma$.
2. **Online:** On input $[\boldsymbol{w}]_\gamma$.
   (a) For each gate $(x, y, z, T) \in \mathcal{C}$, in a topological order:
      – If T=$\texttt{Add}$, then $\mathcal{P}$ and $\mathcal{V}$ locally compute $[w_z]_\gamma := [w_x]_\gamma + [w_y]_\gamma$.
      – If T=$\texttt{Mul}$ and this is the i-th multiplication gate, then $\mathcal{P}$ sends $d_i := w_x \cdot w_y - \nu_i$ to $\mathcal{V}$, and they locally compute $[w_z]_\gamma := [\nu_i]_\gamma + d_i$.
   (b) For the i-th multiplication gate, the two parties hold $([w_x]_\gamma, [w_y]_\gamma, [w_z]_\gamma)$ with $K_{w_j} = M_{w_j} + w_j \cdot \gamma$ for $j \in \{x, y, z\}$.
      – The prover $\mathcal{P}$ computes $A_{0,i} := M_{w_x} \cdot M_{w_y} \in \texttt{GR}(2^k, d)$ and $A_{1,i} := w_x \cdot M_{w_y} + w_y \cdot M_{w_x} - M_{w_z} \in \texttt{GR}(2^k, d)$.
      – The verifier $\mathcal{V}$ computes $B_i := K_{w_x} \cdot K_{w_y} - K_{w_z} \cdot \gamma \in \texttt{GR}(2^k, d)$.
   (c) The two parties do the following check:
      – For $i \in [t]$, $\mathcal{P}$ computes $X_i := A_{0,i} + M_{\pi_i} \in \texttt{GR}(2^k, d)$ and $Y_i := A_{1,i} + \pi_i \in \texttt{GR}(2^k, d)$, and sends (X_i, Y_i) to $\mathcal{V}$.
      – For $i \in [t]$, $\mathcal{V}$ computes $Z_i := B_i + K_{\pi_i} \in \texttt{GR}(2^k, d)$, and checks whether $Z_i = X_i + Y_i \cdot \gamma$ holds. If the check fails, $\mathcal{V}$ outputs $\texttt{false}$ and aborts.
   (d) For the single output wire ω_h, they hold $[\omega_h]_\gamma$.
      – The prover $\mathcal{P}$ sends M_{ω_h} to $\mathcal{V}$.
      – The verifier $\mathcal{V}$ checks whether $K_{\omega_h} = M_{\omega_h} + \gamma$. If the check fails, $\mathcal{V}$ outputs $\texttt{false}$. Otherwise, $\mathcal{V}$ outputs $\texttt{true}$.

Fig. 15: Zero-knowledge protocol for circuit satisfiability over $\texttt{GR}(2^k, d)$ in the $\mathcal{F}^{\texttt{GR}(2^k,d)}_{\text{VOLE}}$-hybrid model.

Theorem 7 (Proposition 5, restated). *Protocol $\Pi_{\text{NIZK}}^{q,t}$ realizes $\mathcal{F}_{\text{ZK}}$ in the $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model with soundness error $1/2^{d-1}$ and statistical security.*

Proof. *We divide our proof into two parts. We first consider $\mathcal{P}$ is corrupted, then we consider $\mathcal{V}$ is corrupted. In each case, we build a* PPT *simulator $\mathcal{S}$ to interact with the corrupted party in the ideal world, which can read the corrupted party's inputs to functionalities $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$.*

Corrupted $\mathcal{P}$: *The simulator $\mathcal{S}$ interacts with $\mathcal{A}$ as follows:*

1. *The simulator $\mathcal{S}$ samples $\gamma \xleftarrow{\$} \mathtt{GR}(2^k,d)$ and records $\boldsymbol{\nu}, \boldsymbol{\pi} \in \mathtt{GR}(2^k,d)^t$ and $\boldsymbol{M} \in \mathtt{GR}(2^k,d)^{2t}$ that $\mathcal{A}$ sends to $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$. Thus, $\mathcal{S}$ can immediately obtain the MACs $[\boldsymbol{\mu}]_\gamma, [\boldsymbol{\pi}]_\gamma$. Similarly, $\mathcal{S}$ can obtain the input MACs $[\boldsymbol{w}]_\gamma$.*
2. *The simulator $\mathcal{S}$ runs the rest of the protocol as an honest verifier, using the MACs generated in previous steps. If the honest verifier outputs* `true`, *$\mathcal{S}$ sends $\boldsymbol{w}$ and the circuit $\mathcal{C}$ to $\mathcal{F}_{\text{ZK}}$. Otherwise, $\mathcal{S}$ sends $\boldsymbol{w} := \perp$ and $\mathcal{C}$ to $\mathcal{F}_{\text{ZK}}$ and aborts.*

From the simulation, we can see that $\mathcal{S}$ behaves like an honest verifier towards $\mathcal{A}$, therefore, the environment $\mathcal{Z}$ can not distinguish the ideal simulation and real execution from the adversary $\mathcal{A}$'s view. Note that $\mathcal{Z}$ has access to the output of the honest party, the situation remains to be considered is that honest verifier $\mathcal{V}$ accepts the proof while $\mathcal{A}$ does not hold the witnesses. Below we show the probability that $\mathcal{V}$ accepts a proof of wrong statements (i.e. the soundness error) is upper bounded by $1/2^{d-2}$.

First we prove that all the values on the wires in the circuit are correct. It can be immediately obtained that the values associated with input wires and the output wires of `Add` *gates are computed correctly, since they are computed locally. Thus, we need to consider the correctness of values on the output wires of* `Mul` *gates, which is guaranteed by the correctness of d_i, for all $i \in [t]$ in our protocol $\Pi_{\text{ZK}}^{q,t}$. Consider that some of components of $\boldsymbol{d}$ are incorrect, e.g. there is an error in the i-th* `Mul` *gate. Let $d_i := w_x \cdot w_y - \nu_i + e_i$, where $e_i \in \mathtt{GR}(2^k,d)$. Thus we have that*

$$\begin{aligned} K_{\hat{w}_z} :&= K_{\nu_i} + \gamma \cdot d_i = K_{\nu_i} + \gamma \cdot (w_x \cdot w_y - \nu_i + e_i) \\ &= M_{\nu_i} + \gamma \cdot \nu_i + \gamma \cdot (w_x \cdot w_y - \nu_i + e_i) \\ &= M_{\hat{w}_z} + \gamma \cdot (w_x \cdot w_y) + \gamma \cdot e_i, \end{aligned}$$

and

$$\begin{aligned} B_i :&= K_{w_x} \cdot K_{w_y} - \gamma \cdot K_{\hat{w}_z} \\ &= (M_{w_x} + \gamma \cdot w_x) \cdot (M_{w_y} + \gamma \cdot w_x) - \gamma \cdot (M_{\hat{w}_z} + \gamma \cdot (w_x \cdot w_y) + \gamma \cdot e_i) \\ &= (M_{w_x} \cdot M_{w_y}) + \gamma \cdot (w_x \cdot M_{w_y} + w_y \cdot M_{w_x} - M_{\hat{w}_z}) - \gamma^2 \cdot e_i \\ &= A_{0,i} + \gamma \cdot A_{1,i} - \gamma^2 \cdot e_i, \end{aligned}$$

which leads to

$$
\begin{aligned}
Z_i :&= B_i + K_{\pi_i} \\
&= X_i + \gamma \cdot Y_i - \gamma^2 \cdot e_i,
\end{aligned}
$$

for all $i \in [t]$. Assume $\mathcal{A}$ sends $X_i' = X + e_{X_i}$ and $Y_i' = Y + e_{Y_i}$ to the honest verifier, where $e_{X_i}, e_{Y_i} \in \mathtt{GR}(2^k, d)$, for all $i \in [t]$. $\mathcal{V}$ accepts if and only if

$$Z_i = X_i' + \gamma \cdot Y_i' \iff 0 = e_{X_i} + \gamma \cdot e_{Y_i} + \gamma^2 \cdot e_i,$$

holds for all $i \in [t]$. By Lemma 1, we obtain that the above equations hold with probability at most $2^{-(d-1)}$.

Finally, we show that if $\mathcal{C}(\boldsymbol{w}) = 0$, and all the values on the wires in the circuit are correct, the probability that $\mathcal{A}$ successfully provides a $M_{w_h}' := M_{w_h} + e_{w_h}$ such that $K_{w_h} = M_{w_h}' + \gamma$ is upper bounded by 2^{-kd}. The honest verifier accepts if and only if

$$K_{w_h} = M_{w_h}' + \gamma \iff 0 = e_{w_h} + \gamma,$$

which holds for a random $\gamma \in \mathtt{GR}(2^k, d)$ with probability at most $1/2^{-kd}$.

Thus, the overall soundness error is bounded by $2^{-(d-1)} + 2^{-kd} \approx 2^{-(d-1)}$. Namely, a PPT $\mathcal{Z}$ can distinguish between the real world and the ideal world with advantage approximately at most $2^{-(d-1)}$.

***Corrupted $\mathcal{V}$:** If $\mathcal{S}$ receives `false` from $\mathcal{F}_{\mathrm{ZK}}$, then it just aborts. Otherwise, $\mathcal{S}$ interacts with $\mathcal{A}$ as follows:*

1. *In the offline phase: $\mathcal{S}$ records $\gamma \in \mathtt{GR}(2^k, d)$ that $\mathcal{A}$ sends to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$, also, $\mathcal{S}$ records $K_{\nu_i}, K_{\pi_i} \in \mathtt{GR}(2^k, d)$ for $i \in [t]$ that $\mathcal{A}$ sends to $\mathcal{F}_{\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$. Similarly, $\mathcal{S}$ records K_{w_i} for $i \in [q]$ for the input MACs.*
2. *The simulator $\mathcal{S}$ samples $\boldsymbol{\nu}, \boldsymbol{\pi} \xleftarrow{\$} \mathtt{GR}(2^k, d)^t$, and $\boldsymbol{w} \xleftarrow{\$} \mathtt{GR}(2^k, d)^q$.*
3. *For each gate $(x, y, z, T) \in \mathcal{C}$, in a topological order:*
   - *If T=`Add`, $\mathcal{S}$ computes $K_{w_z} := K_{w_x} + K_{w_y}$ as the honest verifier would do, and sets $w_z := w_x + w_y$.*
   - *If T=`Mul`, and this is the i-th multiplication gate, then $\mathcal{S}$ sends $d_i := w_x \cdot w_y - \nu_i$ to $\mathcal{A}$. The simulator $\mathcal{S}$ computes $K_{w_z} := K_{\nu_i} + \gamma \cdot d_i$, and $B_i := K_{w_x} \cdot K_{w_y} - \gamma \cdot K_{w_z}$ as the honest verifier would do, and sets $w_z := w_x \cdot w_y$.*
4. *For $i \in [t]$, $\mathcal{S}$ computes $Z_i := B_i + K_{\pi_i} \in \mathtt{GR}(2^k, d)$, then $\mathcal{S}$ samples $Y_i \xleftarrow{\$} \mathtt{GR}(2^k, d)$ and sets $X_i := Z_i - \gamma \cdot Y_i$. The simulator $\mathcal{S}$ sends $\{(X_i, Y_i)\}_{i \in [t]}$ to $\mathcal{A}$.*
5. *For the single output wire w_h, $\mathcal{S}$ already holds K_{w_h}. Finally, $\mathcal{S}$ computes $M_{w_h} := K_{w_h} - \gamma$, and sends M_{w_h} to $\mathcal{A}$.*

It can be observed that $\{(X_i, Y_i)\}_{i \in [t]}$ provided by the honest prover are distributed uniformly at random due to the masks M_{π_i}, π_i, respectively, under the equality constraints $Z_i = X_i + \gamma \cdot Y_i$, $i \in [n]$. Besides, M_{w_h} is identically distributed in both real execution and ideal simulation. Therefore, the simulation is perfect. This completes the proof. □

B.2 eVOLE & cVOLE

Theorem 8 (Proposition 4, restated). *$\Pi_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model with reusable malicious security.*

Proof. *If the receiver is corrupted, the simulator $\mathcal{S}$ emulates the $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$ functionality, and α_1, α_2 from the adversary. $\mathcal{S}$ forwards α_1, α_2 to the ideal functionality $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$, and receives $\boldsymbol{v}_1 \in \mathtt{GR}(2^k,d)^{l_1}, \boldsymbol{v}_2 \in \mathtt{GR}(2^k,d)^{l_2}$. $\mathcal{S}$ samples random $\hat{\boldsymbol{v}}_1 \in \mathtt{GR}(2^k,d)^{l_1+1}, \hat{\boldsymbol{v}}_2 \in \mathtt{GR}(2^k,d)^{l_2+1}$ and sends them to the adversary. For $t = 1, 2$, $\mathcal{S}$ samples random $\boldsymbol{u}_t$ from $\mathtt{GR}(2^k,d)^{l_t}$, and computes $\boldsymbol{w}_t := \boldsymbol{v}_t - \boldsymbol{u}_t \cdot \alpha_t - \hat{\boldsymbol{v}}_t$. For $t = 1, 2$, $\mathcal{S}$ sends $\boldsymbol{u}_t, \boldsymbol{w}_t$ to the adversary. $\mathcal{S}$ forwards the verification command $(\textbf{Verify}\dagger, i, j)$ received from the adversary to $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$. $\mathcal{S}$ samples random u_{1,l_1+1}, u_{2,l_2+1}. If $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ returns $\perp$, $\mathcal{S}$ samples random $\hat{b}$, otherwise, $\mathcal{S}$ computes $\hat{b} := \alpha_2 \cdot v_{1,i} - \alpha_1 \cdot v_{2,j} + \hat{v}_{1,l_1+1} + \alpha_1 \cdot u_{1,l_1+1} - \hat{v}_{2,l_2+1} - \alpha_2 \cdot u_{2,l_2+1}$. $\mathcal{S}$ sends $u_{1,l_1+1}, u_{2,l_2+1}, \hat{b}$ to the adversary. Similarly, $\mathcal{S}$ forwards the verification command $(\textbf{Verify}\ddagger, i, j)$ received from the adversary to $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$. $\mathcal{S}$ samples random w_{1,l_1+1}, u_{2,l_2+1}. If $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ returns $\perp$, $\mathcal{S}$ samples random $\hat{b}$, otherwise, $\mathcal{S}$ computes $\hat{b}$ such that $\alpha_1 \cdot \hat{b} := v_{2,j} - \hat{v}_{1,l_1+1} - w_{1,l_1+1} - v_{1,i} \cdot \alpha_2 + (\hat{v}_{2,l_2+1} + u_{2,l_2+1} \cdot \alpha_2) \cdot \alpha_1$. $\mathcal{S}$ sends $w_{1,l_1+1}, u_{2,l_2+1}, \hat{b}$ to the adversary. One can check that the simulation is perfect.*

For reusable security in the malicious sender setting, the simulator $\mathcal{S}$ emulates the $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$ functionality and extracts $\mathcal{A}$'s inputs from $\mathcal{A}$' messages, then $\mathcal{S}$ forwards the inputs to the $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ functionality. As discussed in Section 3, the soundness error here is upper bounded by $1/2^{d-1}$. This completes the proof. □

Theorem 9 (Corollary 2, restated). *$\Pi_{\text{cVOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\text{cVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $(\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}, \mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)})$-hybrid model.*

Proof. *If the receiver is corrupted. We construct a simulator $Sim_{\mathcal{R}}$ that invokes a simulator $Sim_{\mathcal{R}}^{\text{NIZK}}$ for $\Pi_{\text{NIZK}}^{q,t}$. The simulator $Sim_{\mathcal{R}}$ emulates the ideal functionality $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$, and records the adversary's inputs $(\alpha'_1, ..., \alpha'_n, \beta, \gamma)$. Then $Sim_{\mathcal{R}}$ computes $\alpha_i := \alpha'_i - \beta$, for $i \in [n]$, and sends them to the ideal functionality $\mathcal{F}_{\text{cVOLE}}^{\mathtt{GR}(2^k,d)}$, which returns $\boldsymbol{v}_1, ..., \boldsymbol{v}_n$ to $Sim_{\mathcal{R}}$. For $i \in [n]$, $Sim_{\mathcal{R}}$ samples $\hat{\boldsymbol{v}}_i \xleftarrow{\$} \mathtt{GR}(2^k,d)^{l_i}$, and $\hat{v}_{n+2} \xleftarrow{\$} \mathtt{GR}(2^k,d)^{q+q_b}$. Besides, $Sim_{\mathcal{R}}$ sets $\hat{\boldsymbol{v}}_{n+1} := \hat{\boldsymbol{v}}_1 - \boldsymbol{v}_1 \parallel ... \parallel \hat{\boldsymbol{v}}_{n+1} - \boldsymbol{v}_n$. For $i \in [n+2]$, $Sim_{\mathcal{R}}$ sends $\hat{\boldsymbol{v}}_i$ to the adversary. For the adversary's eVOLE queries, $Sim_{\mathcal{R}}$ responds with "yes". Finally, $Sim_{\mathcal{R}}$ invokes $Sim_{\mathcal{R}}^{\text{NIZK}}$. The indistinguishability is straightforward.*

*If the sender is corrupted. We construct a simulator $Sim_{\mathcal{S}}$ that invokes a simulator $Sim_{\mathcal{S}}^{\text{NIZK}}$ for $\Pi_{\text{NIZK}}^{q,t}$. The simulator $Sim_{\mathcal{S}}$ emulates the ideal functionality $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$, and records the adversary's inputs $(\boldsymbol{a}_i, \boldsymbol{b}_i)$, for $i \in [n+2]$. $Sim_{\mathcal{S}}$ checks equality constraints that a honest receiver would check (Also, $Sim_{\mathcal{S}}$ emulates the **Verify Phase** of $\mathcal{F}_{\text{eVOLE}}^{\mathtt{GR}(2^k,d)}$ and invokes $Sim_{\mathcal{S}}^{\text{NIZK}}$.). If any of the equality check* *fails, $Sim_\mathcal{S}$ sends* **`aborting`** *to $\mathcal{F}_{\text{cVOLE}}^{\mathtt{GR}(2^k,d)}$. Finally, $Sim_\mathcal{S}$ sends $(\boldsymbol{a}_i, \boldsymbol{b}_i)$, for $i \in [n]$ to $\mathcal{F}_{\text{cVOLE}}^{\mathtt{GR}(2^k,d)}$. The environment can distinguish ideal world and real execution with probability at most the soundness error of $\mathit{\Pi}_{\text{NIZK}}^{q,t}$.* □

B.3 Re-embedding VOLE

We remark that our $\mathit{\Pi}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ protocol (Figure 16) allows to re-embed a subset J of the $\boldsymbol{a}$ entries. Also, $\mathit{\Pi}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ can be made non-interactive via Fiat-Shamir transform, where $(l+r)$ VOLE correlations are obtained and $|J| + 3r$ Galois ring elements are communicated. We have the following theorem.

Protocol $\mathit{\Pi}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$

Parameterized by a Galois ring $\mathtt{GR}(2^k, d)$, length parameters $l \in \mathbb{N}$. Let (ϕ, ψ) be an $(m, d; D)$-RMFE over $\mathbb{Z}_{2^k}$, and $\tau := \phi \circ \psi$. Let $J \subseteq [l]$, and denote $J \cup [l+1, l+r]$ by $\hat{J}$.

1. $P_\mathcal{S}$ and $P_\mathcal{R}$ invoke the **Setup phase** of $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$, then $P_\mathcal{R}$ receives $\alpha \in \mathtt{GR}(2^k, d)$.
2. $P_\mathcal{S}$ and $P_\mathcal{R}$ invoke the **Send phases** of $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$, then $P_\mathcal{S}$ receives $\boldsymbol{a}, \boldsymbol{b} \in \mathtt{GR}(2^k, d)^{l+r}$, and $P_\mathcal{R}$ receives $\boldsymbol{v}$ such that $\boldsymbol{v} = \boldsymbol{a} \cdot \alpha + \boldsymbol{b}$.
3. **Deliver:**
   (a) Let $\boldsymbol{\eta} := \mathbf{0} \in \mathtt{GR}(2^k, d)^{l+r}$. $P_\mathcal{S}$ resets $\boldsymbol{\eta}|_{\hat{J}} := (\tau(\boldsymbol{a}) - \boldsymbol{a})|_{\hat{J}}$, and sends $\boldsymbol{\eta}$ to $P_\mathcal{R}$. If $\boldsymbol{\eta}|_{\hat{J}} \notin \mathtt{Ker}(\psi)^{|J|+r}$, $P_\mathcal{R}$ aborts.
   (b) $P_\mathcal{R}$ samples $\boldsymbol{\chi}^1, ..., \boldsymbol{\chi}^r \xleftarrow{\$} \mathbb{Z}_{2^k}^l$, and sends them to $P_\mathcal{S}$.
   (c) For $i \in [r]$, $P_\mathcal{S}$ computes $x_i := a_{l+i} + \sum_{j \in J} \chi_j^i \cdot a_j$ and $y_i := \tau(a_{l+i}) + \sum_{j \in J} \chi_j^i \cdot \tau(a_j)$. $P_\mathcal{S}$ sends $\boldsymbol{x}, \boldsymbol{y}$ to $P_\mathcal{R}$. $P_\mathcal{S}$ computes $z_i := b_{l+i} + \sum_{j \in J} \chi_j^i \cdot b_j$ for $i \in [r]$.
   (d) $P_\mathcal{R}$ checks $y_i - x_i = \eta_{l+i} + \sum_{j \in J} \chi_j^i \cdot \eta_j$, for $i \in [r]$ and $\boldsymbol{y} \in \mathtt{Im}(\phi)^r$. If the check fails, $P_\mathcal{R}$ aborts. $P_\mathcal{R}$ computes $\hat{z}_i := v_{l+i} + \sum_{j \in J} \chi_j^i \cdot v_j - x_i \cdot \alpha$ for $i \in [r]$.
   (e) $P_\mathcal{R}$ sends $\boldsymbol{z}$ to $\mathcal{F}_{\text{EQ}}$ as $\mathcal{V}$. $P_\mathcal{S}$ sends $\hat{\boldsymbol{z}}$ to $\mathcal{F}_{\text{EQ}}$ as $\mathcal{P}$. $P_\mathcal{R}$ receives $\hat{\boldsymbol{z}}$. $P_\mathcal{S}$ and $P_\mathcal{R}$ abort if the equality test fails.
4. **Output:** $P_\mathcal{S}$ outputs $\boldsymbol{a}|_{[l]}, \boldsymbol{b}|_{[l]}$ and $P_\mathcal{R}$ outputs $\boldsymbol{v}|_{[l]}, \boldsymbol{\eta}|_J$.

Fig. 16: Protocol for re-embedding VOLE over $\mathtt{GR}(2^k, d)$ in the $(\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}, \mathcal{F}_{\text{EQ}})$-hybrid model.

Theorem 10 (Adapted from [20]). *$\mathit{\Pi}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ UC-realizes $\mathcal{F}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$ in the $(\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}, \mathcal{F}_{\text{EQ}})$-hybrid model. In particular, no* PPT *environment $\mathcal{Z}$ can* *distinguish the real world execution from the ideal world simulation except with advantage at most* $2^{-r} + 2^{-d}$.

Proof. We divide our proof into two parts. First, we consider $P_{\mathcal{S}}$ *is corrupted and construct a* PPT *simulator* $Sim_{\mathcal{S}}$*, then we consider* $P_{\mathcal{R}}$ *is corrupted and build a* PPT *simulator* $Sim_{\mathcal{R}}$ *as well. Both Simulators interact with the corrupted party in the ideal world and can read the corrupted party's inputs to functionalities* $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}, \mathcal{F}_{\text{EQ}}$.

Corrupted $P_{\mathcal{S}}$***:*** $Sim_{\mathcal{S}}$ *acts as follows:*

1. $Sim_{\mathcal{S}}$ *reads* $\boldsymbol{a}, \boldsymbol{b} \in \mathtt{GR}(2^k, d)^{l+r}$ *that* $\mathcal{A}$ *sends to* $\mathcal{F}_{\text{VOLE}}^{\mathtt{GR}(2^k,d)}$.
2. *Upon receiving* $\boldsymbol{\eta} \in \mathtt{GR}(2^k, d)^{l+r}$ *from* $\mathcal{A}$*, if* $\boldsymbol{\eta}|_{\hat{\jmath}} \notin \mathtt{Ker}(\psi)^{|J|+r}$*,* $Sim_{\mathcal{S}}$ *aborts.*
3. $Sim_{\mathcal{S}}$ *samples* $\boldsymbol{\chi}^1, ..., \boldsymbol{\chi}^r \xleftarrow{\$} \mathbb{Z}_{2^k}^l$*, and sends them to* $\mathcal{A}$.
4. *Upon receiving* $\boldsymbol{x}, \boldsymbol{y} \in \mathtt{GR}(2^k, d)^r$ *from* $\mathcal{A}$*. If* $y_i - x_i \neq \eta_{l+i} + \sum_{j \in J} \chi_j^i \cdot \eta_j$*, for some* $i \in [r]$*, or* $\boldsymbol{y} \notin \mathtt{Im}(\phi)^r$*,* $Sim_{\mathcal{S}}$ *aborts.*
5. $Sim_{\mathcal{S}}$ *reads* $\hat{z}_i, i \in [r]$ *that* $\mathcal{A}$ *sends to* $\mathcal{F}_{\text{EQ}}$*.* $Sim_{\mathcal{S}}$ *computes* $z_i := b_{l+i} + \sum_{j \in J} \chi_j^i \cdot b_j$*, for* $i \in [r]$*. If both* $\boldsymbol{\eta}|_{\hat{\jmath}} = (\tau(\boldsymbol{a}) - \boldsymbol{a})|_{\hat{\jmath}}$ *and* $\hat{\boldsymbol{z}} = \boldsymbol{z}$ *hold,* $Sim_{\mathcal{S}}$ *sends* `true` *to* $\mathcal{A}$ *(emulating* $\mathcal{F}_{\text{EQ}}$*), and sends* $(\boldsymbol{a}|_{[l]}, \boldsymbol{b}|_{[l]})$ *to* $\mathcal{F}_{\text{embVOLE}}^{\mathtt{GR}(2^k,d)}$*. Otherwise,* $Sim_{\mathcal{S}}$ *sends* `false` *to* $\mathcal{A}$ *and aborts.*

It can be observed that when $Sim_{\mathcal{S}}$ *aborts in step 2 or step 4 of the simulation, the honest* $P_{\mathcal{R}}$ *aborts in corresponding step of the protocol as well. Besides,* $\boldsymbol{\chi}^1, ..., \boldsymbol{\chi}^r$ *are sampled in the same way of the ideal simulation and real execution. Therefore, it remains to consider the simulation for* $\mathcal{F}_{\text{EQ}}$*. Basically,* $\mathcal{A}$ *can cheat by sending* $\boldsymbol{\eta}|_{\hat{\jmath}} \in \mathtt{Ker}(\psi)^{|J|+r}$*, but* $\boldsymbol{\eta}|_{\hat{\jmath}} \neq (\tau(\boldsymbol{a}) - \boldsymbol{a})|_{\hat{\jmath}}$*. Let* $\boldsymbol{\eta}|_{\hat{\jmath}} = (\tau(\boldsymbol{a}) - \boldsymbol{a})|_{\hat{\jmath}} + \boldsymbol{\varepsilon}$*, where* $\boldsymbol{\varepsilon} \in \mathtt{Ker}(\psi)^{|J|+r}$*. We have that*

$$\boldsymbol{\eta}|_{\hat{\jmath}} = \tau(\boldsymbol{a}|_{\hat{\jmath}}) - (\boldsymbol{a}|_{\hat{\jmath}} - \boldsymbol{\varepsilon}) = \tau(\boldsymbol{a}|_{\hat{\jmath}} - \boldsymbol{\varepsilon}) - (\boldsymbol{a}|_{\hat{\jmath}} - \boldsymbol{\varepsilon}).$$

Therefore, $\mathcal{A}$ *can set*

$$x_i^* := (a_{l+i} - \varepsilon_{l+i}) + \sum_{j \in J} \chi_j^i \cdot (a_j - \varepsilon_j),$$

and

$$y_i^* := \tau(a_{l+i} - \varepsilon_{l+i}) + \sum_{j \in J} \chi_j^i \cdot \tau(a_j - \varepsilon_j) = \tau(a_{l+i}) + \sum_{j \in J} \chi_j^i \cdot \tau(a_j) = y_i,$$

for $i \in [r]$*. These* x_i^*, y_i^* *would pass the check of honest* $P_{\mathcal{R}}$*. Now in real protocol, we have that*

$$\begin{aligned}
z_i &= v_{l+i} + \sum_{j \in J} \chi_j^i \cdot v_j - \alpha \cdot x_i^* \\
&= (b_{l+i} + \alpha a_{l+i}) + \sum_{j \in J} \chi_j^i (b_j + \alpha a_j) - \alpha (a_{l+i} - \varepsilon_{l+i} + \sum_{j \in J} \chi_j^i (a_j - \varepsilon_j)) \\
&= b_{l+i} + \sum_{j \in J} \chi_j^i \cdot b_j + \alpha \cdot (\varepsilon_{l+i} + \sum_{j \in J} \chi_j^i \cdot \varepsilon_j) \\
&= \hat{z}_i + \alpha \cdot (\varepsilon_{l+i} + \sum_{j \in J} \chi_j^i \cdot \varepsilon_j).
\end{aligned}$$

Thus, $\mathcal{F}_{\text{EQ}}$ returns `true` *if and only if*

$$\alpha \cdot (\varepsilon_{l+i} + \sum_{j\in J} \chi_j^i \cdot \varepsilon_j) = 0,$$

for all $i \in [r]$. If $\varepsilon_{l+i} + \sum_{i\in J} \chi_j^i \cdot \varepsilon_j \neq 0$, from lemma 1, the above equality holds with probability at most 2^{-d}. Since $\chi_j^i \in \mathbb{Z}_{2^k}$, for $j \in [l]$, $\varepsilon_{l+i} + \sum_{j\in J} \chi_j^i \cdot \varepsilon_j = 0$ holds with probability at most $1/2$. Combining together, $\mathcal{F}_{\text{EQ}}$ returns `true` *with probability at most $2^{-r} + 2^{-d}$ in real execution if $\boldsymbol{\eta}|_{\hat{J}} \neq (\tau(\boldsymbol{a}) - \boldsymbol{a})|_{\hat{J}}$, while in simulation, this will leads to abort. Note that if $\boldsymbol{\eta}|_{\hat{J}}$ is correct, the outputs of honest $P_{\mathcal{R}}$ are computed in the same way in two worlds. Therefore, environment $\mathcal{Z}$ can distinguish the ideal simulation and real execution with advantage at most $2^{-r} + 2^{-d}$.*

***Corrupted** $P_{\mathcal{R}}$**:** $Sim_{\mathcal{R}}$ does as follows:*

1. *$Sim_{\mathcal{R}}$ reads $\alpha \in \texttt{GR}(2^k, d)$ that $\mathcal{A}$ sends to $\mathcal{F}_{\text{VOLE}}^{\texttt{GR}(2^k,d)}$ in the **Setup phase**. $Sim_{\mathcal{R}}$ then sends α to $\mathcal{F}_{\text{embVOLE}}^{\texttt{GR}(2^k,d)}$.*
2. *$Sim_{\mathcal{R}}$ records $\boldsymbol{v} \in \texttt{GR}(2^k, d)^{l+r}$ sent by $\mathcal{A}$. Upon receiving $\boldsymbol{\eta} \in \texttt{Ker}(\psi)^l$ from $\mathcal{F}_{\text{embVOLE}}^{\texttt{GR}(2^k,d)}$, $Sim_{\mathcal{R}}$ samples $\boldsymbol{\eta}' \xleftarrow{\$} \texttt{Ker}(\psi)^r$ and sends $\hat{\boldsymbol{\eta}} := (\boldsymbol{\eta}, \boldsymbol{\eta}') \in \texttt{Ker}(\psi)^{l+r}$ to $\mathcal{A}$.*
3. *$Sim_{\mathcal{R}}$ sets $\boldsymbol{v}' := \boldsymbol{v} + \alpha \cdot \hat{\boldsymbol{\eta}}$. Upon receiving $\boldsymbol{\chi}^i \in \mathbb{Z}_{2^k}^l, i \in [r]$ from $\mathcal{A}$, $Sim_{\mathcal{R}}$ samples $\boldsymbol{y} \xleftarrow{\$} \texttt{Im}(\phi)^r$, and computes $x_i := y_i - \eta_{l+i} - \sum_{j\in J} \chi_j^i \cdot \eta_j$, for $i \in [r]$. Then, $Sim_{\mathcal{R}}$ sends $\boldsymbol{x}, \boldsymbol{y}$ to $\mathcal{A}$.*
4. *$Sim_{\mathcal{R}}$ reads $\boldsymbol{z}$ that $\mathcal{A}$ sends to $\mathcal{F}_{\text{EQ}}$. $Sim_{\mathcal{R}}$ computes $\hat{z}_i := v_{l+i} + \sum_{j\in J} \chi_j^i \cdot v_j - \alpha \cdot x_i$, for $i \in [r]$. $Sim_{\mathcal{R}}$ sends $\hat{\boldsymbol{z}}$ to $\mathcal{A}$ (emulating $\mathcal{F}_{\text{EQ}}$). If $\boldsymbol{z} = \hat{\boldsymbol{z}}$, $Sim_{\mathcal{R}}$ sends* `true` *to $\mathcal{A}$. Otherwise, $Sim_{\mathcal{R}}$ sends* `abort` *to $\mathcal{A}$ and aborts.*
5. *$Sim_{\mathcal{R}}$ sends $\boldsymbol{v}|_{[l]}$ to $\mathcal{F}_{\text{embVOLE}}^{\texttt{GR}(2^k,d)}$.*

The indistinguishability between ideal simulation and real execution for corrupted $P_{\mathcal{R}}$ is simple. We first consider the view of $\mathcal{A}$. In real protocol, $\mathcal{A}$ receives $\boldsymbol{\eta} \in \texttt{Ker}(\psi)^{l+r}$. Since $\boldsymbol{a}$ is distributed uniformly at random in $\texttt{GR}(2^k, d)^{l+r}$, $\boldsymbol{\eta}|_{\hat{J}}$ is distributed uniformly at random in $\texttt{Ker}(\psi)^{|J|+r}$. While in simulation, $\hat{\boldsymbol{\eta}}|_{\hat{J}}$ are uniformly sampled from $\texttt{Ker}(\psi)^{|J|+r}$ as well. As for $(\boldsymbol{x}, \boldsymbol{y} = \tau(\boldsymbol{x}))$ in real protocol, $\boldsymbol{x}$ is masked with $\boldsymbol{a}|_{[l+1:l+r]}$, which makes $\boldsymbol{y}$ have the uniform distribution on $\texttt{Im}(\phi)^r$. Thus, $(\boldsymbol{x}, \boldsymbol{y})$ generated by $Sim_{\mathcal{R}}$ have the same distribution as that in the real protocol. The final message that $\mathcal{A}$ receives from $\mathcal{F}_{\text{EQ}}$ is $\hat{\boldsymbol{z}}$. It can be easily verified that

$$z_i = v_{l+i} + \sum_{j\in J} \chi_j^i \cdot v_j - \alpha \cdot x_i = b_{l+i} + \sum_{j\in J} \chi_j^i \cdot b_j = \hat{z}_i,$$

Thus $\hat{\boldsymbol{z}}$ has the same distribution in both worlds. Further, if $\mathcal{F}_{\text{EQ}}$ aborts in real execution, $Sim_{\mathcal{R}}$ will send `false` *to $\mathcal{A}$ and aborts as well. Finally, we turn to the output of the honest $P_{\mathcal{S}}$. In real protocol, a_i is conditioned on that $\tau(a_i) - a_i = \eta_i$, for all $i \in J$, while they have the same properties in ideal simulation. Therefore, no* PPT *environment $\mathcal{Z}$ can distinguish the ideal simulation and the real execution. This completes the proof.* □

B.4 $\ddot{e}$VOLE & $\ddot{c}$VOLE

Functionality $\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$

$\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ extends the ideal functionality $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$ in Figure 5. **Setup phase**, **Send phases**, and **Deliver phases** are identical to those in $\mathcal{F}_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$, respectively. For the **Verify phases**, $\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ additionally supports RMFE verification. Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, length parameters $l_1,...,l_n \in \mathbb{N}$. Let (ϕ,ψ) be an $(m,d;3)$-RMFE over $\mathbb{Z}_{2^k}$.

Verify phases: *(RMFE verification)* Upon receiving $(sid; \mathtt{Verify}\sharp; l_t; J)$ from $P_{\mathcal{R}}$, where $t \in [n], J \subseteq [l_t]$ and sid is a session identifier, verify that there are stored inputs $(sid; \boldsymbol{a}, \boldsymbol{b}; l_t)$ from $P_{\mathcal{S}}$; else ignore the message. Then, verify that $\boldsymbol{a}|_J \in \mathrm{Im}(\phi)^{|J|}$. If the check fails, send $\perp$ to $P_{\mathcal{R}}$.

Fig. 17: Distributional certified VOLE with equality and RMFE constraints.

Theorem 11 (Lemma 6, restated). *$\Pi_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ in the $\mathcal{F}_{\mathrm{embVOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model with reusable malicious security.*

Proof. If the receiver is corrupted, the simulator $\mathcal{S}$ acts as same as the simulator for $\Pi_{\mathrm{eVOLE}}^{\mathtt{GR}(2^k,d)}$, except that $\mathcal{S}$ here need to additionally respond to the command $(\mathbf{Verify}\sharp, t, J_t)$ from the adversary. On this command, $\mathcal{S}$ sends $\hat{\boldsymbol{\eta}}_t$ to the adversary such that $\hat{\boldsymbol{\eta}}_t|_{J_t} := (\boldsymbol{u}_t - \tau(\boldsymbol{u}_t))|_{J_t}$ and zeros everywhere else. If the sender is corrupted, as we realize $\ddot{e}$VOLE in the $\mathcal{F}_{\mathrm{embVOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model, the receiver $P_{\mathcal{R}}$ verifies the RMFE constraints for free. The simulator $\mathcal{S}$ emulates the $\mathcal{F}_{\mathrm{embVOLE}}^{\mathtt{GR}(2^k,d)}$ functionality and extracts $\mathcal{A}$'s inputs from $\mathcal{A}$' messages, then $\mathcal{S}$ forwards the inputs to the $\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ functionality. The soundness error here is upper bounded by $1/2^{d-1}$ as that in Corollary 4. This completes the proof. □

Theorem 12 (Lemma 7, restated). *Instantiating $\mathcal{F}_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ with $\Pi_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$, we have that $\Pi_{\ddot{\mathrm{c}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ realizes $\mathcal{F}_{\ddot{\mathrm{c}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ in the $\mathcal{F}_{\mathrm{embVOLE}}^{\mathtt{GR}(2^k,d)}$-hybrid model with reusable malicious security.*

Proof. Correctness and security are immediately obtained from the correctness and security of the underlying protocols (as it is built directly upon $\Pi_{\ddot{\mathrm{e}}\mathrm{VOLE}}^{\mathtt{GR}(2^k,d)}$ and $\Pi_{\mathrm{NIZK}\cdot}^{q,t}$). □

B.5 Reverse subring VOLE

Theorem 13. *Protocol $\Pi_{\mathrm{rsVOLE}}^{\mathtt{GR}(2^k,d)}$ perfectly realizes $\mathcal{F}_{\mathrm{rsVOLE}}^{\mathtt{GR}(2^k,d)}$ with semi-honest security.*

Protocol $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\ddot{\mathrm{e}}\mathrm{VOLE}}$

Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, length parameters $l_1, l_2 \in \mathbb{N}$. Let (ϕ,ψ) be an $(m,d;3)$-RMFE over $\mathbb{Z}_{2^k}$. $P_{\mathcal{S}}$ has inputs $\boldsymbol{a}_t, \boldsymbol{b}_t \in \mathtt{GR}(2^k,d)^{l_t}$, $t \in [2]$. $P_{\mathcal{R}}$ has (random) inputs $\alpha_1, \alpha_2 \in \mathtt{GR}(2^k,d)$.

1. The sender $P_{\mathcal{S}}$ and the receiver $P_{\mathcal{R}}$ invoke the **Setup phase** of $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$ with $P_{\mathcal{R}}$'s inputs (α_1, α_2).
2. For $t \in [2]$, $P_{\mathcal{S}}$ and $P_{\mathcal{R}}$ invoke the **Send phases** of $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$ with inputs $(t; l_t + 1)$. The sender $P_{\mathcal{S}}$ receives $\hat{\boldsymbol{a}}_t, \hat{\boldsymbol{b}}_t \in \mathtt{GR}(2^k,d)^{l_t+1}$, while $P_{\mathcal{R}}$ receives $\hat{\boldsymbol{v}}_t \in \mathtt{GR}(2^k,d)^{l_t+1}$, such that $\hat{\boldsymbol{v}}_t = \hat{\boldsymbol{a}}_t \cdot \alpha_t + \hat{\boldsymbol{b}}_t$.
3. For $t \in [2]$, $P_{\mathcal{S}}$ sends $\boldsymbol{u}_t := \boldsymbol{a}_t - \hat{\boldsymbol{a}}_t|_{[l_t]}$, $\boldsymbol{w}_t := \boldsymbol{b}_t - \hat{\boldsymbol{b}}_t|_{[l_t]}$ to $P_{\mathcal{R}}$.

Verify♯: On input session id t, length l_t and $J_t \subseteq [l_t]$.

(i) The sender $P_{\mathcal{S}}$ and the receiver $P_{\mathcal{R}}$ invoke the **Deliver phases** of $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$ with inputs $(t; l_t; J_t)$. Upon receiving $\hat{\boldsymbol{\eta}}_t \in \mathtt{GR}(2^k,d)^{l_t}$ from $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$, where $\hat{\boldsymbol{\eta}}_t|_{J_t} = (\tau(\hat{\boldsymbol{a}}_t) - \hat{\boldsymbol{a}}_t)|_{J_t}$ and 0's in all of the remaining entries, $P_{\mathcal{R}}$ checks that $(\boldsymbol{u}_t - \tau(\boldsymbol{u}_t))|_{J_t} = \hat{\boldsymbol{\eta}}_t|_{J_t}$. If the check fails, $P_{\mathcal{R}}$ aborts.

Verify†: The same as **Verify†** of $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\mathrm{eVOLE}}$.

Verify‡: The same as **Verify‡** of $\mathit{\Pi}^{\mathtt{GR}(2^k,d)}_{\mathrm{eVOLE}}$.

Fig. 18: Protocol for ëVOLE over $\mathtt{GR}(2^k,d)$ in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\mathrm{embVOLE}}$-hybrid model.

Functionality $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{\mathrm{e}}\mathrm{VOLE}}$

Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, a sequence of n positive integers $l_1, ..., l_n$, a series of subsets $J_i \subseteq [l_i]$, for $i \in [n]$, and an arithmetic circuit $\mathcal{C}$ over $\mathtt{GR}(2^k,d)$ on $q \leq 2\sum_{i=1}^{n} l_i$ inputs. Let (ϕ,ψ) be an $(m,d;3)$-RMFE over $\mathbb{Z}_{2^k}$. Suppose $P_{\mathcal{S}}$ has input $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathtt{GR}(2^k,d)^{l_i}$, and $P_{\mathcal{R}}$ has input $\alpha_i \in \mathtt{GR}(2^k,d)$, for $i \in [n]$.

1. Receive $(\alpha_1, ..., \alpha_n)$ from $P_{\mathcal{R}}$, and $(\boldsymbol{a}_1, ..., \boldsymbol{a}_n, \boldsymbol{b}_1, ..., \boldsymbol{b}_n)$ from $P_{\mathcal{S}}$.
2. Verify that $\boldsymbol{a}_i|_{J_i} \in \mathrm{Im}(\phi)^{|J_i|}$ for $i \in [n]$ and $(\boldsymbol{a}_1, ..., \boldsymbol{a}_n, \boldsymbol{b}_1, ..., \boldsymbol{b}_n)$ is a satisfying assignment for circuit $\mathcal{C}$. If the check fails, send $\perp$ to $P_{\mathcal{R}}$. Otherwise, compute $\boldsymbol{v}_i := \boldsymbol{a}_i \cdot \alpha_i + \boldsymbol{b}_i$ for $i \in [n]$ and send $(\boldsymbol{v}_1, ..., \boldsymbol{v}_n)$ to $P_{\mathcal{R}}$.

Fig. 19: Certified VOLE with a general arithmetic constraint

Protocol $\Pi^{\mathtt{GR}(2^k,d)}_{\ddot{c}\mathrm{VOLE}}$

Parameterized by a Galois ring $\mathtt{GR}(2^k,d)$, a sequence of n positive integers $l_1,...,l_n$, a series of subsets $J_i \subseteq [l_i]$, for $i \in [n]$, and an arithmetic circuit $\mathcal{C}$ over $\mathtt{GR}(2^k,d)$ on $q_a + q_b = q \leq 2\sum_{i=1}^{n} l_i$ inputs with t multiplication gates. Let $L_1 = 0$ and for $i = 2,3,...,n+1$, let $L_i = l_1 + ... + l_{i-1}$. Let (ϕ,ψ) be an $(m,d;3)$-RMFE over $\mathbb{Z}_{2^k}$. The receiver $P_{\mathcal{R}}$ has inputs $\alpha_i \in \mathtt{GR}(2^k,d)$ and the sender $P_{\mathcal{S}}$ has inputs $\boldsymbol{a}_i, \boldsymbol{b}_i \in \mathtt{GR}(2^k,d)^{l_i}$, for $i \in [n]$. Suppose $\mathcal{C}$ takes q_a inputs from $\boldsymbol{a}$ entries and q_b inputs from $\boldsymbol{b}$ entries.

1. The two parties invoke the **Setup phase** of $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$ with $P_{\mathcal{R}}$'s inputs $(\alpha_1 + \beta, ..., \alpha_s + \beta, \beta, \gamma)$, where $\beta, \gamma \xleftarrow{\$} \mathtt{GR}(2^k,d)$.
2. For $i \in [n]$, $P_{\mathcal{S}}$ picks $\boldsymbol{e}_i \xleftarrow{\$} \mathtt{GR}(2^k,d)^{l_i}$ and sends $(\boldsymbol{a}_i, \boldsymbol{b}_i + \boldsymbol{e}_i)$ with session id i to $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$. For the $(n+1)$-st instance of VOLE, $P_{\mathcal{S}}$ computes $\boldsymbol{a}_{n+1} := \boldsymbol{a}_1 \parallel ... \parallel \boldsymbol{a}_n, \boldsymbol{b}_{n+1} := \boldsymbol{e}_1 \parallel ... \parallel \boldsymbol{e}_n$ and sends $(n+1; \boldsymbol{a}_{n+1}, \boldsymbol{b}_{n+1}; L_{n+1})$ to $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$. For the $(n+2)$-nd instance of VOLE, if $a_{i,j}$ is the k-th input from $\boldsymbol{a}$ entries to circuit $\mathcal{C}$, set $a_{n+2,k} := a_{i,j}$; else if $b_{i,j}$ is the k-th input from $\boldsymbol{b}$ entries to circuit $\mathcal{C}$, set $a_{n+2,q_a+k} := b_{i,j}$ and $a_{n+2,q+k} := b_{n+1,L_i+j}$. Additionally, $P_{\mathcal{S}}$ picks $\boldsymbol{b}_{n+2} \xleftarrow{\$} \mathtt{GR}(2^k,d)^{q+q_b}$. Then, $P_{\mathcal{S}}$ sends $(n+2; \boldsymbol{a}_{n+2}, \boldsymbol{b}_{n+2}; q)$ to $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$. The receiver $P_{\mathcal{R}}$ receives $\boldsymbol{v}_1, ..., \boldsymbol{v}_{n+2}$ from $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$.
3. By invoking the **Verify phases** of $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$, $P_{\mathcal{R}}$ verifies that
   (i) $\boldsymbol{a}_i|_{J_i} \in \mathrm{Im}(\phi)^{|J_i|}$, for $i \in [n]$. (*This is the main difference from* $\Pi^{\mathtt{GR}(2^k,d)}_{\mathrm{cVOLE}}$.)
   (ii) $a_{i,j} = a_{n+2,k}$ and $a_{n+1,L_i+j} = a_{n+2,k}$, if $a_{i,j}$ is the k-th input from $\boldsymbol{a}$ entries to circuit $\mathcal{C}$, for $k \in [q_a]$.
   (iii) $b_{i,j} = a_{n+2,q_a+k}$ and $b_{n+1,L_i+j} = a_{n+2,q+k}$, if $b_{i,j}$ is the k-th input from $\boldsymbol{b}$ entries to circuit $\mathcal{C}$, for $k \in [q_b]$. Recompute v_{n+2,q_a+k} by subtracting $v_{n+2,q+k}$.
4. Invoke the subprotocol $\Pi^{q,t}_{\mathrm{NIZK}}$ with inputs $\{[a_{n+2,i}]_\gamma\}_{i\in[q]}$ to verify that $\{[a_{n+2,i}]_\gamma\}_{i\in[q]}$ is a satisfying assignment for $\mathcal{C}$. If any of above verifications fails, $P_{\mathcal{R}}$ aborts.

Fig. 20: Protocol for Certified VOLE with a general arithmetic constraint in the $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\ddot{e}\mathrm{VOLE}}$-hybrid model

Functionality $\mathcal{F}_{\text{rsVOLE}}^{\texttt{GR}(2^k,d)}$

Parameterized by a ring $\texttt{GR}(2^k, d)$, length parameters $l_1, ..., l_n \in \mathbb{N}$. The **Send phase**, and **Deliver phases** are the same as those in the functionality $\mathcal{F}_{\phi\text{VOLE}}^{\texttt{GR}(2^k,d)}$, respectively.

Setup phase: Upon receiving input $(sid; \alpha)$ from $P_{\mathcal{R}}$ where $\alpha \in \mathbb{Z}_{2^k}$ and sid is a session identifier, store $(sid; \alpha)$, send $(sid; \texttt{initialized})$ to the adversary and ignore any further inputs from $P_{\mathcal{R}}$ with the same session identifier sid.

Fig. 21: Ideal functionality for chosen-input reverse subring VOLE over $\texttt{GR}(2^k, d)$.

Proof. If $P_{\mathcal{R}}$ is corrupted, the simulator $Sim_{\mathcal{R}}$ receives $\alpha_0, ..., \alpha_{k-1} \in \{0, 1\}$ from the adversary $\mathcal{A}$. $Sim_{\mathcal{R}}$ computes $\alpha := \sum_{i=0}^{k-1} \alpha_i \cdot 2^i \in \mathbb{Z}_{2^k}$, and sends α to $\mathcal{F}_{\text{rsVOLE}}^{\texttt{GR}(2^k,d)}$. Upon receiving $\boldsymbol{v}$ from $\mathcal{F}_{\text{rsVOLE}}^{\texttt{GR}(2^k,d)}$, $Sim_{\mathcal{R}}$ samples $\boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1} \xleftarrow{\$} \texttt{GR}(2^k, d)^l$ and computes $\boldsymbol{v}_0 := \boldsymbol{v} - \sum_{i=1}^{k-1} \boldsymbol{v}_i \cdot 2^i$. $Sim_{\mathcal{R}}$ sends $\boldsymbol{v}_0, ..., \boldsymbol{v}_{k-1}$ to $\mathcal{A}$. The indistinguishability is clear since $\boldsymbol{v}_1, ..., \boldsymbol{v}_{k-1}$ are distributed uniformly at random both in the real world and the ideal world.

If $P_{\mathcal{S}}$ is corrupted, the simulator $Sim_{\mathcal{S}}$ receives $\boldsymbol{a}, \boldsymbol{b}_0, ..., \boldsymbol{b}_{k-1} \in \texttt{GR}(2^k, d)^l$ from the adversary $\mathcal{A}$. $Sim_{\mathcal{S}}$ computes $\boldsymbol{b} := \sum_{i=0}^{k-1} \boldsymbol{b}_i \cdot 2^i$, and sends $\boldsymbol{a}, \boldsymbol{b}$ to $\mathcal{F}_{\text{rsVOLE}}^{\texttt{GR}(2^k,d)}$. The indistinguishability is clear since the outputs $\boldsymbol{v}$ in the real world and the ideal world are identical. Thus, we conclude the proof. □

Protocol $\Pi_{\text{rsVOLE}}^{\texttt{GR}(2^k,d)}$

Parameterized by a Galois ring $\texttt{GR}(2^k, d)$, length parameter l. Suppose $P_{\mathcal{R}}$ has input $\alpha \in \mathbb{Z}_{2^k}$, and $P_{\mathcal{S}}$ has input $\boldsymbol{a}, \boldsymbol{b} \in \texttt{GR}(2^k, d)^l$. Write α as $\alpha = \alpha_0 + \alpha_1 \cdot 2 + ... + \alpha_{k-1} \cdot 2^{k-1}$, where $\alpha_0, ..., \alpha_{k-1} \in \{0, 1\}$.

1. For $i = 0, ..., k-1$, $P_{\mathcal{R}}$ sends $(i; \alpha_i)$ to $\mathcal{F}_{\text{OT}}$.
2. For $i = 1, ..., k-1$, $P_{\mathcal{S}}$ picks random $\boldsymbol{b}_i \in \texttt{GR}(2^k, d)^l$, and sets $\boldsymbol{b}_0 := \boldsymbol{b} - \sum_{i=1}^{k-1} \boldsymbol{b}_i \cdot 2^i$.
3. For $i = 0, 1, ..., k-1$, $P_{\mathcal{S}}$ sends $(i; \boldsymbol{b}_i, \boldsymbol{a} + \boldsymbol{b}_i)$ to $\mathcal{F}_{\text{OT}}$.
4. Upon receiving $(i; \boldsymbol{v}_i)$ from $\mathcal{F}_{\text{OT}}$, where $\boldsymbol{v}_i = \boldsymbol{a} \cdot \alpha_i + \boldsymbol{b}_i$, for $i = 0, 1, ..., k-1$, $P_{\mathcal{R}}$ computes $\boldsymbol{v} := \sum_{i=0}^{k-1} \boldsymbol{v}_i \cdot 2^i$.

Fig. 22: Protocol for reverse subring VOLE over $\texttt{GR}(2^k, d)$ in the $\mathcal{F}_{\text{OT}}$-hybrid model.

B.6 ϕVOLE.

Theorem 14 (Thorem 5, restated). *Assuming a two-message reusable VOLE protocol over* $\mathtt{GR}(2^k, d)$*, and a linearly-homomorphic commitment scheme over* $\mathtt{GR}(2^k, d)$*,* $\Omega^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$ *realizes* $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$ *in the random oracle model.*

Proof. Sketch. Correctness of $\Omega^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$ *is directly obtained by the correctness of the underlying VOLE protocol. Similarly, the reusable security against a malicious sender is obtained by that of the underlying VOLE protocol as well. We consider security against a malicious receiver. In general, a malicious* $P_{\mathcal{R}}$ *has three cheating strategies. First,* $P_{\mathcal{R}}$ *may compute hash functions or commitments incorrectly. These cheating behaviors can be detected by the check step-(a) of round-2 and security reduces to the security of the underlying hash function and the commitment scheme. Second,* $P_{\mathcal{R}}$ *may provide inconsistent inputs in different iterations, e.g. computes some* $\pi_{i,j,1}$ *from* $\hat{\alpha}_i + \beta_j$ *instead of* $\alpha_i + \beta_j$*. This will be detected by the random linear combination check (over* $\mathtt{GR}(2^k, d)$*), with soundness error* $1/2^d$*, by Lemma 1. Finally,* $P_{\mathcal{R}}$ *may compute some* $\pi_{i,j,1}$ *incorrectly or input some* $\alpha_i + \beta_j \notin \mathrm{Im}(\phi)$*. The resulting soundness error can be* $\mathsf{negl}(\lambda)$ *by choosing the cut-and-choose parameters appropriately. This completes the proof.* □

Theorem 15 (Theorem 6, restated). *Protocol* $\Pi^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$ *perfectly realizes* $\mathcal{F}^{\mathtt{GR}(2^k,d)}_{\phi\mathrm{VOLE}}$ *with semi-honest security.*

Proof. If $P_{\mathcal{R}}$ *is corrupted, the simulator* $Sim_{\mathcal{R}}$ *receives* $\alpha_1, ..., \alpha_m \in \mathbb{Z}_{2^k}$ *from the adversary* $\mathcal{A}$*.* $Sim_{\mathcal{R}}$ *computes* $\alpha := \phi(\boldsymbol{\alpha})$*, and sends* α *to the ideal VOLE functionality. Upon receiving* $\boldsymbol{v}$ *from the ideal VOLE functionality,* $Sim_{\mathcal{R}}$ *samples* $\boldsymbol{v}_2, ..., \boldsymbol{v}_m \xleftarrow{\$} \mathtt{GR}(2^k, d)^l$ *and computes* $\boldsymbol{v}_1 := \gamma_1^{-1}(\boldsymbol{v} - \sum_{i=2}^{m} \boldsymbol{v}_i \cdot \gamma_i)$*.* $Sim_{\mathcal{R}}$ *sends* $\boldsymbol{v}_1, ..., \boldsymbol{v}_m$ *to* $\mathcal{A}$*. The indistinguishability is clear since* $\boldsymbol{v}_2, ..., \boldsymbol{v}_m$ *are distributed uniformly at random both in the real world and the ideal world.*

If $P_{\mathcal{S}}$ *is corrupted, the simulator* $Sim_{\mathcal{S}}$ *receives* $\boldsymbol{a}, \boldsymbol{b}_1, ..., \boldsymbol{b}_m \in \mathtt{GR}(2^k, d)^l$ *from the adversary* $\mathcal{A}$*.* $Sim_{\mathcal{S}}$ *computes* $\boldsymbol{b} := \sum_{i=1}^{m} \boldsymbol{b}_i \cdot \gamma_i$*, and sends* $\boldsymbol{a}, \boldsymbol{b}$ *to the ideal VOLE functionality. The indistinguishability is clear since the outputs* $\boldsymbol{v}$ *in the real world and the ideal world are identical. Thus, we conclude the proof.* □