More Efficient Lattice-based OLE
from Circuit-private Linear HE with Polynomial Overhead

Leo de Castro[1], Duhyeong Kim[2], Miran Kim[3],
Keewoo Lee[4], Seonhong Min[5], and Yongsoo Song[5]

[1] MIT, Cambridge, MA, USA.
ldec@mit.edu
[2] Intel Labs, Hillsboro, OR, USA.
duhyeong.kim@intel.com
[3] Hanyang University, Seoul, Korea.
miran@hanyang.ac.kr
[4] UC Berkeley, Berkeley, CA.
keewoo.lee@berkeley.edu
[5] Seoul National University, Seoul, Korea.
{minsh, y.song}@snu.ac.kr.

Abstract. We present a new and efficient method to obtain circuit privacy for lattice-based linearly homomorphic encryptions (LHE). In particular, our method does not involve noise-flooding with exponetially large errors or iterative bootstrapping. As a direct result, we obtain a semi-honest oblivious linear evaluation (OLE) protocol with the same efficiency, reducing the communication cost of the prior state of the art by 50%. Consequently, the amortized time of our protocol improves the prior work by 33% under 100Mbps network setting. Our semi-honest OLE is the first to achieve both concrete efficiency and asymptotic quasi-optimality. Together with an extension of the recent zero-knowledge proof of plaintext knowledge, our LHE yields actively-secure OLE with 2.7x reduced communication from the prior work. When applied to Overdrive (Eurocrypt '18), an MPC preprocessing protocol, our method provides 1.4x improvement in communication over the state of the art.

Keywords: Homomorphic encryption · Circuit privacy · Oblivious linear evaluation · Secure multi-party computation.

1 Introduction

Homomorphic encryption (HE) [50, 32], which is a cryptosystem that supports computation on encrypted data, is a versatile tool in cryptography. This is still true even when homomorphic computation of *linear functions* is supported. So-called *linearly homomorphic encryption* (LHE) has found use in various contexts [25, 47, 8, 41, 20].

However, when designing cryptographic protocols with HE, we often need to guarantee not only the data privacy but also the *circuit privacy*. That is, we want the resulting ciphertext of homomorphic evaluation not to leak any

information about the evaluated circuit other than the output of the circuit. For lattice-based HE schemes, several methods to achieve circuit-privacy have been proposed, but they are unsatisfactory in the aspect of practical efficiency. They either (i) use the *noise-flooding* technique that adds *exponentially large* noise to resulting ciphertexts [32] or (ii) repeat exhaustive fully homomorphic encryption (FHE) *bootstrapping* [29].[6]

In this work, we introduce a new and efficient technique to achieve circuit-private LHE from the BFV scheme [15, 31]. Our technique does not involve noise-flooding or FHE bootstrapping, allowing us to maintain low computation costs while our ciphertext modulus is only polynomial in the security parameter. To demonstrate the performance, we apply our efficient LHE to construct oblivious linear evaluation (OLE) protocols and a general-purpose multi-party computation (MPC) protocol.

Oblivious Linear Evaluation. Oblivious linear evaluation (OLE) [46] is an arithmetic analog of oblivious transfer. OLE is a two-party protocol between a sender with $a, b \in R$ and a receiver with $x \in R$, where R is some finite ring. At the end of the protocol, the receiver obtains the value of $ax + b \in R$ while learning nothing about a and b, and the sender learns nothing about x. OLE is a fundamental building block in various cryptographic protocols: general-purpose secure multi-party computation (MPC) [39, 4, 28, 37, 21, 30], zero-knowledge proof (ZKP) [12, 52, 53], and private set intersection (PSI) [35, 49, 22].

A circuit-private LHE directly can lead to a two-round OLE protocol with passive security. We exploit this idea to construct the OLE protocol and then leverage ideas from the recent zero-knowledge proof system of Kim, Lee, Seo, and Song [42] to achieve malicious security. Both the semi-honest and malicious protocols inherit the compact HE parameters from the LHE scheme, resulting in lower communication costs than the noise-flooding approach [20].

Furthermore, our semi-honest OLE is asymptotically quasi-optimal (AQO)[7] [19]. Our protocol is concretely much more efficient and arguably more intuitive than the previous AQO-OLE of [19], which relies on *correlation extractor* of [10].

MPC Preprocessing. A popular approach in modern MPC protocols is the *preprocessing model*. Such protocols are divided into *offline* phase and *online* phase. In an offline phase, before input values or a circuit to compute is determined, parties generate *correlated randomnesses* (e.g., Beaver's triples [7]). Then, in an online phase, the parties consume these correlated randomnesses to carry out the secure computation. The main point of the preprocessing model is to push heavy cryptographic machinery into the offline phase so that the online phase enjoys high efficiency.

In this work, we focus on preprocessing for SPDZ-style protocols [27, 26], which is secure against actively corrupted majority. By utilizing our efficient

[6] One exception is [11]. See Sec. 1.3.

[7] We say a cryptographic scheme is *asymptotically quasi-optimal (AQO)* if it solves a size-n cryptographic problem with $\tilde{O}(n + \lambda)$ cost, where λ is the security parameter.

circuit-private LHE scheme, we substantially improve the communication cost of the state-of-the-art preprocessing protocol.

1.1 Our Contribution

In this work, we construct a circuit-private linear HE scheme from the RLWE-based BFV scheme [15, 31]. The existing method relies on either bootstrapping [29] or noise flooding [32], which introduces significant overhead in terms of space or time complexity. Our method is based on a simple observation that there is still room for further randomizations in the conventional homomorphic evaluation algorithm beyond noise flooding. More precisely, we observe that the coefficients of a linear function in the plaintext space have several corresponding elements in the ciphertext space so that any of them can be used in the evaluation while preserving the correctness of homomorphic computation. Our algorithm is surprisingly simple in that it only requires one more Gaussian sampling and a single encryption compared to the conventional (non-circuit-private) linear evaluation algorithm for BFV. Furthermore, its overhead in parameter size is very small, in terms of both concrete and asymptotic manners.

We also extend this idea to design efficient OLE and MPC preprocessing protocols. A semi-honest OLE protocol is directly derived from our circuit-private linear evaluation technique, while a maliciously secure version can be obtained by adding some proof-of-knowledge techniques. In particular, we modify a proof technique of [42] and apply it to our protocol for minimizing the parameter and communication costs without rejection process. Finally, we improve a state-of-the-art MPC protocol for generating authenticated triples by substituting its HE part with our method. Our analysis shows that the amortized communication cost is reduced by a factor of about 1.4 compared to the prior work.

1.2 Technical Overview

Circuit-private Linear Evaluation in BFV. For a polynomial ring $R = \mathbb{Z}[X]/(X^n+1)$ and a plaintext modulus t, suppose that we are given a linear function $f(z) = az + b$ over the residue ring $R_t := \mathbb{Z}_t[X]/(X^n+1)$ for $a, b \in R_t$ and a BFV encryption $\mathbf{c} := \mathtt{Enc}(x, \mathbf{e}) \in R_q^2$ of a message $x \in R_t$ and an encryption randomness $\mathbf{e} \in R^3$. In BFV, the conventional way of homomorphically evaluating the linear function f is to compute $\mathbf{c}' = r \cdot \mathbf{c} + (\Delta b, 0) \pmod q$ where $r = [a]_t \in R$ and $\Delta = q/t \in \mathbb{Z}$. However, this method does not guarantee circuit privacy since the second component of $\mathbf{c}'$ completely reveals the information about $r = [a]_t$, and therefore b is also fully recovered. To achieve circuit privacy, prior works adopted a method to add an encryption of zero with an exponentially large randomness $\mathbf{e}'$ to the resulting ciphertext $\mathbf{c}'$, so-called noise-flooding, so that the information of a and b are statistically obliterated. However, this method yields an exponentially large ciphertext modulus, which significantly increases computation and communication costs.

This work starts with the following observations: (i) r does not necessarily need to be exactly $[a]_t$ but can be any small element in the coset $a + t\mathbb{Z}^n$ to

guarantee the correctness, and (ii) the addition of two discrete Gaussians over the coset $a + t\mathbb{Z}^n$ and $\mathbb{Z}^n$ with a proper width parameter can be statistically close to a discrete Gaussian distribution over $\mathbb{Z}^n$, which is notably independent of a. From these observations, we construct a new linear evaluation algorithm as follows. We first sample r and $\mathbf{e}'$ from discrete Gaussian distributions $D_{a+t\mathbb{Z}^n,\sigma}$ and $D^3_{\mathbb{Z}^n,\tau}$ respectively, for some width parameters $\sigma, \tau > 0$. Then, we compute a ciphertext $\mathbf{c}'$ as

$$\mathbf{c}' := r \cdot \mathbf{c} + \mathtt{Enc}(b, \mathbf{e}') \pmod q.$$

Interestingly, our algorithm computes an essentially identical formula to the previous method; however, the difference in the sampling procedure of r results in a significant reduction of the size of $\mathbf{e}'$. Technically, by setting $\sigma = O(t)$ and $\tau = O(\sigma n \|\mathbf{e}\|_\infty)$, the distribution of the convolved noise term $r \cdot \mathbf{e} + \mathbf{e}'$ of $\mathbf{c}'$ is statistically indistinguishable from the discrete Gaussian distribution $D_{\mathbb{Z}^{3n},\boldsymbol{\kappa},\sqrt{\Sigma}}$ where both $\boldsymbol{\kappa}$ and Σ are independent of r (refer to Theorem 1). This fact directly implies that the BFV scheme with our linear evaluation algorithm is circuit private, since $\mathbf{c}' = r \cdot \mathtt{Enc}(x, \mathbf{e}) + \mathtt{Enc}(b, \mathbf{e}') = \mathtt{Enc}(ax + b, r \cdot \mathbf{e} + \mathbf{e}') \pmod q$ and the only information dependent on a and b in $\mathbf{c}'$ is the resulting message $ax + b$.

When we compare the correctness condition of our linear evaluation algorithm (Lem. 10) and not circuit private conventional algorithm, we can easily verify that our algorithm only requires $\tilde{O}_\lambda(1)$ additional bits on the ciphertext modulus q for the width parameters $\sigma = O(t)$ and $\tau = O(\sigma n \|\mathbf{e}\|_\infty)$. Therefore, our linear evaluation algorithm provides circuit privacy *almost for free*.

Application to OLE and MPC Preprocessing. We remark that the semi-honest OLE protocol can be directly obtained from a homomorphic linear evaluation algorithm with circuit privacy. To achieve active security against malicious sender and receiver of the OLE protocol, we adopt some proof of knowledge (PoK) protocols so that both sender and receiver can be ensured that the information from the other is contained in the proven language. To be precise, the receiver first sends an encryption $\mathbf{c}$ of the plaintext x through the PoK protocol for the plaintext knowledge. We adopt the state-of-the-art proof of plaintext knowledge protocol, recently proposed by Kim, Lee, Seo, and Song [42], whose security is guaranteed under the hardness assumption of the Ring Learning with Errors (RLWE). In the evaluation phase, the sender sends the ciphertext $\mathbf{c}'$ obtained from our homomorphic linear evaluation algorithm on the linear function $f(z) = az + b$ through the PoK protocol for the knowledge of the coefficients a and b. In contrast to the first PoK, the second PoK does not require any computational hardness assumption (refer to Lem. 14).

In the case of MPC preprocessing, we observe that the LowGear protocol of Overdrive [41], which is the state-of-the-art preprocessing for SPDZ-style protocols [27], can be instantiated with any circuit-private LHE and a corresponding protocol for zero-knowledge proof of plaintext knowledge (ZKPoPK). Whereas the original LowGear protocol leverages noise-flooding for circuit privacy, we plug in our efficient circuit-private LHE to achieve better performance.

1.3 Related Work

Circuit Privacy. While most approaches to achieving circuit privacy for lattice-based HE leverage noise-flooding [32] or iterative FHE bootstrapping [29], Bourse-DelPino-Minelli-Wee [11] presented a new efficient method to achieve circuit privacy through *randomized gadget decomposition*. However, their approach only applies to the GSW-style encryptions [34]. We note that the GSW-style encryptions suffer from significantly larger ciphertext size and higher computation costs compared to the BFV encryption, hindering the practical usage.

Oblivious Linear Evaluation. Several prior works [6, 20] have attempted to optimize OLE protocols while still relying on the noise-flooding technique. The work of Baum et al. [6] constructs a one-round (simultaneous message) protocol for generating *random* OLE correlations. However, the ciphertexts in this protocol require either one or two applications of noise flooding, resulting in high communication costs. The work of de Castro, Juvekar, and Vaikuntanathan [20] optimized the "folklore" noise-flooding approach by removing the need to sample the flooding noise. While this saves computation time, the communication remains identical to that of the noise-flooding approach.

As a point of theoretical interest, the work of de Castro et al. [19] constructed AQO-OLE protocols. We observe that our semi-honest OLE protocol also achieves the same asymptotic performance without requiring the computationally expensive machinery of correlation extractors [10] as in the previous work.

A recent category of the OLE protocols generates random correlations with communication that is asymptotically sublinear in the length of the correlations [14, 23]. For the task of generating *random* OLE correlations, these protocols will eventually outperform our work. However, even beyond this crossover point, these protocols require a substantial amount of *seed* OLEs to expand out to the full protocol output. Therefore, our protocol remains applicable even in the setting where the overall sublinear correlation generator outperforms any linear-time OLE protocol.

There are previous works [18, 16] that achieve OLE protocols with appealing properties. For instance, [16] achieves *rate-1* OLE. However, they are mostly theoretical results, and analysis of their concrete efficiency is not provided. On the other hand, we aim to achieve concretely efficient OLE protocol.

MPC Preprocessing. Our main point of comparison for our MPC preprocessing protocol is the LowGear protocol of Overdrive [41], which follows the approach of [8] and SPDZ [27]. It is the state-of-the-art general-purpose MPC protocol in the setting of an actively corrupted majority when the number of participating parties is fairly small (e.g., 2-party computation).

A few optimizations for the LowGear protocol have been proposed up to date, such as TopGear [5] and LowGear2.0 [36]. These optimizations are orthogonal to our techniques, and therefore can be applied to our protocol, yielding performance improvements at similar rates. Thus, we describe and compare our protocol to the original LowGear protocol for simplicity.

Meanwhile, pseudorandom correlation generator (PCG) can also be used for MPC preprocessing [13, 14]. However, the secure setup of PCG itself requires a substantial amount of authenticated triples, which is a goal of our MPC preprocessing. Our protocol can be used to generate such triples for the initial setup of PCG.

1.4 Organization

The rest of the paper is organized as follows. Section 2 reviews some preliminaries on lattice and RLWE, and describes OLE-related functionalities. In Section 3, we present a new idea to construct a circuit-private HE scheme for affine functions over a polynomial ring. In Section 4, we present efficient OLE protocols against passive and malicious adversaries, together with correctness proof and security analysis. In Section 5, we apply our circuit-private LHE to MPC preprocessing.

2 Preliminary

2.1 Basic Notations and Terminology

For two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over a countable domain Ω, the *statistical distance* between $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as $\frac{1}{2}\sum_{x\in\Omega}|\mathcal{D}_1(x)-\mathcal{D}_2(x)|$. For a distribution $\mathcal{D}$ over Ω, we denote by $x \leftarrow \mathcal{D}$ sampling x from $\mathcal{D}$. When $\mathcal{D}$ is the uniform distribution over Ω, then we simply denote the sampling by $x \leftarrow \Omega$. For two vectors u, v with the same size, we denote the point-wise multiplication (Hadamard multiplication) by $u \odot v$.

Let $R := \mathbb{Z}[X]/(X^n+1)$ be the $(2n)$-th cyclotomic ring and $R_q := \mathbb{Z}_q[X]/(X^n+1)$ be the residue ring of R modulo an integer q. We identify $a = \sum_{i=0}^{n-1} a_i X^i$ in R (or R_q) with the vector of its coefficients $\mathbf{a} = (a_0, \cdots, a_{n-1})$. Hence, $\|a\|_\infty$ and $\|a\|_1$ denote the infinity norm and ℓ^1-norm of $\mathbf{a}$, respectively. We also identify a ring element $a = \sum_{i=0}^{n-1} a_i X^i \in R$ with a negacyclic matrix $\mathbf{A}$, where

$$\mathbf{A} = \begin{bmatrix} a_0 & -a_{n-1} & \dots & -a_1 \\ a_1 & a_0 & \dots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \dots & a_0 \end{bmatrix}.$$

For any $a, b \in R$, it holds that $\|ab\|_\infty \le \|a\|_\infty \|b\|_1 \le n\|a\|_\infty \|b\|_\infty$. Moreover, the vector representation of ab is obtained as $\mathbf{Ab}$ where $\mathbf{A}$ is the negacyclic matrix of a and $\mathbf{b}$ is the coefficient vector of b.

A symmetric real matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ called *positive-definite*, if $\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x} > 0$ holds for all $\mathbf{0} \neq \mathbf{x} \in \mathbb{R}^n$. Equivalently, positive-definite matrices can be characterized by their spectral decomposition of the form $\boldsymbol{\Sigma} = \mathbf{Q}\mathbf{D}^2\mathbf{Q}^\top$, where $\mathbf{Q} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix (i.e., $\mathbf{Q}^{-1} = \mathbf{Q}^\top$) and $\mathbf{D}$ is a diagonal matrix with positive diagonal entries. Thus, any positive-definite $\boldsymbol{\Sigma}$ is invertible and $\boldsymbol{\Sigma}^{-1}$ is also positive-definite.

We say that matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is a square root of $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$, written $\mathbf{A} = \sqrt{\boldsymbol{\Sigma}}$, if $\mathbf{A}\mathbf{A}^\top = \boldsymbol{\Sigma}$ holds. We use the notation $\sqrt{\boldsymbol{\Sigma}}$ when specific choice of square root is irrelevant. Note that every positive-definite matrix has a square root by the above spectral decomposition.

For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, we define the *matrix norm* of $\mathbf{A}$ as:

$$\|\mathbf{A}\| = \sup_{\mathbf{0}\neq\mathbf{x}\in\mathbb{R}^n} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2}.$$

We denote by $s_{\max}(\mathbf{A})$ and $s_{\min}(\mathbf{A})$ the largest and smallest singular values of $\mathbf{A}$, respectively. Note that $\|\mathbf{A}\| = s_{\max}(\mathbf{A})$.

$[a]_t$ denotes the residue of a modulo t, *i.e.,* $[a]_t = a \pmod t$.

2.2 Discrete Gaussians on Lattices

The n-dimensional Gaussian function $\rho : \mathbb{R}^n \mapsto (0, 1]$ is defined as

$$\rho(\mathbf{x}) = \exp(-\pi \cdot \|\mathbf{x}\|_2^2).$$

For a non-singular matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and a vector $\boldsymbol{\mu} \in \mathbb{R}^n$, we define $\rho_{\boldsymbol{\mu},\mathbf{A}}(\mathbf{x})$ as $\rho\left(\mathbf{A}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$. Note that, for a positive-definite $\boldsymbol{\Sigma}$, we have

$$\rho_{\boldsymbol{\mu},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}) = \exp\left(-\pi(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right).$$

When $\boldsymbol{\mu} = 0$, we omit $\boldsymbol{\mu}$ and denote it by $\rho_{\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x})$, for simplicity. If $\boldsymbol{\Sigma}$ is also $\sigma^2\mathbf{I}$ for some $\sigma > 0$, we denote it by $\rho_\sigma(\mathbf{x})$.

A *lattice* Λ is a discrete additive subgroup of $\mathbb{R}^n$. The *dual lattice* of a lattice $\Lambda \subset \mathbb{R}^n$ is defined as $\Lambda^* = \{\mathbf{v} \in \mathbb{R}^n | \forall \mathbf{u} \in \Lambda, \langle \mathbf{u}, \mathbf{v}\rangle \in \mathbb{Z}\}$. For a lattice $\Lambda \subset \mathbb{R}^n$, a vector $\boldsymbol{\mu} \in \mathbb{R}^n$ and a positive-definite matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$, we define the *discrete Gaussian distribution* over Λ centered at $\boldsymbol{\mu}$ and of covariance $\boldsymbol{\Sigma}$ as

$$D_{\Lambda,\boldsymbol{\mu},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}) = \frac{\rho_{\boldsymbol{\mu},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x})}{\rho_{\boldsymbol{\mu},\sqrt{\boldsymbol{\Sigma}}}(\Lambda)}.$$

It is simply denoted by $D_{\Lambda,\boldsymbol{\mu},\sigma}(\mathbf{x})$ When $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ for some $\sigma > 0$. When $\boldsymbol{\mu} = 0$, we omit $\boldsymbol{\mu}$ and denote by $D_{\Lambda,\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x})$ (or $D_{\Lambda,\sigma}(\mathbf{x})$).

Definition 1 (Smoothing parameter [44]). *For an n-dimensional lattice Λ and positive real $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest s such that $\rho_{1/s}(\Lambda^*\backslash\{\mathbf{0}\}) \leq \epsilon$.*

Lemma 1 ([33, Lem. 4.2]). *For any $c \in \mathbb{R}$, any $\epsilon > 0$, any $\sigma \geq \eta_\epsilon(\mathbb{Z})$, and any $\kappa > 0$, the following inequality holds.*

$$\Pr_{x\leftarrow D_{c+\mathbb{Z},\sigma}}[|x-c| \geq \kappa\cdot\sigma] \leq 2e^{-\pi\kappa^2}\cdot\frac{1+\epsilon}{1-\epsilon}$$

The following lemma states that the total Gaussian measures over cosets of a given lattice are essentially the same when the covariance is *sufficiently large* regarding the smoothing parameter.

Lemma 2 (Simplified Convolution Lemma [48]). *Let $\sigma_1, \sigma_2 > 0$ be reals such that $\sigma_3^{-2} := \sigma_1^{-2} + \sigma_2^{-2}$ satisfies $\sigma_3 \geq \eta_\epsilon(\mathbb{Z}^n)$ for some $0 < \epsilon < 1/2$. Then for an arbitrary $\mathbf{c} \in \mathbb{Z}^n$, the distribution*

$$\{\mathbf{x}_1 + \mathbf{x}_2 \mid \mathbf{x}_1 \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma_1},\ \mathbf{x}_2 \leftarrow \mathcal{D}_{\mathbb{Z}^n,\mathbf{c},\sigma_2}\}$$

is within statistical distance 2ϵ of $\mathcal{D}_{\mathbb{Z}^n,\mathbf{c},\sqrt{\sigma_1^2+\sigma_2^2}}$.

Lemma 3 ([1, Lem. 4]). *Let Λ be a full-rank n-dimensional lattice. For any real $\epsilon \in (0,1)$, vector $\mathbf{c} \in \mathbb{R}^n$, and non-singular matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ such that $s_{\min}(\mathbf{A}) \geq \eta_\epsilon(\Lambda)$,*

$$\rho_{\mathbf{A}}(\Lambda + \mathbf{c}) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \cdot \rho_{\mathbf{A}}(\Lambda).$$

Finally, we note a useful bound on smoothing parameters.

Lemma 4 ([44, Lem. 3.3]). *For any n-dimensional lattice Λ and $\epsilon > 0$,*

$$\eta_\epsilon(\Lambda) \leq \sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_n(\Lambda)$$

where $\lambda_n(\Lambda)$ is the n-th successive minimum of Λ. That is, $\lambda_n(\Lambda)$ is the smallest real number $r > 0$ such that $\dim(\mathrm{span}(\Lambda \cap rB)) = n$ where B is the n-dimensional unit ball centered at the origin.

The following is a direct corollary of Lem. 4.

Corollary 1. *For any $t > 0$ and $\epsilon > 0$, the following inequality holds.*

$$\eta_\epsilon(t\mathbb{Z}^n) \leq t \cdot \sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}}$$

2.3 RLWE

Definition 2. *Let n be a power of two, $q > 0$ be an integer and $\rho > 0$ be a real number. The Ring Learning with Errors (RLWE) distribution with parameter (n, q, ρ) and a secret $s \in R$ is a distribution over R_q^2 obtained as sampling $a \leftarrow R_q$ and $e \leftarrow D_{\mathbb{Z}^n,\rho}$, then returning (b, a) where $b = as + e \pmod q$. For a secret key distribution χ over R, the decisional RLWE problem denoted by $\mathsf{RLWE}(n, q, \rho, \chi)$ is to distinguish the RLWE distribution for a secret $s \leftarrow \chi$ from a uniform distribution over R_q^2.*

Throughout the paper, we assume that the secret is chosen from the ternary distribution, which means to select each coefficient from $\{-1, 0, 1\}$. When χ is a discrete Gaussian distribution with width parameter σ, we simply denote the RLWE problem by $\mathsf{RLWE}(n, q, \rho, \sigma)$.

2.4 Oblivious Linear Evaluation

The oblivious linear evaluation (OLE) is a secure computation protocol of a linear function between two parties: the receiver and the sender. A standard OLE over a field F allows the sender with $a, b \in F$ to securely compute $ax + b$ and send the output to the receiver who holds $x \in F$ [46, 39]. As an extension of the conventional OLE protocol, there exist two variants of the OLE protocol, namely *Batch OLE* (BOLE) and *Vector OLE* (VOLE). In BOLE, the receiver can learn a set of values $a_i \cdot x_i + b_i$ where the sender holds $a_i, b_i \in F$ and the receiver holds $x_i \in F$ for $1 \leq i \leq n$. To put it in other way, BOLE enables the receiver to perform multiple oblivious linear evaluations through a single protocol. In VOLE, the receiver learns a vector $\mathbf{a}x + \mathbf{b}$ where $\mathbf{a}, \mathbf{b} \in F^n$ and $x \in F$. VOLE can be viewed as a special case of a BOLE protocol in which the receiver sends only one input x.

More generally, the standard OLE functionality can be extended over a (finite) ring, referred to as *ring-OLE* [22]. Throughout the paper, we concentrate on the ring-OLE protocol over the residue ring R_t, since it is amenable to parallelism for VOLE and BOLE (when the ring dimension n and the plaintext modulus t satisfy the condition $2n|t-1$). The formal descriptions of the functionalities of the key generation and OLE are given in Fig. 1 and Fig. 2, respectively.

Functionality $\mathcal{F}_{\mathsf{KeyGen}}$

Input: a public parameter $pp = (n, q, \rho)$ and a secret key distribution χ.
Output:

1. Sample a secret key $s \leftarrow \chi$. Sample $a \leftarrow R_q$ and $e \leftarrow D_{\mathbb{Z}^n,\rho}$, and set the public key as $\mathbf{p} = (-as + e, a) \in R_q^2$.
2. Send s and $\mathbf{p}$ to the receiver.
3. Send $\mathbf{p}$ to the sender.

Fig. 1. The KeyGen functionality.

3 Circuit-private Linear Evaluation from BFV

In this section, we present a novel idea to build a circuit-private HE scheme for affine functions over a polynomial ring. Our approach is quite simple but offers a randomized evaluation method with only small overhead compared to the naïve approach.

We start with describing the RLWE-based BFV scheme as a *linearly homomorphic encryption* and present a new linear evaluation algorithm. Then, we provide correctness and security proofs for our algorithm.

Functionality $\mathcal{F}_{\mathsf{OLE}}$

Input:

1. A message $x \in R_t$ from the receiver
2. Coefficients $a, b \in R_t$ of a linear function $f(z) = az + b$ from the sender.

Output:

1. Send $f(x) = ax + b \in R_t$ to the receiver.
2. The sender receives nothing.

Fig. 2. The OLE functionality.

3.1 The BFV Scheme

In this section, we provide a brief description of the RLWE-based BFV encryption scheme [15, 31].

• $\mathtt{Setup}(1^\lambda)$: Given a security parameter λ, choose a power-of-two integer n, a ciphertext modulus q, a plaintext modulus t, an error parameter $\rho > 0$, and a key distribution χ over R. Return the public parameter $pp = (n, q, t, \rho, \chi)$.

• $\mathtt{KeyGen}(pp)$: Sample a secret key $s \leftarrow \chi$. Sample $p_1 \leftarrow R_q$ and $e_p \leftarrow D_{\mathbb{Z}^n,\rho}$, and set a public key as $\mathbf{p} = (p_0, p_1)$ where $p_0 = -p_1 \cdot s + e_p \pmod q$.

• $\mathtt{Enc}_{\mathbf{p}}(x)$: For a message $x \in R_t$, sample $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n,\rho}$, return $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$.

• $\mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$: For a message $x \in R_t$ and an encryption randomness $\mathbf{e} = (e_0, e_1, e_2) \in R^3$, return $\mathbf{c} = e_2 \cdot \mathbf{p} + (\Delta \cdot x + e_0, e_1) \pmod q$.

• $\mathtt{Dec}_s(\mathbf{c})$: For $\mathbf{c} = (c_0, c_1)$, return $\lfloor (c_0 + c_1 s)/\Delta \rceil \pmod t$.

Throughout this paper, we suppose that the plaintext modulus t divides the ciphertext modulus q. This assumption simplifies the construction of our algorithms and protocols, as well as their security analysis. We provide the security analysis of general case where t does not divide q in Appendix B. We also denote by $\Delta = q/t$ the scaling factor.

Two BFV encryption algorithms are presented above. By default, the BFV encryption algorithm samples an encryption randomness internally as described in the first algorithm. However, we also exploit the second algorithm where the randomness for encryption is given specifically.

Note that the BFV encryption is semantically secure: $\mathtt{Enc}_{\mathbf{p}}(x)$ is indistinguishable from a uniform random variable over R_q^2 under the RLWE assumption of parameter (n, q, ρ, χ). In addition, a fresh BFV encryption $\mathbf{c} = (c_0, c_1) \in R_q^2$ satisfies that

$$c_0 + c_1 s = e_2 \cdot (p_0 + p_1 s) + (\Delta \cdot x + e_0 + e_1 s) = \Delta \cdot x + e \pmod q$$

where $e = e_0 + e_1 s + e_2 e_p$ denotes the encryption error.

Finally, we describe a linear evaluation algorithm for the BFV scheme. For an affine function $f(z) = az + b$ defined over R_t and a given BFV encryption $\mathbf{c}$ of $x \in R_t$, it homomorphically evaluates f to obtain an encryption of $y = f(x) \in R_t$. However, it cannot be simply done by computing f over the input ciphertext $\mathbf{c} \in R_q^2$ since the coefficients $a, b \in R_t$ cannot be directly used in the computation over the ring R_q. Hence, these coefficients in R_t are embedded into elements of R and R_q via $a \mapsto r := [a]_t \in R$ and $b \mapsto \Delta \cdot b \in R_q$, so that the scalar multiplication and addition are well defined over R_q. A formal description is given below.

• $\mathtt{LinEval}(\mathbf{c}; a, b)$: Given a ciphertext $\mathbf{c} \in R_q^2$ and coefficients $a, b \in R_t$, let $r = [a]_t \in R$ and output $\mathbf{c}' = r \cdot \mathbf{c} + (\Delta b, 0) \pmod q$.

The correctness of this algorithm can be shown as follows: if $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ is a BFV encryption for some $x \in R_t$ and $\mathbf{e} \in R^3$, and a, b are elements of R_t with $r = [a]_t$, then we get

$$\mathbf{c}' = r \cdot \mathbf{c} + (\Delta b, 0) = \mathtt{Enc}_{\mathbf{p}}(y, r\mathbf{e}) \pmod q,$$

where $y = rx + b = ax + b \in R_t$.

3.2 Circuit Private Linear Evaluation

As described in the previous section, it is possible to homomorphically evaluate an arbitrary affine function $f(x) = ax + b$ using the BFV scheme; however, the basic linear evaluation algorithm does not guarantee the privacy of the evaluation circuit f. To be precise, for a ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$ and $a, b \in R_t$, the linear evaluation algorithm returns $\mathbf{c}' = (c_0', c_1') \leftarrow \mathtt{LinEval}(\mathbf{c}; a, b)$ which satisfies $c_1' = rc_1 \pmod q$ for $r = [a]_t$. Therefore, anyone can recover $a, b \in R_t$ by computing $a = c_1^{-1} \cdot c_1' \pmod t$ and $b = \Delta^{-1} \cdot (\mathbf{c}' - [a]_t \cdot \mathbf{c}) \pmod t$ from a pair of input and output ciphertexts. Similarly, it is also possible to recover r (and thereby $a = r \pmod t$) using the error terms of two ciphertexts: if $c_0 + c_1 s = \Delta \cdot x + e \pmod q$, then $c_0' + c_1' s = \Delta \cdot y + e' \pmod q$ for $e' = re \in R$.

Due to this problem, the notion of *circuit privacy* was introduced [38]. Briefly speaking, an evaluation algorithm is called to be circuit-private, if there is only negligible information leakage about the evaluation circuit from the output ciphertext. Since this work focuses on linear circuits, the notion of circuit privacy can be formally defined as follows.

Definition 3 (Linear Circuit Privacy). *A linearly homomorphic encryption scheme* $(\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{Enc}, \mathtt{Dec}, \mathtt{Eval})$ *is called circuit private for linear circuits if there exists a PPT simulator algorithm* Sim *such that for any* $x \in R_t$, $a, b \in R_t$ *and a PPT algorithm* $\mathcal{A}$,

$$|\Pr[\mathcal{A}(\mathtt{Eval}_{\mathsf{pk}}(\mathbf{c}; a, b), \mathbf{c}, \mathsf{sk}, \mathsf{pk})] - \Pr[\mathcal{A}(\mathsf{Sim}(\mathsf{sk}, \mathsf{pk}, \mathbf{c}, y), \mathbf{c}, \mathsf{sk}, \mathsf{pk}) = 1]|$$

is negligible where $pp \leftarrow \mathtt{Setup}(1^\lambda)$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathtt{KeyGen}(pp)$, $\mathbf{c} \leftarrow \mathtt{Enc}_{\mathsf{pk}}(x)$ *and* $y = ax + b \in R_t$.

To achieve the circuit privacy, the *noise-flooding* technique [32] has been mostly used in prior work. This method randomizes a ciphertext by adding an encryption of zero with a noise from a wide distribution so that the initial noise is overwhelmed by a newly introduced error. To be specific, a circuit private version of the linear evaluation algorithm can be obtained by modifying the previous evaluation algorithm as $\mathbf{c}' = \mathtt{LinEval}(\mathbf{c}; a, b) + \mathtt{Enc}_{\mathbf{p}}(0, \mathbf{e}')$ where $\mathbf{e}'$ is sampled from a discrete Gaussian distribution with an exponentially large width.

Although the noise flooding technique enables us to achieve the circuit privacy, it introduces a significant overhead since the ciphertext modulus q should be exponentially large to support a correct decryption of the resulting ciphertext $\mathbf{c}'$, thereby affecting both computation and communication costs.

In this work, we propose a new idea to achieve the circuit privacy with much smaller overhead compared to the noise flooding approach. We first remark that the output ciphertext of the previous method can be written as

$$\mathbf{c}' = \mathtt{LinEval}(\mathbf{c}; a, b) + \mathtt{Enc}_{\mathbf{p}}(0, \mathbf{e}') = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}}) \pmod q$$

where $r = [a]_t$ and $\tilde{\mathbf{e}} = r \cdot \mathbf{e} + \mathbf{e}' \in R$. Consequently, the noise flooding technique was inevitable to statistically obliterate the information of $r \cdot \mathbf{e}$.

Our main observation is that there is still room for further randomization in the evaluation algorithm. When we embed the coefficient a into an element $r \in R$ before multiplying it to $\mathbf{c}$, an arbitrary small element of R which is congruent to a modulo t can be chosen instead of setting r exactly as $[a]_t$. The correctness of linear evaluation still holds if r is reasonably small since the plaintext space is R_t. Based on this idea, we propose a novel *randomized* linear evaluation algorithm which samples r from a discrete Gaussian over the coset $a + t\mathbb{Z}^n$, the set of elements $r \in R$ such that $r = a \pmod t$. A formal description is given below:

• $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$: Given a ciphertext $\mathbf{c} \in R_q^2$ and ring elements $a, b \in R_t$, sample $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}$ and $\mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n,\tau}$. Compute and output $\mathbf{c}' := r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') \pmod q$.

This surprisingly simple algorithm enables us to use much smaller (asymptotically optimal) parameters, and therefore achieve substantial performance improvements. In the next section, we will analyze the algorithm and explain how the parameters $\sigma, \tau > 0$ for discrete Gaussian distributions should be chosen to meet the correctness and security requirements.

3.3 Correctness and Security

In this section, we show the correctness and security of our linear evaluation algorithm. We will use B_ρ, B_σ and B_τ to denote *essential* upper bounds of distributions $D_{\mathbb{Z}^n,\rho}$, $D_{a+t\mathbb{Z}^n,\sigma}$ and $D_{\mathbb{Z}^n,\tau}$, respectively, where a is an arbitrary element in R_t. To put it in another way, a sample from each distribution is bounded by the corresponding bound with overwhelming probability. We remark that these bounds can be obtained explicitly from Lem. 1.

Lemma 5 (Correctness). *Let* $\mathbf{c} \leftarrow \mathtt{Enc}_{\mathbf{p}}(x)$ *for some* $x \in R_t$*. Then, the algorithm* $\mathtt{RandLinEval}(\mathbf{c}; a, b)$ *outputs a BFV encryption of* $f(x) = ax + b$ *if*

$$(nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) < \Delta/2.$$

Proof. From the definition, we have $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ and

$$\mathbf{c}' = r \cdot \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e}) + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') = \mathtt{Enc}_{\mathbf{p}}(y, r \cdot \mathbf{e} + \mathbf{e}') \pmod{q} \tag{1}$$

for some $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n,\rho}$, $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}$ and $\mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n,\tau}$.

Let $\tilde{\mathbf{e}} = (\tilde{e}_0, \tilde{e}_1, \tilde{e}_2) = r \cdot \mathbf{e} + \mathbf{e}'$. Then, the output ciphertext $\mathbf{c}' = (c'_0, c'_1)$ is equal to $\mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}})$ and satisfies that $c'_0 + c'_1 s = \Delta y + e^* \pmod{q}$ with an error $e^* = \tilde{e}_0 + \tilde{e}_1 s + \tilde{e}_2 e_p$. Note that the key s is sampled from a ternary distribution and each coefficient of $\tilde{\mathbf{e}}$ is bounded by $n \cdot B_\rho B_\sigma + B_\tau$. Therefore, $\|e^*\|_\infty \le (nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) < \Delta/2$ and the resulting ciphertext $\mathbf{c}'$ decrypts to $y = ax + b$ correctly. □

We now show that our randomized evaluation algorithm achieves the circuit privacy property. We start with some useful lemmas to analyze the distribution of encryption randomness.

Lemma 6. *Let* $t > 0$ *be an integer,* $\mathbf{E} \in \mathbb{Z}^{m \times n}$ *a matrix, and* $\sigma, \tau > 0$ *reals such that*

$$\frac{1}{\sigma^2} + \frac{1}{\tau^2} \|\mathbf{E}\|^2 \le \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}$$

for some $0 < \epsilon \le 1/2$*. Then, for arbitrary* $\mathbf{a}$*,* $\boldsymbol{\mu} \in \mathbb{R}^n$ *and* $\boldsymbol{\nu} \in \mathbb{R}^m$*, the following distribution over* $\mathbb{Z}^m$

$$\mathcal{D} := \left\{ \mathbf{E}\mathbf{r} + \mathbf{e}' : \mathbf{r} \leftarrow D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}, \mathbf{e}' \leftarrow D_{\mathbb{Z}^m,\boldsymbol{\nu},\tau} \right\}$$

is within statistical distance 4ϵ *of* $D_{\mathbb{Z}^m,\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}$ *where* $\boldsymbol{\kappa} = \mathbf{E}\boldsymbol{\mu} + \boldsymbol{\nu}$ *and* $\boldsymbol{\Sigma} = \sigma^2 \cdot \mathbf{E}\mathbf{E}^\top + \tau^2 \cdot \mathbf{I}_m$*.*

Proof. For $\mathbf{x} \in \mathbb{Z}^m$, the probability that $\mathcal{D}$ outputs $\mathbf{x}$ can be written as follows:

$$\begin{aligned}
\mathcal{D}(\mathbf{x}) &= \Pr[\mathbf{E}\mathbf{r} + \mathbf{e}' = \mathbf{x} \mid \mathbf{r} \leftarrow D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}, \mathbf{e}' \leftarrow D_{\mathbb{Z}^m,\boldsymbol{\nu},\tau}] \\
&= \sum_{\mathbf{y} \in \mathbf{a}+t\mathbb{Z}^n} D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}(\mathbf{y}) \cdot D_{\mathbb{Z}^m,\boldsymbol{\nu},\tau}(\mathbf{x} - \mathbf{E}\mathbf{y}) \\
&\propto \sum_{\mathbf{y} \in \mathbf{a}+t\mathbb{Z}^n} \rho_\sigma(\mathbf{y} - \boldsymbol{\mu}) \cdot \rho_\tau(\mathbf{x} - \boldsymbol{\nu} - \mathbf{E}\mathbf{y}) \\
&= \sum_{\mathbf{y} \in \mathbf{a}-\boldsymbol{\mu}+t\mathbb{Z}^n} \exp\left[-\pi \left(\frac{1}{\sigma^2} \|\mathbf{y}\|^2 + \frac{1}{\tau^2} \|(\mathbf{x} - \boldsymbol{\kappa}) - \mathbf{E}\mathbf{y}\|^2 \right) \right].
\end{aligned}$$

Then, we have

$$\begin{aligned}
&\frac{1}{\sigma^2} \|\mathbf{y}\|^2 + \frac{1}{\tau^2} \|(\mathbf{x} - \boldsymbol{\kappa}) - \mathbf{E}\mathbf{y}\|^2 \\
&= \mathbf{y}^\top \boldsymbol{\Sigma}_1^{-1} \mathbf{y} - \frac{1}{\tau^2} \left(\mathbf{y}^\top \mathbf{E}^\top (\mathbf{x} - \boldsymbol{\kappa}) + (\mathbf{x} - \boldsymbol{\kappa})^\top \mathbf{E}\mathbf{y} \right) + \frac{1}{\tau^2} (\mathbf{x} - \boldsymbol{\kappa})^\top (\mathbf{x} - \boldsymbol{\kappa}) \\
&= \left(\mathbf{y} - \frac{1}{\tau^2} \boldsymbol{\Sigma}_1 \mathbf{E}^\top (\mathbf{x} - \boldsymbol{\kappa}) \right)^\top \boldsymbol{\Sigma}_1^{-1} \left(\mathbf{y} - \frac{1}{\tau^2} \boldsymbol{\Sigma}_1 \mathbf{E}^\top (\mathbf{x} - \boldsymbol{\kappa}) \right) + (\mathbf{x} - \boldsymbol{\kappa})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\kappa}),
\end{aligned}$$

where $\boldsymbol{\Sigma}_1^{-1} := \frac{1}{\sigma^2}\mathbf{I}_n + \frac{1}{\tau^2}\mathbf{E}^\top\mathbf{E}$. This implies that

$$\begin{aligned}
&\sum_{\mathbf{y}\in\mathbf{a}-\boldsymbol{\mu}+t\mathbb{Z}^n} \rho_{\sqrt{\boldsymbol{\Sigma}_1}}\left(\mathbf{y} - \frac{1}{\tau^2}\boldsymbol{\Sigma}_1\mathbf{E}^\top(\mathbf{x}-\boldsymbol{\kappa})\right)\cdot\rho_{\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}-\boldsymbol{\kappa}) \\
&= \rho_{\sqrt{\boldsymbol{\Sigma}_1}}\left(\mathbf{a}-\boldsymbol{\mu} - \frac{1}{\tau^2}\boldsymbol{\Sigma}_1\mathbf{E}^\top(\mathbf{x}-\boldsymbol{\kappa}) + t\mathbb{Z}^n\right)\cdot\rho_{\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}-\boldsymbol{\kappa}) \\
&\in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right]\cdot\rho_{\sqrt{\boldsymbol{\Sigma}_1}}(t\mathbb{Z}^n)\cdot\rho_{\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}-\boldsymbol{\kappa}) \propto \left[\frac{1-\epsilon}{1+\epsilon}, 1\right]\cdot D_{\mathbb{Z}^m,\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x})
\end{aligned}$$

where the last inclusion is derived from Lem. 3 and the condition

$$s_{\min}(\sqrt{\boldsymbol{\Sigma}_1}) = \frac{1}{\left\|\boldsymbol{\Sigma}_1^{-1}\right\|} \geq \frac{1}{\sqrt{\frac{1}{\sigma^2}+\frac{1}{\tau^2}\left\|\mathbf{E}\right\|^2}} \geq \eta_\epsilon(t\mathbb{Z}^n).$$

As a result, we obtain

$$\mathcal{D}(\mathbf{x}) \in \left[\frac{1-\epsilon}{1+\epsilon}, \frac{1+\epsilon}{1-\epsilon}\right]\cdot D_{\mathbb{Z}^m,\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}) \subseteq [1-4\epsilon, 1+4\epsilon]\cdot D_{\mathbb{Z}^m,\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}(\mathbf{x}),$$

and conclude that the statistical distance between $\mathcal{D}$ and $D_{\mathbb{Z}^m,\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}$ is bounded by 4ϵ. □

Corollary 2. *Let $\mathbf{e} = (e_0, e_1, e_2)$ be an element of R^3 with $\|\mathbf{e}\|_\infty \leq B_\rho$. If*

$$\frac{1}{\sigma^2} + \frac{3n^2B_\rho^2}{\tau^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}$$

for some $0 < \epsilon \leq 1/2$, then for any $\boldsymbol{\mu} \in R$ and $\boldsymbol{\nu} \in R^3$, the distribution of $\tilde{\mathbf{e}} = r\cdot\mathbf{e} + \mathbf{e}'$ over R^3 defined by $r \leftarrow D_{a+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}$ and $\mathbf{e}' \leftarrow D_{\mathbb{Z}^{3n},\boldsymbol{\nu},\tau}$ is within statistical distance 4ϵ of $D_{\mathbb{Z}^{3n},\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}$ for $\boldsymbol{\kappa} = \boldsymbol{\mu}\cdot\mathbf{e}+\boldsymbol{\nu}$ and $\boldsymbol{\Sigma} = \sigma^2\cdot\mathbf{E}\mathbf{E}^\top + \tau^2\cdot\mathbf{I}_{3n}$, where $\mathbf{E}_i \in \mathbb{Z}^{n\times n}$ is the negacyclic matrix corresponding to e_i for $j = 0, 1, 2$ and $\mathbf{E} = \begin{bmatrix}\mathbf{E}_0\\ \mathbf{E}_1\\ \mathbf{E}_2\end{bmatrix} \in \mathbb{Z}^{3n\times n}$.

Proof. Note that the coefficient representation of $r\cdot e_i$ can be written as $\mathbf{E}_i\mathbf{r}$ where $\mathbf{r}$ is the coefficient vector of r. Hence, this corollary is directly derived from the fact $\|\mathbf{E}\|^2 \leq 3n^2B_\rho^2$ and Lem. 6. □

Theorem 1 (Linear Circuit Privacy). *The BFV scheme, together with the randomized linear evaluation algorithm* $\texttt{RandLinEval}(\mathbf{c}; a, b)$ *under parameters σ and τ, is circuit-private for all affine functions over R_t if*

$$\frac{1}{\sigma^2} + \frac{3n^2B_\rho^2}{\tau^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}$$

for some negligible $\epsilon > 0$.

Proof. Let $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ be an encryption of $x \in R_t$ for some $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n,\rho}$. Let $f(z) = az + b$ be an affine function over R_t for some coefficients $a, b \in R_t$, and $y = ax + b \in R_t$. We define the simulator Sim for the output ciphertext $\mathbf{c}'$ of our linear evaluation algorithm as $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; 0, y)$, *i.e.,*

$$\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y) = \{\mathbf{c}' = r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(y, \mathbf{e}') \in R_q^2 : r \leftarrow D_{t\mathbb{Z}^n,\sigma}, \mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n,\tau}\}.$$

We use a hybrid argument to show that the distributions of $\mathbf{c}'$ from $\mathtt{RandLinEval}(\mathbf{c}; a, b)$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ are statistically indistinguishable.

Let us consider the following distribution:

$$\mathcal{H} := \{\mathbf{c}' = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}}) : \tilde{\mathbf{e}} = (\tilde{e}_0, \tilde{e}_1, \tilde{e}_2) \leftarrow D_{\mathbb{Z}^{3n},\sqrt{\boldsymbol{\Sigma}}}\}$$

where $\boldsymbol{\Sigma} = \sigma^2 \cdot \mathbf{E}\mathbf{E}^\top + \tau^2 \cdot \mathbf{I}_{3n}$ and $\mathbf{E}_i$ is the negacyclic matrix corresponding to e_i for $i = 0, 1, 2$ and $\mathbf{E} = \begin{bmatrix} \mathbf{E}_0 \\ \mathbf{E}_1 \\ \mathbf{E}_2 \end{bmatrix} \in \mathbb{Z}^{3n \times n}$.

We first claim that the real algorithm $\mathtt{RandLinEval}$ and $\mathcal{H}$ are statistically indistinguishable for any a and b. As shown in the proof of Lem. 5, the output ciphertext from $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ can be written as $\mathbf{c}' = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}})$ for some $r \in R$ sampled from $D_{a+t\mathbb{Z}^n,\sigma}$ and $\tilde{\mathbf{e}} = r\mathbf{e} + \mathbf{e}'$. Therefore, the distribution of $\mathtt{RandLinEval}(\mathbf{c}; a, b)$ is within statistical distance 4ϵ of the distribution of $\mathcal{H}$ from Cor. 2.

Secondly, we claim that $\mathcal{H}$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ are also statistically indistinguishable. This can be shown by using the same argument as above, since $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ is just a special case of $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ where $a = 0$ and $b = y$. Hence, the statistical distance between $\mathcal{H}$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ is bounded by 4ϵ.

Combining two claims, we conclude that $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ are within a negligible statistical distance $\leq 8\epsilon$. □

It is worth noting that we do not use the secret key s when building the simulator in this theorem. In addition, the hybrid game $\mathcal{H}$ relies on the fact that $\mathbf{c}$ can be written in the form of $\mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ and the randomness $\mathbf{e}$ is used to set the covariance matrix $\boldsymbol{\Sigma}$, but the simulator can be defined without this information.

Now, we discuss the optimality of our algorithm from Thm. 1. Recall that the main condition on parameter selection is:

$$\frac{1}{\sigma^2} + \frac{3n^2 B_\rho^2}{\tau^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}.$$

Roughly speaking, the error parameters can be chosen so that σ and $\tau / \|\mathbf{E}\|$ have a small bitsize comparable to the smoothing parameter $\eta(t\mathbb{Z}^n)$. It is worth noting that these values can be bounded by $8t$ in practical parameter sets (see Table 1). As a result, the random variables $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}$ and $\mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n,\tau}$ from the actual algorithm exhibit a comparable size to a and $t \cdot \mathbf{e}$ with only

$\tilde{O}_\lambda(1)$ bit differences. Now, recall that the final noise of $\texttt{RandLinEval}$, denoted by $r \cdot \mathbf{e} + \mathbf{e}'$, possesses an asymptotic size of $a \cdot \mathbf{e}$, which corresponds to the final noise from $\texttt{LinEval}$, the linear evaluation without noise-flooding. Consequently, our linear evaluation algorithm can effectively manage asymptotically optimal sized parameters, resulting in only a few bit difference compared to the optimal case.

4 Oblivious Linear Evaluation Protocols

In this section, we build efficient OLE protocols against passive and malicious adversaries by leveraging our circuit-private linear evaluation technique presented in the previous section.

4.1 Passively Secure OLE

Our OLE protocol in the semi-honest model $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ is directly derived from the circuit-private evaluation technique in the previous section. Its formal description is described in Fig. 3.

Protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$

Receiver's input: A message $x \in R_t$

Sender's input: The coefficients $a, b \in R_t$ of a linear function $f(z) = az + b$

Setup Phase: The receiver generates a secret key $s \in R$ and a public key $\mathbf{p} = (p_0, p_1) \in R_q^2$.

Encryption Phase: The receiver generates a ciphertext $\mathbf{c} \leftarrow \texttt{Enc}_{\mathbf{p}}(x)$ and sends it to the sender.

Evaluation Phase: The sender performs $\mathbf{c}' = \texttt{RandLinEval}_{\mathbf{p}}(\mathbf{c}, a, b)$ and sends it to the receiver.

Decryption Phase: The receiver outputs $y = \texttt{Dec}_s(\mathbf{c}')$.

Fig. 3. Semi-honest OLE protocol.

Lemma 7 (Correctness). *Suppose that both the receiver and the sender act honestly. Then, the protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ correctly computes the functionality $\mathcal{F}_{\mathsf{OLE}}$ with an overwhelming probability if $(nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) < \Delta/2$.*

Proof. The proof of this lemma directly follows from Lem. 5.

We now prove that $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ (Fig. 3) securely computes $\mathcal{F}_{\mathsf{OLE}}$ (Fig. 2) in the presence of a static passive adversary.

Lemma 8 (Security Against Sender). *The protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ satisfies receiver privacy against an honest-but-curious sender if the underlying BFV encryption scheme is semantically secure.*

Proof. We define $\mathcal{S}_{\mathsf{Receiver}}$ as a simulator which outputs a ciphertext $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(0, \mathbf{e})$ with $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n,\rho}$, independent of the receiver's input x. Since the BFV encryption scheme is IND-CPA secure under the RLWE assumption of parameter (n, q, ρ), the simulated view from $\mathcal{S}_{\mathsf{Receiver}}$ is computationally indistinguishable from the sender's view in the real protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$. □

Lemma 9 (Security Against Receiver). *The protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ is secure against an honest-but-curious receiver if*

$$\frac{1}{\sigma^2} + \frac{3n^2 B_\rho^2}{\tau^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2} \tag{2}$$

for some negligible $\epsilon > 0$.

Proof. To show the security against the receiver, it suffices to show that the ciphertext $\mathbf{c}'$ is simulatable. The proof directly follows from Thm. 1. □

Theorem 2. *The protocol $\Pi_{\mathsf{OLE}}^{\mathsf{passive}}$ securely realizes the functionality $\mathcal{F}_{\mathsf{OLE}}$ in the presence of static passive adversaries in the $\mathcal{F}_{\mathsf{KeyGen}}$-hybrid model if the conditions of Lem. 7, 8, and 9 hold.*

We remark that our scheme enjoys asymptotic optimality in terms of the communication cost and practical efficiency. As discussed in Sec. 3.3, the underlying linear evaluation algorithm in our protocol achieves asymptotic optimality for the noise parameters, which extends to our OLE protocol as well. Therefore our scheme can leverage a small ciphertext modulus, which only needs to accommodate the optimal-sized noise, thereby resulting in a low communication cost. We also present the recommended parameter sets which support 16, 32, 64, 80 and 128 bit of the plaintext modulus for our semi-honest OLE protocol in Table 1. A more comprehensive analysis and comparison of communication and computation costs of our protocol will be provided in the end of this section.

4.2 Maliciously Secure OLE

We now extend our passively-secure OLE protocol (Sec. 4.1) to a maliciously secure OLE via the Proof of Knowledge (PoK). To accomplish this goal, we exploit an efficient PoK protocol of [42] for the BFV cryptosystem. The high-level idea of [42] is to measure the leakage of the secrets when the transcripts are disclosed to the adversary during the PoK protocol by analyzing the conditional distribution of the secrets for given transcripts. Their main result is that it is possible to conceal the distribution of the noise with sub-linear sized additional noise,

Table 1. Recommended parameter sets for the semi-honest OLE with security level $\lambda = 128$.

$\log t$	n	$\log \sigma$	$\log \tau$	$\log q$
16	2^{12}	19	36	72
32	2^{12}	35	52	104
64	2^{13}	67	85	170
80	2^{13}	83	101	202
128	2^{14}	131	150	300

instead of super-polynomial sized noise. To elaborate further, let us consider the scenario where the prover holds the encryption $\mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ for the message x and the noise $\mathbf{e}$, and the verifier learns the distribution of $\alpha_i \cdot \mathbf{e} + \mathbf{f}_i$ for the i-th challenge α_i during the PoK protocol. Instead of hiding the distribution of $\mathbf{e}$ by noise-flooding with $\mathbf{f}_i$, one can measure the conditional distribution of $\mathbf{e}$ when $(\mathbf{e}, \alpha_1 \cdot \mathbf{e} + \mathbf{f}_1, \ldots, \alpha_\ell \cdot \mathbf{e} + \mathbf{f}_\ell)$ is given and choose the small-yet-sufficient size of $\mathbf{f}_i$ $(1 \leq i \leq \ell)$. In a similar manner, we construct a secure OLE protocol for malicious adversaries by measuring an information leak of the message x, a and b using the convolution lemma [48].

The formal description of our maliciously secure OLE protocol $\Pi_{\mathsf{OLE}}^{\mathsf{active}}$ is described in Fig. 4. Here, the first PoK protocol for the encryption phase, denoted by Π_{PoK1} is described in Fig. 5 and the second PoK protocol for the

Protocol $\Pi_{\mathsf{OLE}}^{\mathsf{active}}$

Receiver's input: A message $x \in R_t$

Sender's input: The coefficients $a, b \in R_t$ of a linear function $f(z) = az + b$

Setup Phase: The receiver generates a secret key $s \in R$ and a public key $\mathbf{p} = (p_0, p_1) \in R_q^2$.

Encryption Phase: The receiver and sender engage in the protocol Π_{PoK1} as the prover and verifier, respectively. If it succeeds, the sender accepts a ciphertext $\mathbf{c} \in R_q^2$ from the receiver. Otherwise, the parties abort.

Evaluation Phase: The receiver and sender engage in the protocol Π_{PoK2} as the verifier and prover, respectively. If it succeeds, the receiver accepts a ciphertext $\mathbf{c}' \in R_q^2$ from the sender. Otherwise, the parties abort.

Decryption Phase: The receiver outputs $y \leftarrow \mathtt{Dec}_s(\mathbf{c}')$.

Fig. 4. Maliciously secure OLE protocol.

Protocol Π_{PoK1}

Receiver (Prover)'s input
A message $x \in R_t$
A public key $\mathbf{p} = (p_0, p_1) \in R_q^2$

Sender (Verifier)'s input
A public key $\mathbf{p} = (p_0, p_1) \in R_q^2$

Encryption Phase

1. The receiver samples an encryption randomness $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n, \rho_1}$.
2. The receiver sends $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e}) \in R_q^2$ to the sender.

Commitment Phase

1. The receiver samples $z_i \leftarrow R_t$ and $\mathbf{f}_i \leftarrow D^3_{\mathbb{Z}^n, \rho_2}$, then computes $\mathbf{d}_i = \mathtt{Enc}_{\mathbf{p}}(z_i, \mathbf{f}_i) \in R_q^2$ for $1 \le i \le \ell$.
2. The receiver sends $\mathsf{comm} = (\mathbf{d}_1, \ldots, \mathbf{d}_\ell)$ to the sender.

Challenge Phase

1. Upon receiving $\mathsf{comm} = (\mathbf{d}_1, \ldots, \mathbf{d}_\ell)$, the sender samples $\alpha_1, \ldots, \alpha_\ell \leftarrow \mathcal{C}$.
2. The sender sends $\mathsf{chal} = (\alpha_1, \ldots, \alpha_\ell)$ to the receiver.

Response Phase

1. Upon receiving $\mathsf{chal} = (\alpha_1, \ldots, \alpha_\ell)$, the receiver computes $\mathbf{g}_i = \mathbf{f}_i + \alpha_i \mathbf{e} \in R^3$ and $w_i = z_i + \alpha_i x \in R_t$ for $1 \le i \le \ell$.
2. The receiver sends $\mathsf{resp} = ((w_1, \mathbf{g}_1), \ldots, (w_\ell, \mathbf{g}_\ell))$ to the sender.

Verification Phase

1. Upon receiving $\mathsf{resp} = ((w_1, \mathbf{g}_1), \ldots, (w_\ell, \mathbf{g}_\ell))$, the sender checks whether the following holds for all $1 \le i \le \ell$:

$$\begin{aligned} &\mathtt{Enc}_{\mathbf{p}}(w_i, \mathbf{g}_i) = \mathbf{d}_i + \alpha_i \mathbf{c} \pmod q, \\ &\|\mathbf{g}_i\|_\infty \le B_{\rho_1} + B_{\rho_2}. \end{aligned}$$

2. The sender outputs accept if all checks pass, otherwise reject.

Fig. 5. The first PoK protocol for the encryption phase.

Protocol Π_{PoK2}

Receiver (Verifier)'s input

A public key $\mathbf{p} = (p_0, p_1) \in R_q^2$
A ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$

Sender (Prover)'s input

A public key $\mathbf{p} = (p_0, p_1) \in R_q^2$
A ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$
The coefficients $a, b \in R_t$ of a linear function $f(z) = az + b$ over R_t

---

Evaluation Phase

1. The sender samples $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma_1}$.
2. The sender samples $\mathbf{e}' = (e'_0, e'_1, e'_2) \leftarrow D^3_{\mathbb{Z}^n,\tau_1}$.
3. The sender computes $\mathbf{c}' = (c'_0, c'_1) \leftarrow r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') \in R_q^2$ and sends it to the receiver.

Commitment Phase

1. The sender samples $u_i \leftarrow D_{\mathbb{Z}^n,\sigma_2}$, $h_i \leftarrow R_t$ and $\mathbf{f}'_i = (f'_{i,0}, f'_{i,1}, f'_{i,2}) \leftarrow D^3_{\mathbb{Z}^n,\tau_2}$, then computes $\mathbf{d}'_i = u_i \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(h_i, \mathbf{f}'_i) \in R_q^2$ for $1 \le i \le \ell$.
2. The sender sends $\mathsf{comm}' = (\mathbf{d}'_1, \ldots, \mathbf{d}'_\ell)$ to the receiver.

Challenge Phase

1. Upon receiving the commitment, the receiver samples $\alpha'_1, \ldots, \alpha'_\ell \leftarrow \mathcal{C}$.
2. The receiver sends $\mathsf{chal}' = (\alpha'_1, \ldots, \alpha'_\ell)$ to the sender.

Response Phase

1. Upon receiving $\mathsf{chal}' = (\alpha'_1, \ldots, \alpha'_\ell)$, the sender computes $v_i = u_i + \alpha'_i r \in R$, $k_i = h_i + \alpha'_i b \in R_t$, and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}' \in R^3$ for $1 \le i \le \ell$.
2. The sender sends $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$ to the receiver.

Verification Phase

1. Upon receiving $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$, the receiver checks whether the following holds for all $1 \le i \le \ell$:

$$
\begin{aligned}
& v_i \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(k_i, \mathbf{g}'_i) = \mathbf{d}'_i + \alpha'_i \mathbf{c}' \pmod q, \\
& \|v_i\|_\infty \le B_{\sigma_1} + B_{\sigma_2}, \\
& \|\mathbf{g}'_i\|_\infty \le B_{\tau_1} + B_{\tau_2}.
\end{aligned}
$$

2. The receiver outputs accept if all checks pass, otherwise reject.

Fig. 6. The second PoK protocol for the evaluation phase.

evaluation phase, denoted by Π_{PoK2} is described in Fig. 6. Analogous to our semi-honest oblivious linear evaluation protocol, we introduce the notions of $B_{\rho_1}, B_{\rho_2}, B_{\sigma_1}, B_{\sigma_2}, B_{\tau_1}$ and B_{τ_2}, which denotes the essential upper bound of $D_{\mathbb{Z}^n,\rho_1}, D_{\mathbb{Z}^n,\rho_2}, D_{a+t\mathbb{Z}^n,\sigma_1}, D_{\mathbb{Z}^n,\sigma_2}, D_{\mathbb{Z}^n,\tau_1}$ and $D_{\mathbb{Z}^n,\tau_2}$, for an arbitrary $a \in R_t$, respectively. Note that we set the challenge space as $\mathcal{C} = \{0,1\}$ for a fair comparison with one of the previous state-of-the-arts in MPC preprocessing called LowGear (See Sec. 5).

Security Proof. Here, we provide a security analysis of our maliciously secure OLE protocol $\Pi_{\mathsf{OLE}}^{\mathsf{active}}$. We first show that our parameters incorporate *slackness* of our PoKs, then show that our PoKs are indeed proper PoKs and compile into a maliciously-secure OLE protocol.

In the encryption and evaluation phases, the receiver and sender generate $\mathbf{c}$ and $\mathbf{c}'$, respectively, each of which is given with a proof of knowledge to guarantee its well-formedness. As is usual in lattice-based PoKs, there is a difference between honest and proven languages.

We define the witness relations for PoK1 as follows:

$$\mathbf{R}_1 = \{(\mathbf{c}, x, \mathbf{e}) \mid \mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e}), \|\mathbf{e}\|_\infty \leq B_{\rho_1}\},$$
$$\mathbf{R}_1' = \left\{(\mathbf{c}, x, \mathbf{e}) \mid \mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e}), \|\mathbf{e}\|_\infty \leq B_\rho'\right\},$$

where $B_\rho' = 2(B_{\rho_1} + B_{\rho_2})$. Then, $(x, \mathbf{e})$ can be viewed as a witness for the statement about the ciphertext $\mathbf{c}$. The honest and proven languages are

$$\mathbf{L}_1 = \left\{\mathbf{c} \in R_q^2 \mid \exists (x, \mathbf{e}) \in R_t \times R^3, (\mathbf{c}, x, \mathbf{e}) \in \mathbf{R}_1\right\},$$
$$\mathbf{L}_1' = \left\{\mathbf{c} \in R_q^2 \mid \exists (x, \mathbf{e}) \in R_t \times R^3, (\mathbf{c}, x, \mathbf{e}) \in \mathbf{R}_1'\right\},$$

respectively.

Similar to PoK1, the witness relations for PoK2 are defined as follows:

$$\mathbf{R}_2 = \{(\mathbf{c}', b, r, \mathbf{e}') \mid \mathbf{c}' = r\mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}'), \|r\|_\infty \leq B_{\sigma_1}, \|\mathbf{e}'\|_\infty \leq B_{\tau_1}\},$$
$$\mathbf{R}_2' = \{(\mathbf{c}', b, r, \mathbf{e}') \mid \mathbf{c}' = r\mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}'), \|r\|_\infty \leq B_\sigma', \|\mathbf{e}'\|_\infty \leq B_\tau'\},$$

where $B_\sigma' = 2(B_{\sigma_1} + B_{\sigma_2})$ and $B_\tau' = 2(B_{\tau_1} + B_{\tau_2})$. Then, the honest and proven languages are

$$\mathbf{L}_2 = \left\{\mathbf{c}' \in R_q^2 \mid \exists (b, r, \mathbf{e}') \in R_t \times R \times R^3, (\mathbf{c}', b, r, \mathbf{e}') \in \mathbf{R}_2\right\},$$
$$\mathbf{L}_2' = \left\{\mathbf{c}' \in R_q^2 \mid \exists (b, r, \mathbf{e}') \in R_t \times R \times R^3, (\mathbf{c}', b, r, \mathbf{e}') \in \mathbf{R}_2'\right\},$$

respectively. In other words, the sender provides a proof of knowledge that $\mathbf{c}'$ is honestly generated by evaluating a linear function. We also note that PoK2 relies on the proven language of PoK1.

Lemma 10 (Correctness). *If both the sender and the receiver honestly follow the protocol and* $(nB_{\rho_1}B_{\sigma_1} + B_{\tau_1})(1 + n + nB_{\rho_1}) < \Delta/2$, $\Pi_{\mathsf{OLE}}^{\mathsf{active}}$ *realizes the correct OLE functionality.*

Proof. See Appendix A.

We now provide a simplified lemma from [42] and show that our protocol achieves security against both a malicious sender and a malicious receiver.

Lemma 11 ([42, Simplified Lemma 7]). *Let $\sigma_1, \sigma_2 > 0$ be reals and $\alpha_1, \ldots, \alpha_\ell \in \{0,1\}$ for an integer $\ell \geq 0$. Let $\sigma_0' > 0$ be such that $\frac{1}{\sigma_0'^2} = \frac{1}{\sigma_1^2} + \frac{\ell'}{\sigma_2^2}$ where ℓ' is the number of i's with $\alpha_i = 1$. Then, for any coset $a + t\mathbb{Z}^n$, the following procedures produce statistically identical distributions over $(a + t\mathbb{Z}^n) \times \mathbb{Z}^n \times \cdots \times \mathbb{Z}^n$:*

(i) Sample $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma_1}$. Sample $u_i \leftarrow D_{\mathbb{Z}^n,\sigma_2}$ and let $v_i = u_i + \alpha_i r$ for $1 \leq i \leq \ell$. Return $(r, v_1, \ldots, v_\ell)$.
(ii) Sample $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma_1}$. Sample $u_i \leftarrow D_{\mathbb{Z}^n,\sigma_2}$ and let $v_i = u_i + \alpha_i r$ for $1 \leq i \leq \ell$. Sample $\hat{r} \leftarrow D_{a+t\mathbb{Z}^n,(\sigma_0'^2/\sigma_2^2)\cdot\sum_{i=1}^{\ell}\alpha_i v_i,\sigma_0'}$ and return $(\hat{r}, v_1, \ldots, v_\ell)$.

Note that the statistical identity still holds even when the coset $a + t\mathbb{Z}^n$ is substituted by $\mathbb{Z}^n$.

Proof. See Appendix A.

Lemma 12. *Assuming that we are provided with the same conditions as stated in Lem. 11, if $\sigma_0' \geq \eta_\epsilon(t\mathbb{Z}^n)$ for $0 < \epsilon \leq 1/2$, then the distribution of $(v_1, \ldots, v_\ell)$ is within statistical distance 2ϵ of a distribution which is independent of a.*

Proof. See Appendix A.

The security proof of Π_{PoK1} directly follows from [42, Theorem 2].

Lemma 13 (PoK1). *Let $\rho_0 > 0$ be a real such that $\frac{1}{\rho_0^2} = \frac{1}{\rho_1^2} + \frac{\ell}{\rho_2^2}$. If $\rho_0 \geq 2 \cdot \eta_\epsilon(\mathbb{Z}^n)$ for a negligible $\epsilon > 0$ and $2^{-\ell}$ is negligible, then Π_{PoK1} is a secure proof-of-knowledge protocol for the pair of languages $(\mathbf{L}_1, \mathbf{L}_1')$ under hardness assumption of $\mathsf{RLWE}(n, q, \rho, \chi)$ and $\mathsf{RLWE}(n, q, \frac{\rho_0}{\sqrt{2}}, \frac{\rho_0}{\sqrt{2}})$.*

Proof. See Appendix A.

Lemma 14 (PoK2). *Let σ_0, τ_0 be reals such that $\frac{1}{\sigma_0^2} = \frac{1}{\sigma_1^2} + \frac{\ell}{\sigma_2^2}$ and $\frac{1}{\tau_0^2} = \frac{1}{\tau_1^2} + \frac{\ell}{\tau_2^2}$. If $\sigma_0 \geq \eta_\epsilon(t\mathbb{Z}^n)$ and $\tau_0 \geq \eta_\epsilon(\mathbb{Z}^n)$ for a negligible $\epsilon > 0$, then the protocol $\Pi_{\mathsf{OLE}}^{\mathsf{active}}$ achieves security against a malicious sender. Moreover, Π_{PoK2} is a secure proof-of-knowledge protocol for the honest language $\mathbf{L}_2$ and the proven language $\mathbf{L}_2'$ if $2^{-\ell}$ is negligible.*

Proof. See Appendix A.

In Table 2, we provide five parameter sets that support various bit lengths of the plaintext modulus for the malicious OLE. These parameter sets correspond to 16, 32, 64, 80 and 128 bits of plaintext modulus. While we initially chose the challenge set $\mathcal{C} = \{0, 1\}$ in our security proof, we utilize the monomial challenge space to optimize the number of the challenges, following the prior works such

Simulator $\mathcal{S}_{\mathrm{PoK2}}$

Input
Public key $\mathbf{p} \in R_q^2$
Encryption randomness $\mathbf{e} \in R^3$
Ciphertext $\mathbf{c} \in R_q^2$
Output $y \in R_t$

---

1. Sample $\alpha'_i \leftarrow \mathcal{C}$ for $1 \le i \le \ell$ and let $\mathsf{chal}' = (\alpha'_1, \ldots, \alpha'_\ell)$. Let ℓ' be the number of $1 \le i \le \ell$ such that $\alpha'_i = 1$.
2. Sample $r \leftarrow D_{t\mathbb{Z}^n,\sigma_1}$ and $\mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n,\tau_1}$. Compute $\mathbf{c}' = r\mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(y, \mathbf{e}')$.
3. Sample $u_i \leftarrow D_{\mathbb{Z}^n,\sigma_2}$, $k_i \leftarrow R_t$ and $\mathbf{f}'_i \leftarrow D^3_{\mathbb{Z}^n,\tau_2}$ for $1 \le i \le \ell$. Let $v_i = u_i + \alpha'_i r$ and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}'$. Set $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$.
4. Compute $\mathbf{d}'_i = v_i\mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(k_i, \mathbf{g}'_i) - \alpha'_i\mathbf{c}'$ for $1 \le i \le \ell$. Set $\mathsf{comm} = (\mathbf{d}'_1, \ldots, \mathbf{d}'_\ell)$.
5. Output $(\mathbf{c}', \mathsf{comm}', \mathsf{chal}', \mathsf{resp}')$.

Fig. 7. The simulator $\mathcal{S}_{\mathrm{PoK2}}$ for PoK2

Table 2. Recommended Parameter Sets for Malicious OLE, when $\lambda = 128$.

$\log t$	n	ρ_0	$\log \sigma_0$	$\log \tau_0$	$\log q$
16	2^{12}	11.15	18	38	77
32	2^{13}	11.19	34	55	111
64	2^{13}	11.19	66	87	175
80	2^{13}	11.19	82	103	207
128	2^{14}	11.23	130	152	305

as [9, 42]. It is worth noting that our security proof can be naturally extended to accommodate the monomial challenge set, with two additional conditions of q being an odd prime number, and the encryption randomness multiplied by 2. These modifications guarantee the correctness of the argument for the knowledge extractor. As a result, the size of the challenge space becomes $2n$, and thus we can set $\ell = \lceil \frac{\lambda}{\log_2(2n)} \rceil$ for the security parameter λ.

4.3 Performance Analysis

Here we provide the analysis of the performance of our improved OLE protocol and compare the performance of our protocol to the prior arts. For a detailed discussion on parameter sets, we refer to Appendix D. We begin with a concise explanation of how the communication cost of the protocol is affected by the parameters. Recall that a bulk of communications during the protocol are

elements in the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$, whose elements can be represented as a coefficient vector of size n with each component bounded by q. Consequently, the communication cost for both semi-honest and malicious OLE is roughly proportional to $n \log q$. In order to minimize the communication cost of the scheme, we first fix the plaintext modulus t and then set q and n as small as possible.

Table 3. Parameter Comparison for the OLE protocols.

Security	Scheme	$\log t$	$\log q$	n
Passive	[20]	32	220	2^{13}
	Ours		104	2^{12}
Active	[6]	64	480	2^{14}
	Ours		176	2^{13}

Semi-honest OLE. In [20], the authors present a secure OLE protocol against semi-honest adversaries, providing 80-bit security. In their parameter set, $\log t \approx 32, \log q \approx 220$ and $n = 2^{13}$. On the other hand, in our construction, we can leverage a smaller ciphertext modulus and the ring dimension of $\log q \approx 104$ and $n = 2^{12}$ with the same size of plaintext modulus t, under 128-bit security.

Table 4. OLE performance comparison of our protocol and the protocol from [20]

	$\log t$	n	Elapsed Time	Communication Cost	Estimated Communication time	Time per OLE
Ours	32	2^{12}	4.25ms	256.6KiB	20.0ms	5.92μs
[20]	32	2^{13}	4.89ms	768.6KiB	60.0ms	7.92μs

We also provide a proof-of-concept implementation of the semi-honest version of our protocol, and compare it to the previous work [20]. The proposed method and prior method [20] were implemented in Lattigo v5 [45], and all the experiments were performed on a machine with Intel(R) Xeon(R) Platinum 8268 @ 2.90GHz CPU and 192GB RAM running Ubuntu 20.04.2 LTS. We estimate the communication time assuming that the parties are connected by a network with a bandwidth of 100Mbps (WAN). Table. 4 describes the benchmark results. Our protocol achieves ciphertext size which is 3 times more compact than the construction from [20], as discussed above. Moreover, the amortized time of our

protocol improves the prior work by 33%. It is worth noting that the main bottleneck of our method is Gaussian sampling, while rescaling operation is also a time-consuming operation in the construction of [20]. In fact, this can bring a lot more efficiency in the real world setting since the sender can pre-compute Gaussian samples during the communication while rescaling cannot be performed until the sender receives the ciphertext from the receiver. In our implementation, a rounded Gaussian Sampler was utilized instead of discrete Gaussian for sampling from $D^3_{\mathbb{Z}^n,\tau}$. Note that the statistical distance between the rounded Gaussian and discrete Gaussian is small enough since τ is sufficiently large in our parameters (≥ 36 bits).

Malicious OLE. During the OLE protocol against malicious adversaries, the most resource-intensive communication operation is the transmission of the BFV ciphertext. It is because the challenge, commitment, and the response are ring elements residing in the polynomial ring R, which can be represented with a single-limb integers while the BFV ciphertext consists of multi-limb integers. Therefore, we mainly focus on the size of the BFV encryption used in the protocol. Baum et al. [6] provides a secure OLE protocol against malicious adversaries, achieving 116-bit security. Their recommended parameter set is as follows: $\log t \approx 64, \log q \approx 480$ and $n = 2^{14}$. In contrast, our protocol employs a 176-bit ciphertext modulus q and smaller ring dimension $n = 2^{13}$, achieving 128 bit of security. Hence, the ciphertext size of our construction is 5.5x more compact than that of [6]. Similar to the case of Malicious OLE, our protocol achieves 2.7x lower (amortized) communication cost during the protocol, compared to the prior work.

5 MPC Preprocessing

In this section, we apply our circuit-private LHE (Sec. 3) to MPC preprocessing. In particular, we describe how our LHE can improve the communication cost of the LowGear protocol in Overdrive [41]. We only give a very brief overview of LowGear here. For a more detailed description, please refer to the Appendix C.

5.1 LowGear with Reduced Communication

One of the most important settings in MPC is the security against *actively corrupted majority*. The dishonest majority setting encompasses the significant case of two-party computation. And modeling the security threat as passive (honest-but-curious) adversaries is often unsatisfactory in real-life scenarios.

The LowGear protocol of Overdrive [41], which follows the framework of SPDZ[8] [27], is a state-of-the-art general-purpose MPC protocol in this setting. It is the fastest protocol (modulo follow-up optimizations[9]) when the number of

[8] Refer to Appendix C for an overview of the SPDZ framework.
[9] A few optimizations for LowGear have been proposed [5, 36]. See Sec. 1.3.

participating parties is small, especially in the case of 2-party computation.[10] The LowGear protocol leverages the LHE version of the BGV scheme [17] to preprocess so-called *(authenticated) Beaver's triples*. It also deploys noise-flooding and ZKPoPK[11] upon BGV in order to achieve claimed security. Meanwhile, the roles of the BGV scheme and noise-flooding in LowGear can be abstracted in terms of circuit-private LHE. That is, we can plug in any circuit-private LHE in place of BGV and noise-flooding. (Refer to the Appendix C for a generic description of LowGear in terms of circuit-private LHE.) The performance of the resulting protocol depends on the underlying LHE scheme and accompanied ZKPoPK.

Our approach to improving the LowGear protocol is straightforward: we plug in our circuit-private LHE from Sec. 3 and deploy ZKPoPK from [42]. While still enjoying high parallelism as the BGV scheme, our scheme allows us to use smaller parameters by avoiding noise-flooding. This directly improves the communication cost of the protocol.

5.2 Comparison

We compare the communication cost of our improved LowGear with the original LowGear from [41]. As the total communication of LowGear is dominated by the exchanges of LHE ciphertexts, it is sufficient to compare the LHE parameters. We give LHE parameters for the following three versions of LowGear in Table 5. For each plaintext modulus bit-size $\log |\mathbb{F}|$[12] and desired statistical security parameter sec, we list the smallest possible pair of ring dimension n and ciphertext modulus bit-size $\log q$.[13]

1. The original LowGear [41]. It leverages noise-flooding for circuit-privacy and deploys ZKPoPK from [27], which is based on the rejection sampling technique of [43].
2. LowGear with our circuit-private LHE but with ZKPoPK from [27].
3. LowGear with our circuit-private LHE and with ZKPoPK from [42].

Comparing the original LowGear with ours plus [42], the ciphertext size is reduced from 4.1MB to 1.4MB for $\log |\mathbb{F}| = 128$ and $\mathsf{sec} = 128$, which gives 2.9x improvement in total communication cost. Considering that we preprocess n triples in parallel using a single ciphertext via packing, ours plus [42] (resp., [27]) shows 1.4x (resp., 1.3x) improvement in amortized communication cost. A similar level of improvement is obtained in the case of $\log |\mathbb{F}| = 64$ and $\mathsf{sec} = 64$.

[10] Another nice feature of the LowGear approach is that it requires only PKI (public key infrastructure) for underlying LHE. On the other hand, protocols based on SHE (somewhat HE), such as the original SPDZ protocol [27] or HighGear protocol of Overdrive [41], require the parties to securely generate *distributed* SHE secret key, which is already a highly non-trivial task [51].

[11] Zero-Knowledge Proof of Plaintext Knowledge. An encryptor can prove that ciphertext is honestly generated through ZKPoPK.

[12] LowGear supports secure computations of arithmetic circuits over a prime field $\mathbb{F}$.

[13] All parameters are set for the case where we apply the technique of [24] for amortized proofs.

Table 5. LHE Parameters for LowGear-style MPC Preprocessing.

	$\log \lvert\mathbb{F}\rvert$	sec	n	$\log q$
LowGear [41]	64	64	2^{14}	294
	128	128	2^{15}	524
Ours + [27]	64	64	2^{14}	226
	128	128	2^{14}	392
Ours + [42]	64	64	2^{13}	201
	128	128	2^{14}	365

References

1. Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete gaussian leftover hash lemma over infinite domains. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 97–116. Springer (2013)
2. Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Halevi, S., Hoffstein, J., Laine, K., Lauter, K., et al.: Homomorphic encryption standard. In: Protecting Privacy through Homomorphic Encryption, pp. 31–62. Springer (2021)
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Cryptology ePrint Archive, Paper 2015/046 (2015), https://eprint.iacr.org/2015/046
4. Applebaum, B., Damgård, I., Ishai, Y., Nielsen, M., Zichron, L.: Secure arithmetic computation with constant computational overhead. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017. pp. 223–254. Springer International Publishing, Cham (2017)
5. Baum, C., Cozzo, D., Smart, N.P.: Using TopGear in overdrive: a more efficient ZKPoK for SPDZ. In: International Conference on Selected Areas in Cryptography. pp. 274–302. Springer (2019)
6. Baum, C., Escudero, D., Pedrouzo-Ulloa, A., Scholl, P., Troncoso-Pastoriza, J.R.: Efficient protocols for oblivious linear function evaluation from ring-lwe. In: Security and Cryptography for Networks: 12th International Conference, SCN 2020, Amalfi, Italy, September 14–16, 2020, Proceedings. pp. 130–149 (2020)
7. Beaver, D.: Efficient multiparty protocols using circuit randomization. In: Feigenbaum, J. (ed.) Advances in Cryptology — CRYPTO '91. pp. 420–432. Springer Berlin Heidelberg, Berlin, Heidelberg (1992)
8. Bendlin, R., Damgård, I., Orlandi, C., Zakarias, S.: Semi-homomorphic encryption and multiparty computation. In: Paterson, K.G. (ed.) Advances in Cryptology – EUROCRYPT 2011. pp. 169–188. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
9. Benhamouda, F., Camenisch, J., Krenn, S., Lyubashevsky, V., Neven, G.: Better zero-knowledge proofs for lattice encryption and their application to group signatures. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 551–572. Springer (2014)
10. Block, A.R., Gupta, D., Maji, H.K., Nguyen, H.H.: Secure computation using leaky correlations (asymptotically optimal constructions). In: Theory of Cryptography:

16th International Conference, TCC 2018, Panaji, India, November 11–14, 2018, Proceedings, Part II. p. 36–65. Springer-Verlag, Berlin, Heidelberg (2018)
11. Bourse, F., Del Pino, R., Minelli, M., Wee, H.: Fhe circuit privacy almost for free. In: Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II. pp. 62–89. Springer (2016)
12. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y.: Compressing vector ole. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. p. 896–912. CCS '18, Association for Computing Machinery, New York, NY, USA (2018), https://doi.org/10.1145/3243734.3243868
13. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Efficient pseudorandom correlation generators: Silent ot extension and more. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019. pp. 489–518. Springer International Publishing, Cham (2019)
14. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Efficient pseudorandom correlation generators from ring-lpn. In: Annual International Cryptology Conference. pp. 387–416. Springer (2020)
15. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical GapSVP. In: Annual Cryptology Conference. pp. 868–886. Springer (2012)
16. Brakerski, Z., Branco, P., Döttling, N., Pu, S.: Batch-ot with optimal rate. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 157–186. Springer (2022). https://doi.org/10.1007/978-3-031-07085-3_6, https://doi.org/10.1007/978-3-031-07085-3_6
17. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT) **6**(3), 1–36 (2014)
18. Branco, P., Döttling, N., Mateus, P.: Two-round oblivious linear evaluation from learning with errors. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13177, pp. 379–408. Springer (2022). https://doi.org/10.1007/978-3-030-97121-2_14, https://doi.org/10.1007/978-3-030-97121-2_14
19. de Castro, L., Hazay, C., Ishai, Y., Vaikuntanathan, V., Venkitasubramaniam, M.: Asymptotically quasi-optimal cryptography. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 303–334. Springer (2022)
20. de Castro, L., Juvekar, C., Vaikuntanathan, V.: Fast vector oblivious linear evaluation from ring learning with errors. In: Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography. pp. 29–41 (2021)
21. Chase, M., Dodis, Y., Ishai, Y., Kraschewski, D., Liu, T., Ostrovsky, R., Vaikuntanathan, V.: Reusable non-interactive secure computation. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019. pp. 462–488. Springer International Publishing, Cham (2019)
22. Chongchitmate, W., Ishai, Y., Lu, S., Ostrovsky, R.: Psi from ring-ole. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. p. 531–545. CCS '22, Association for Computing Machinery, New York, NY, USA (2022)

23. Couteau, G., Rindal, P., Raghuraman, S.: Silver: Silent vole and oblivious transfer from hardness of decoding structured ldpc codes. In: Annual International Cryptology Conference. pp. 502–534. Springer (2021)
24. Cramer, R., Damgård, I.: On the amortized complexity of zero-knowledge protocols. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009. pp. 177–191. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
25. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Fumy, W. (ed.) Advances in Cryptology — EUROCRYPT '97. pp. 103–118. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
26. Damgård, I., Keller, M., Larraia, E., Pastro, V., Scholl, P., Smart, N.P.: Practical covertly secure mpc for dishonest majority – or: Breaking the spdz limits. In: Crampton, J., Jajodia, S., Mayes, K. (eds.) Computer Security – ESORICS 2013. pp. 1–18. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
27. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Annual Cryptology Conference. pp. 643–662. Springer (2012)
28. Döttling, N., Ghosh, S., Nielsen, J.B., Nilges, T., Trifiletti, R.: Tinyole: Efficient actively secure two-party computation from oblivious linear function evaluation. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. p. 2263–2276. CCS '17, Association for Computing Machinery, New York, NY, USA (2017), https://doi.org/10.1145/3133956.3134024
29. Ducas, L., Stehlé, D.: Sanitization of fhe ciphertexts. In: Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I 35. pp. 294–310. Springer (2016)
30. Escudero, D., Goyal, V., Polychroniadou, A., Song, Y., Weng, C.: Superpack: Dishonest majority mpc with constant online communication. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 220–250. Springer Nature Switzerland, Cham (2023)
31. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive (2012)
32. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 169–178 (2009)
33. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. pp. 197–206 (2008)
34. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 75–92. Springer (2013)
35. Ghosh, S., Nilges, T.: An algebraic approach to maliciously secure private set intersection. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 154–185. Springer International Publishing, Cham (2019)
36. Hasler, S., Krips, T., Küsters, R., Reisert, P., Rivinius, M.: Overdrive lowgear 2.0: Reduced-bandwidth mpc without sacrifice. Cryptology ePrint Archive, Paper 2023/462 (2023), https://eprint.iacr.org/2023/462
37. Hazay, C., Ishai, Y., Marcedone, A., Venkitasubramaniam, M.: Leviosa: Lightweight secure arithmetic computation. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. p. 327–344.

CCS '19, Association for Computing Machinery, New York, NY, USA (2019), https://doi.org/10.1145/3319535.3354258
38. Ishai, Y., Paskin, A.: Evaluating branching programs on encrypted data. In: Theory of Cryptography: 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings 4. pp. 575–594. Springer (2007)
39. Ishai, Y., Prabhakaran, M., Sahai, A.: Secure arithmetic computation with no honest majority. In: Theory of Cryptography: 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings 6. pp. 294–314. Springer (2009)
40. Keller, M., Orsini, E., Scholl, P.: Mascot: Faster malicious arithmetic secure computation with oblivious transfer. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. p. 830–842. CCS '16, Association for Computing Machinery, New York, NY, USA (2016), https://doi.org/10.1145/2976749.2978357
41. Keller, M., Pastro, V., Rotaru, D.: Overdrive: making SPDZ great again. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 158–189. Springer (2018)
42. Kim, D., Lee, D., Seo, J., Song, Y.: Toward practical lattice-based proof of knowledge from hint-mlwe. In: Annual International Cryptology Conference. pp. 549–580. Springer (2023)
43. Lyubashevsky, V.: Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 598–616. Springer (2009)
44. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing **37**(1), 267–302 (2007)
45. Mouchet, C.V., Bossuat, J.P., Troncoso-Pastoriza, J.R., Hubaux, J.P.: Lattigo: A multiparty homomorphic encryption library in go. In: Proceedings of the 8th Workshop on Encrypted Computing and Applied Homomorphic Cryptography. pp. 64–70 (2020)
46. Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Proceedings of the thirty-first annual ACM symposium on Theory of computing. pp. 245–254 (1999)
47. Ostrovsky, R., Skeith, W.E.: A survey of single-database private information retrieval: Techniques and applications. In: Okamoto, T., Wang, X. (eds.) Public Key Cryptography – PKC 2007. pp. 393–411. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
48. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: Annual Cryptology Conference. pp. 80–97. Springer (2010)
49. Rindal, P., Schoppmann, P.: Vole-psi: Fast oprf and circuit-psi from vector-ole. In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021. pp. 901–930. Springer International Publishing, Cham (2021)
50. Rivest, R.L., Adleman, L., Dertouzos, M.L.: On data banks and privacy homomorphisms. Foundations of secure computation **4**(11), 169–180 (1978)
51. Rotaru, D., Smart, N.P., Tanguy, T., Vercauteren, F., Wood, T.: Actively secure setup for spdz. Journal of Cryptology **35**(1), 5 (2022)
52. Weng, C., Yang, K., Katz, J., Wang, X.: Wolverine: Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. In: 2021 IEEE Symposium on Security and Privacy (SP). pp. 1074–1091 (2021)

53. Yang, K., Sarkar, P., Weng, C., Wang, X.: Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. p. 2986–3001. CCS '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3460120.3484556

A Proofs

Proof of Lemma 10

Proof. We show that both Π_{PoK1} and Π_{PoK2} work correctly if the sender and the receiver are honest during the protocol.

• Π_{PoK1} : In the encryption phase, the ciphertext $\mathbf{c}$ generated by the receiver satisfies the honest language $\mathbf{L}_1$ with $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ and $\|e_i\|_\infty \leq B_{\rho_1}$ $(i = 0, 1, 2)$ with overwhelming probability. On the other hand, the linearity of the BFV encryption implies that $\mathbf{d}_i + \alpha_i \mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(z_i + \alpha_i x, \mathbf{f}_i + \alpha_i \mathbf{e}) = \mathtt{Enc}_{\mathbf{p}}(w_i, \mathbf{g}_i)$. It remains to show the boundedness of $\|\mathbf{g}_i\|_\infty$. Since α_i is either zero or one, we have $\|\mathbf{g}_i\|_\infty = \|\mathbf{f}_i + \alpha_i \mathbf{e}\|_\infty \leq \|\mathbf{f}_i\|_\infty + \|\mathbf{e}\|_\infty \leq B_{\rho_1} + B_{\rho_2}$ with overwhelming probability. Hence, the sender outputs accept when both the sender and the receiver honestly follow the protocol Π_{PoK1}.

• Π_{PoK2} : In the evaluation phase, the ciphertext $\mathbf{c}'$ computed by the sender satisfies the honest language $\mathbf{L}_2$ since $\mathbf{c}' = r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}')$ with $\|r\|_\infty \leq B_{\sigma_1}$ and $\|\mathbf{e}'\|_\infty \leq B_{\tau_1}$ with overwhelming probability. Similar to the case of Π_{PoK1}, we have $\mathbf{d}'_i + \alpha'_i \mathbf{c}' = v_i \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(k_i, \mathbf{g}'_i)$ by the linearity of the BFV encryption. We now check the boundedness of $\|v_i\|_\infty$ and $\|\mathbf{g}'_i\|_\infty$. Since α'_i is either 0 or 1, we have $\|v_i\|_\infty \leq \|u_i\|_\infty + \|r\|_\infty \leq B_{\sigma_1} + B_{\sigma_2}$ and $\|\mathbf{g}'_i\|_\infty \leq \|\mathbf{f}'_i\|_\infty + \|\mathbf{e}'\|_\infty \leq B_{\tau_1} + B_{\tau_2}$ with overwhelming probability. Therefore, the receiver outputs accept when both the sender and the receiver honestly follow the protocol Π_{PoK2}. □

Proof of Lemma 11

Proof. We first note that two procedures generate $v_1, \ldots, v_\ell$ in the same way. Hence, to prove the first statement, it suffices to show that $\Pr[r = x \mid v_1 = y_1, \ldots, v_\ell = y_\ell]$ in the first procedure is equal to $\Pr[\hat{r} = x \mid v_1 = y_1, \ldots, v_\ell = y_\ell]$ in the second procedure for any $x \in a + t\mathbb{Z}^n$ and $y_1, \ldots, y_\ell \in \mathbb{Z}^n$.

For any fixed $y_1, \ldots, y_\ell \in \mathbb{Z}^n$, we have

$$
\begin{aligned}
& \Pr[r = x \mid v_1 = y_1, \ldots, v_\ell = y_\ell] \\
\propto\ & \rho_{\sigma_1}(x) \cdot \prod_{i=1}^{\ell} \rho_{\sigma_2}(y_i - \alpha_i x) = \exp\left[-\pi \left(\frac{1}{\sigma_1^2} \|x\|^2 + \frac{1}{\sigma_2^2} \sum_{i=1}^{\ell} \|y_i - \alpha_i x\|^2 \right) \right] \\
\propto\ & \exp\left[-\frac{\pi}{\sigma_0'^2} \left\| x - \frac{\sigma_0'^2}{\sigma_2^2} \sum_{i=1}^{\ell} \alpha_i y_i \right\|^2 \right] = \rho_{\sigma'_0}\left(x - \frac{\sigma_0'^2}{\sigma_2^2} \sum_{i=1}^{\ell} y_i \right) \\
\propto\ & \Pr[\hat{r} = x \mid v_1 = y_1, \ldots, v_\ell = y_\ell],
\end{aligned}
$$

as desired. □

Proof of Lemma 12

Proof. For $y_1, \ldots, y_\ell \in \mathbb{Z}^n$, the probability of $v_i = y_i$ is

$$\begin{aligned}
&\Pr[v_1 = y_1, \ldots, v_\ell = y_\ell] \\
&\propto \sum_{r \in a + t\mathbb{Z}^n} \rho_{\sigma_1}(r) \cdot \prod_{i=1}^{\ell} \rho_{\sigma_2}(y_i - \alpha_i r) \\
&= \sum_{r \in a + t\mathbb{Z}^n} \exp\left[-\pi\big(\frac{1}{\sigma_1^2}\|r\|^2 + \frac{1}{\sigma_2^2}\sum_{i=1}^{\ell}\|y_i - \alpha_i r\|^2\big)\right] \\
&\propto \sum_{r \in a + t\mathbb{Z}^n} \exp\left[-\pi\big(\frac{1}{\sigma_0'^2}\big\|r - \frac{\sigma_0'^2}{\sigma_2^2}\sum_{i=1}^{\ell}\alpha_i y_i\big\|^2\big)\right] \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \cdot \rho_{\sigma_0'}(t\mathbb{Z}^n),
\end{aligned}$$

where the last inclusion is obtained from Lem. 3. This completes the proof since $\rho_{\sigma_0'}(t\mathbb{Z}^n)$ is independent of a. □

Proof of Lemma 13

Proof. We mainly focus on the soundness of PoK1, since the completeness and simulatability are exactly same to those of [42, Theorem 2]. We refer the reader to [42, Sec. 2.6] for the formal security definition of PoK.

Soundness: Since the soundness error $2^{-\ell}$ is negligible, it suffices to show that there exists an efficient knowledge extractor $\mathcal{E}$, which generates $(x', \mathbf{e}')$ satisfying $(\mathbf{c}, x', \mathbf{e}') \in \mathbf{R}_1'$ from given two accepting transcripts with different challenges. Let $(\mathbf{c}, \mathsf{comm}_0, \mathsf{chal}_0, \mathsf{resp}_0)$ and $(\mathbf{c}, \mathsf{comm}_1, \mathsf{chal}_1, \mathsf{resp}_1)$ be two accepting transcripts with different challenges denoted by $\mathsf{chal}_i = (\alpha_{i,j})_{1 \le j \le \ell}$ for $i = 0, 1$. Since $\mathsf{chal}_0 \neq \mathsf{chal}_1$, there exist an index j such that $\alpha_{0,j} \neq \alpha_{1,j}$, and we can set $\alpha_{0,j} = 0$ and $\alpha_{1,j} = 1$ without loss of generality. The extractor $\mathcal{E}$ computes $x' := w_{1,j} - w_{0,j} \pmod t$ and $\mathbf{e}' := \mathbf{g}_{1,j} - \mathbf{g}_{0,j}$, and outputs $(x', \mathbf{e}')$. Then, we can easily check that

$$\mathtt{Enc}_{\mathbf{p}}(x', \mathbf{e}') = \mathtt{Enc}_{\mathbf{p}}(w_{1,j}, \mathbf{g}_{1,j}) - \mathtt{Enc}_{\mathbf{p}}(w_{0,j}, \mathbf{g}_{0,j}) = (\mathbf{d}_j + \mathbf{c}) - \mathbf{d}_j = \mathbf{c} \pmod q,$$

and $\|\mathbf{e}'\|_\infty \le \|\mathbf{g}_{0,j}\|_\infty + \|\mathbf{g}_{1,j}\|_\infty \le 2(B_{\rho_1} + B_{\rho_2}) = B_\rho'$. Therefore, $\mathcal{E}$ is an knowledge extractor for PoK1. □

Proof of Lemma 14

Proof. If the sender outputs reject in Π_{PoK1}, the protocol is obviously secure against a malicious receiver since the receiver obtains no information from the sender. Otherwise, if the encryption phase is succeeded, we describe a simulator $\mathcal{S}_{\mathrm{PoK2}}$ for the protocol Π_{PoK2} in Fig. 7, which is in fact identical to the real protocol with input $a = 0$ and $b = y$.

From Lem. 14, the receiver has $x \in R_t$ and $\mathbf{e} \in R^3$ such that $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ and $\|\mathbf{e}\|_\infty \le B_\rho'$. If the sender is honest, the receiver's view can be written as a tuple $(\mathbf{c}', \mathsf{comm}', \mathsf{chal}', \mathsf{resp}')$ where

$$\mathbf{c}' = r\mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') = r \cdot \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e}) + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}})$$

for $y = ax + b \in R_t$ and $\tilde{\mathbf{e}} = r\mathbf{e} + \mathbf{e}' \in R^3$.

Now we prove that the output distributions of Π_{PoK2} and $\mathcal{S}_{\mathrm{PoK2}}$ are computationally indistinguishable using the hybrid argument. We define some procedures $\mathcal{H}'_0, \mathcal{H}'_1, \mathcal{H}'_2$ and $\mathcal{H}'_3$, each of which generates a transcript tr' of the form $(\mathbf{c}', \mathsf{comm}', \mathsf{chal}', \mathsf{resp}')$. In these procedures, the challenge $\mathsf{chal}' = (\alpha'_1, \ldots, \alpha'_\ell)$ is generated in the same way $\alpha'_i \leftarrow \mathcal{C}$. We will denote by ℓ' the number of $1 \le i \le \ell$ satisfying $\alpha'_i = 1$, and $\sigma'_0, \tau'_0 > 0$ the reals such that $\frac{1}{\sigma'^2_0} = \frac{1}{\sigma_1^2} + \frac{\ell'}{\sigma_2^2}$ and $\frac{1}{\tau'^2_0} = \frac{1}{\tau_1^2} + \frac{\ell'}{\tau_2^2}$. Note that $\sigma'_0 \ge \sigma_0 \ge \eta_\epsilon(t\mathbb{Z}^n)$ and $\tau'_0 \ge \tau_0 \ge \eta_\epsilon(\mathbb{Z}^n)$. In addition, $\mathsf{comm}' = (\mathbf{d}'_1, \ldots, \mathbf{d}'_\ell)$ can be obtained from chal' and resp' as $\mathbf{d}'_i = v_i \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(k_i, \mathbf{g}') - \alpha'_i \mathbf{c}' \pmod q$. Hence, we will not describe how chal' and comm' are generated in each procedure.

$\mathcal{H}'_0$: This procedure has the same distribution as the real protocol $\Pi_{\mathrm{PoK2}}(\mathbf{p}, \mathbf{c}, a, b)$.
- $\mathbf{c}' = r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}')$ for $r \leftarrow D_{a+t\mathbb{Z}^n, \sigma_1}, \mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n, \tau_1}$.
- $u_i \leftarrow D_{\mathbb{Z}^n, \sigma_2}$, $h_i \leftarrow R_t$ and $\mathbf{f}'_i \leftarrow D^3_{\mathbb{Z}^n, \tau_2}$ for $1 \le i \le \ell$.
- $v_i = u_i + \alpha'_i r$, $k_i = h_i + \alpha'_i b$ and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}'$ for $1 \le i \le \ell$. $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$.

$\mathcal{H}'_1$: This procedure is similar to $\mathcal{H}'_0$ but $\mathbf{c}'$ is generated in a different way.
- $r \leftarrow D_{a+t\mathbb{Z}^n, \sigma_1}, \mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n, \tau_1}$.
- $u_i \leftarrow D_{\mathbb{Z}^n, \sigma_2}$, $h_i \leftarrow R_t$ and $\mathbf{f}'_i \leftarrow D^3_{\mathbb{Z}^n, \tau_2}$ for $1 \le i \le \ell$.
- $v_i = u_i + \alpha'_i r$, $k_i = h_i + \alpha'_i b$ and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}'$ for $1 \le i \le \ell$. $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$.
- $\mathbf{c}' = \hat{r} \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \hat{\mathbf{e}}')$ for $\hat{r} \leftarrow D_{a+t\mathbb{Z}^n, \mu, \sigma'_0}$ and $\hat{\mathbf{e}}' \leftarrow D_{\mathbb{Z}^{3n}, \boldsymbol{\nu}, \tau'_0}$ where $\mu = (\sigma'^2_0/\sigma_2^2) \cdot \sum_{i=1}^{\ell} \alpha'_i v_i$ and $\boldsymbol{\nu} = (\tau'^2_0/\tau_2^2) \cdot \sum_{i=1}^{\ell} \alpha'_i \mathbf{g}'_i$.

$\mathcal{H}'_2$: Compared to $\mathcal{H}'_1$, the distribution of r is changed and k_i is directly sampled without using h_i. In addition, $\mathbf{c}'$ is generated in a different way so that there is no dependency on a and b in $\mathcal{H}'_2$.
- $r \leftarrow D_{t\mathbb{Z}^n, \sigma_1}, \mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n, \tau_1}$.
- $u_i \leftarrow D_{\mathbb{Z}^n, \sigma_2}$ and $\mathbf{f}'_i \leftarrow D^3_{\mathbb{Z}^n, \tau_2}$ for $1 \le i \le \ell$.
- $v_i = u_i + \alpha'_i r$, $k_i \leftarrow R_t$ and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}'$ for $1 \le i \le \ell$. $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$.
- $\mathbf{c}' = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}})$ for $\tilde{\mathbf{e}} \leftarrow D_{\mathbb{Z}^{3n}, \boldsymbol{\kappa}, \sqrt{\boldsymbol{\Sigma}}}$ for $\boldsymbol{\kappa} = \mu \cdot \mathbf{e} + \boldsymbol{\nu}$ and $\boldsymbol{\Sigma} = \sigma'^2_0 \cdot \mathbf{E}\mathbf{E}^\top + \tau'^2_0 \cdot \mathbf{I}_{3n}$, where μ and $\boldsymbol{\nu}$ are defined as in $\mathcal{H}'_1$, $\mathbf{E}_i \in \mathbb{Z}^{n \times n}$ is the negacyclic matrix corresponding to e_i for $i = 0, 1, 2$, and $\mathbf{E} = \begin{bmatrix} \mathbf{E}_0 \\ \mathbf{E}_1 \\ \mathbf{E}_2 \end{bmatrix} \in \mathbb{Z}^{3n \times n}$.

$\mathcal{H}'_3$: This is our simulator $\mathcal{S}_{\mathrm{PoK2}}$ whose distribution is the same as $\Pi_{\mathrm{PoK2}}(\mathbf{p}, \mathbf{c}, 0, y)$.
- $\mathbf{c}' = r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(y, \mathbf{e}')$ for $r \leftarrow D_{t\mathbb{Z}^n, \sigma_1}, \mathbf{e}' \leftarrow D^3_{\mathbb{Z}^n, \tau_1}$.
- $u_i \leftarrow D_{\mathbb{Z}^n, \sigma_2}$ and $\mathbf{f}'_i \leftarrow D^3_{\mathbb{Z}^n, \tau_2}$ for $1 \le i \le \ell$.
- $v_i = u_i + \alpha'_i r$, $k_i \leftarrow R_t$ and $\mathbf{g}'_i = \mathbf{f}'_i + \alpha'_i \mathbf{e}'$ for $1 \le i \le \ell$. $\mathsf{resp}' = ((v_1, k_1, \mathbf{g}'_1), \ldots, (v_\ell, k_\ell, \mathbf{g}'_\ell))$.

Claim 1: Transcripts tr' from $\mathcal{H}'_0$ and $\mathcal{H}'_1$ are statistically identical.

From Lem. 11, the distributions of $(r, v_1, \ldots, v_\ell)$ and $(\mathbf{e}', \mathbf{g}'_1, \ldots, \mathbf{g}'_\ell)$ from $\mathcal{H}'_0$ is identical to that of $(\hat{r}, v_1, \ldots, v_\ell)$ and $(\hat{\mathbf{e}}', \mathbf{g}'_1, \ldots, \mathbf{g}'_\ell)$ from $\mathcal{H}'_1$. Therefore, the distributions of $(\mathbf{c}', \mathsf{resp}')$ are identical in $\mathcal{H}'_0$ and $\mathcal{H}'_1$.

Claim 2: Transcripts tr' from $\mathcal{H}'_1$ and $\mathcal{H}'_2$ are statistically indistinguishable.

First of all, the distributions of $(v_1, \ldots, v_\ell)$ from $\mathcal{H}'_1$ and $\mathcal{H}'_2$ are within statistical distance 2ϵ from Lem. 12. Then, we observe that $\mathbf{c}' = \hat{r} \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \hat{\mathbf{e}}') = \mathtt{Enc}_{\mathbf{p}}(y, \tilde{\mathbf{e}})$ in $\mathcal{H}'_1$ where $\tilde{\mathbf{e}} = \hat{r}\mathbf{e} + \hat{\mathbf{e}}'$. From Lem. 6, the distribution of $\tilde{\mathbf{e}}$ is within statistical distance $\leq 2\epsilon$ of $D_{\mathbb{Z}^{3n}, \boldsymbol{\kappa}, \sqrt{\boldsymbol{\Sigma}}}$. Finally, it is obvious that the distribution of k_i is uniformly random over R_t in both procedures. Therefore, the statistical distance between distributions from $\mathcal{H}'_1$ and $\mathcal{H}'_2$ is bounded by 4ϵ.

Claim 3: Transcripts tr' from $\mathcal{H}'_2$ and $\mathcal{H}'_3$ are statistically indistinguishable.

We showed that $\mathcal{H}'_0$ follows the same distribution as $\mathit{\Pi}_{\mathrm{PoK2}}(\mathbf{p}, \mathbf{c}, a, b)$, which is statistically within distance at most 4ϵ of $\mathcal{H}'_2$. In addition, $\mathcal{H}'_2$ is independent from a and b but depends only on y. Hence, we can reverse our proofs for Claims 1 and 2 to show that $\mathcal{H}'_2$ is statistically within distance at most 4ϵ of $\mathit{\Pi}_{\mathrm{PoK2}}(\mathbf{p}, \mathbf{c}, 0, y)$, which follows the same distribution as the simulator $\mathcal{S}_{\mathrm{PoK2}}$.

Extractability. For the second statement, we construct an efficient knowledge extractor $\mathcal{E}'$ as follows. Suppose that $(\mathbf{c}', \mathsf{comm}'_0, \mathsf{chal}'_0, \mathsf{resp}'_0)$ and $(\mathbf{c}', \mathsf{comm}'_1, \mathsf{chal}'_1, \mathsf{resp}'_1)$ are two accepting conversations such that $\mathsf{chal}'_0 = (\alpha'_{0,1}, \ldots, \alpha'_{0,\ell}) \neq \mathsf{chal}'_1 = (\alpha'_{1,1}, \ldots, \alpha'_{1,\ell})$. Without loss of generality, we can assume that $\alpha'_{0,i} = 0$ and $\alpha'_{1,i} = 1$ for some i. Then, we have

$$(v_{1,i} - v_{0,i}) \cdot \mathbf{c} + \mathtt{Enc}(k_{1,i} - k_{0,i}, \mathbf{g}_{1,i} - \mathbf{g}_{0,i}) = (\alpha'_{1,i} - \alpha'_{0,i}) \cdot \mathbf{c}' \pmod{q},$$

and the extractor $\mathcal{E}$ can recover $(r, b, \mathbf{e}')$ by $r = v_{1,i} - v_{0,i} \in R$, $b = k_{1,i} - k_{0,i} \in R_t$ and $\mathbf{e}' = \mathbf{g}_{1,i} - \mathbf{g}_{0,i} \in R^3$ satisfying the proven language $\mathbf{c}' = r\mathbf{c} + \mathtt{Enc}(b, \mathbf{e}')$, $\|r\|_\infty \leq B'_\sigma$ and $\|\mathbf{e}'\|_\infty \leq B'_\tau$. □

Proof of Theorem 1

B Randomized Linear Evaluation when $t \nmid q$

In this section, we discuss how one can achieve a security for OLE protocol when the plaintext modulus t does not divide the ciphertext modulus q.

In fact, it can be achieved with only a slight modification to our randomized linear evaluation algorithm. Our key observation is, given a ciphertext $\mathbf{c} \in R_q$ and the public key $\mathbf{p} \in R_q$, one can simply exploit our novel linear evaluation over R_{tq} instead of R_q. Suppose that we are given a BFV encryption $\mathbf{c} = (c_0, c_1) \in R_q^2$ of $x \in R_t$. Observe that $t \cdot \mathbf{c} = (t \cdot c_0, t \cdot c_1) \in R_{tq}^2$ is a BFV encryption of x with scaling factor q. Similarly, $t \cdot \mathbf{p} \in R_{tq}^2$ is a zero-encryption and thus can serve as a public key. Then, new ciphertext modulus tq is a multiple of t, our randomized

linear evaluation algorithm can be applied to $t \cdot \mathbf{c}$ with public key $t \cdot \mathbf{p}$. Finally, the modulus can be switched back to q for consistency. Note that the resulting ciphertext of the linear evaluation is simulatable, and thus the final output is also simulatable. However, this naïve method requires modular arithmetic over $\mathbb{Z}_{tq}$ instead of $\mathbb{Z}_q$. Therefore, we provide an optimized variant of this method, which operates over $\mathbb{Z}_q$.

• $\mathtt{ModiRandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$: Given a BFV ciphertext $\mathbf{c} \in R_q^2$ and ring elements $a, b \in R_t$, sample $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}$ and $(e'_0, e'_1, e'_2) \in R^3 \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\tau}$. Compute and output $\mathbf{c}' = r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}')$ where

$$\mathbf{e}' = \left(\left\lfloor (\frac{q}{t} - \left\lfloor \frac{q}{t} \right\rceil) \cdot b + e'_0 \right\rceil, e'_1, e'_2 \right) \in R^3.$$

Now we provide a correctness and security proof below.

Lemma 15 (Correctness). *Let $\mathbf{c} \leftarrow \mathtt{Enc}_{\mathbf{p}}(x)$ for some $x \in R_t$. Then, the algorithm $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ outputs a BFV encryption of $f(x) = ax + b$ if*

$$(nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) + \frac{t+1}{2} < \frac{\Delta}{2}.$$

Proof. From the definition, we have $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ and

$$\begin{aligned} \mathbf{c}' &= r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}}(b, \mathbf{e}') \pmod q \\ &= \mathtt{Enc}_{\mathbf{p}}(y, r \cdot \mathbf{e} + \mathbf{e}') \pmod q \end{aligned}$$

where $\mathbf{e}' = (\lfloor (q/t - \lfloor q/t \rceil) \cdot b + e'_0 \rceil, e'_1, e'_2)$ for some $\mathbf{e} \leftarrow D^3_{\mathbb{Z}^n,\rho}$, $r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}$ and $(e'_0, e'_1, e'_2) \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\tau}$. Then, the only difference to $t \mid q$ case is $\lfloor (q/t - \lfloor q/t \rceil) \cdot b + e'_0 \rceil$ term, which is bounded by $1/2 \cdot t + B_\tau + 1/2 = B_\tau + (t+1)/2$. Therefore, it directly follows that the final noise is bounded by $\|e^*\|_\infty \leq (nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) + (t+1)/2 < \Delta/2$ from the proof of Lem. 5, and the resulting ciphertext $\mathbf{c}'$ decrypts to $y = ax + b$ correctly. □

We remark that the additional $(t+1)/2$ term is almost negligible compared to the first term. To prove that our modified randomized linear evaluation algorithm achieves linear circuit privacy, we first prove a modified form of Corollary 2.

Lemma 16. *Let $t > 0$ be an integer, $\mathbf{E} \in \mathbb{Z}^{3n \times n}$ a matrix, and $\sigma, \tau > 0$ reals such that*

$$\frac{1}{\sigma^2} + \frac{1}{\tau^2} \|\mathbf{E}\|^2 \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}$$

for some $0 < \epsilon \leq 1/2$. Then, for arbitrary $\mathbf{a}$, $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\nu} \in \mathbb{R}^{3n}$, the following distribution over $\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n}$

$$\mathcal{D} := \left\{ \mathbf{E}\mathbf{r} + \mathbf{e}' : \mathbf{r} \leftarrow D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}, \mathbf{e}' \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\boldsymbol{\nu},\tau} \right\}$$

is within statistical distance 4ϵ of $D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\boldsymbol{\kappa},\sqrt{\boldsymbol{\Sigma}}}$ where $\boldsymbol{\kappa} = \mathbf{E}\boldsymbol{\mu} + \boldsymbol{\nu}$ and $\boldsymbol{\Sigma} = \sigma^2 \cdot \mathbf{E}\mathbf{E}^\top + \tau^2 \cdot \mathbf{I}_{3n}$.

Proof. For $\mathbf{x} \in \mathbb{Z}^m$, the probability that $\mathcal{D}$ outputs $\mathbf{x}$ can be written as follows:

$$\begin{aligned}\mathcal{D}(\mathbf{x}) &= \Pr[\mathbf{E}\mathbf{r} + \mathbf{e}' = \mathbf{x} \mid \mathbf{r} \leftarrow D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}, \mathbf{e}' \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\boldsymbol{\nu},\tau}] \\ &= \sum_{\mathbf{y} \in \mathbf{a}+t\mathbb{Z}^n} D_{\mathbf{a}+t\mathbb{Z}^n,\boldsymbol{\mu},\sigma}(\mathbf{y}) \cdot D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\boldsymbol{\nu},\tau}(\mathbf{x} - \mathbf{E}\mathbf{y}) \\ &\propto \sum_{\mathbf{y} \in \mathbf{a}+t\mathbb{Z}^n} \rho_\sigma(\mathbf{y} - \boldsymbol{\mu}) \cdot \rho_\tau(\mathbf{x} - \boldsymbol{\nu} - \mathbf{E}\mathbf{y}).\end{aligned}$$

Then, the rest of the proof directly follows from the proof of Lemma. 6. □

Theorem 3 (Linear Circuit Privacy). *The modified randomized linear evaluation algorithm* $\mathtt{ModiRandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ *under parameters* σ *and* τ*, is circuit-private for all affine functions over* R_t *if*

$$\frac{1}{\sigma^2} + \frac{3n^2 B_\rho^2}{\tau^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2}$$

for some negligible $\epsilon > 0$*.*

Proof. Let $\mathbf{c} = \mathtt{Enc}_{\mathbf{p}}(x, \mathbf{e})$ be an encryption of $x \in R_t$ for some $\mathbf{e} = (e_0, e_1, e_2) \leftarrow D^3_{\mathbb{Z}^n,\rho}$, and $f(z) = az + b$ be an affine function over R_t for some coefficients $a, b \in R_t$. Then, we define the simulator Sim for the output ciphertext $\mathbf{c}'$ of our linear evaluation algorithm as $\mathtt{ModiRandLinEval}_{\mathbf{p}}(\mathbf{c}; 0, y)$ where $y = ax + b$. We use a hybrid argument to show that the distributions of $\mathbf{c}'$ from $\mathtt{RandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ are statistically indistinguishable. First, observe that the distribution is $\mathtt{ModiRandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ is essentially identical to the following distribution.

$$\mathcal{H}_1 = \left\{ \mathbf{c}' = \left(\left\lfloor \frac{1}{t} c_0 \right\rceil, \left\lfloor \frac{1}{t} c_1 \right\rceil \right) \;\middle|\; \begin{array}{l} (e_0', e_1', e_2') \in R^3 \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\tau}, r \leftarrow D_{a+t\mathbb{Z}^n,\sigma}, \\ (c_0, c_1) = t \cdot r \cdot \mathbf{c} + t \cdot \mathbf{p} \cdot e_2' + (q \cdot b + t \cdot e_0', t \cdot e_1') \end{array} \right\}$$

This can be shown easily, since

$$\begin{aligned}\mathbf{c}' &= r \cdot \mathbf{c} + \mathbf{p} \cdot e_2' + \left(\left\lfloor \frac{q}{t} \cdot b + e_0' \right\rceil, e_1' \right) \\ &= r \cdot \mathbf{c} + \mathbf{p} \cdot e_2' + \left(\Delta b + \left\lfloor \left(\frac{q}{t} - \Delta \right) \cdot b + e_0' \right\rceil, e_1' \right) \\ &= r \cdot \mathbf{c} + \mathtt{Enc}_{\mathbf{p}} \left(b; \mathbf{e}' = \left(\left\lfloor (\frac{q}{t} - \left\lfloor \frac{q}{t} \right\rceil) \cdot b + e_0') \right\rceil, e_1', e_2' \right) \right).\end{aligned}$$

Now, consider the following distribution:

$$\mathcal{H}_2 := \left\{ \mathbf{c}' = \left(\left\lfloor \frac{1}{t} c_0 \right\rceil, \left\lfloor \frac{1}{t} c_1 \right\rceil \right) \;\middle|\; \begin{array}{l} (\tilde{e}_0, \tilde{e}_1, \tilde{e}_2) \in R^3 \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n},\sqrt{\boldsymbol{\Sigma}}}, \\ (c_0, c_1) = t \cdot \tilde{e}_2 \cdot \mathbf{p} + (q \cdot y + t \cdot \tilde{e}_0, t \cdot \tilde{e}_1) \end{array} \right\}$$

where $\boldsymbol{\Sigma} = \sigma^2 \cdot \mathbf{E}\mathbf{E}^\top + \tau^2 \cdot \mathbf{I}_{3n}$ and $\mathbf{E}_i$ is the negacyclic matrix corresponding to e_i for $i = 0, 1, 2$ and $\mathbf{E} = \begin{bmatrix} \mathbf{E}_0 \\ \mathbf{E}_1 \\ \mathbf{E}_2 \end{bmatrix} \in \mathbb{Z}^{3n \times n}$.

Then, by Lem. 16, $\mathcal{H}_1$ and $\mathcal{H}_2$ are within statistical distance 4ϵ. This can be shown easily with an analogous logic to the proof of Lemma 1. Now, with an idential argument as above, we can show that the statistical distance of $\mathcal{H}_2$ and the following distribution $\mathcal{H}_3$ is bounded by 4ϵ since $\mathcal{H}_3$ is essentially a special case of $\mathcal{H}_1$ where $a = 0$ and $b = y$.

$$\mathcal{H}_3 = \left\{ \mathbf{c}' = \left(\left\lfloor \frac{1}{t} c_0 \right\rceil, \left\lfloor \frac{1}{t} c_1 \right\rceil \right) \;\middle|\; \begin{array}{l} (e'_0, e'_1, e'_2) \in R^3 \leftarrow D_{\frac{1}{t}\mathbb{Z}^n \times \mathbb{Z}^{2n}, \tau}, r \leftarrow D_{t\mathbb{Z}^n, \sigma}, \\ (c_0, c_1) = t \cdot r \cdot \mathbf{c} + t \cdot \mathbf{p} \cdot e'_2 + (q \cdot y + t \cdot e'_0, t \cdot e'_1) \end{array} \right\}$$

Finally, $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ outputs an identical distribution as $\mathcal{H}_3$ as discussed above. To sum up it all, we conclude that the statistical distance between two distributions $\texttt{ModiRandLinEval}_{\mathbf{p}}(\mathbf{c}; a, b)$ and $\mathsf{Sim}(s, \mathbf{p}, \mathbf{c}, y)$ is bounded by 8ϵ. □

We remark that this modified randomized linear evaluation algorithm can be easily extended to the malicious setting, by simply replacing `RandLinEval` with `ModiRandLinEval`. It is also worth noting that linear circuit privacy for BGV cryptosystem [17] can be achieved utilizing this modified randomized linear evaluation algorithm. To be precise, a BGV ciphertext can be converted into a BFV ciphertext by simply multiplying a constant $(q-1)/t \in \mathbb{Z}$ where q and t denotes the ciphertext and plaintext modulus, respectively. Then the affine function can be securely computed efficiently using our algorithm, and the resulting BFV ciphertext can be converted back to a BGV ciphertext by multiplying a constant $-t$.

C Appendix to Sec. 5

C.1 Basic Functionalities: $\mathcal{F}_{\mathsf{Rand}}$, $\mathcal{F}_{\mathsf{Commit}}$, $\mathcal{F}_{\mathsf{PKI}}$

In the following subsections, we use the following basic functionalities when describing protocols: $\mathcal{F}_{\mathsf{Rand}}$ that outputs a uniform random element from a given set, $\mathcal{F}_{\mathsf{Commit}}$ for an ideal commitment scheme, and $\mathcal{F}_{\mathsf{PKI}}$ that generates secret/public key pair for each party and distributes public key to all other parties, with respect to an underlying cryptosystem (LHE).

C.2 SPDZ Framework

The SPDZ protocol [27] is the first practical protocol that can securely compute arbitrary circuits in the presence of an *actively corrupted majority*. Under the blueprint of SPDZ, a long line of works proposed variants and optimizations [8, 26, 40, 41, 5, 36].

SPDZ is based on secret-sharing over a moderate-sized (e.g., 64-bit) finite field $\mathbb{F}$. But in order to detect malicious behaviors of adversaries, SPDZ authenticates the shares with information-theoretic (linear) MAC. That is, each party P_i keeps an additive share $[\alpha]_i$ of MAC key $\alpha \in \mathbb{F}$, while nobody knows the actual value of $\alpha = \sum_{i=1}^{n} [\alpha]_i$. And, whenever data x is additively secret-shared

as $[x]_i$'s during the protocol, secret-shares $[\alpha x]_i$ of its MAC value αx are accompanied to provide authenticity. We denote the *authenticated share* of party P_i for a secret value $x \in \mathbb{F}$ as $[\![x]\!]_i = ([x]_i, [\alpha x]_i)$.

Another important feature of SPDZ is that they take advantage of the preprocessing model. The protocols are divided into *offline* phase, which can be run before input values or circuit to compute is determined, and *online* phase. Heavy cryptographic machinery is pushed into the offline phase, and thus parties can carry out secure computation efficiently in the online phase.

Offline Phase. The goal of the offline phase is to generate a sufficient amount of *correlated randomnesses*. In particular, parties generate (i) *input mask* $[\![r]\!]$, which is an authenticated share of random r whose value is only known to a designated party, and (ii) *authenticated triple* $([\![a]\!], [\![b]\!], [\![c]\!])$, which are authenticated shares of random a, b, c subject to $a \times b = c$. A formal description of functionality $\mathcal{F}_{\mathsf{Prep}}$ for the MPC preprocessing is given in Fig. 8. For a cleaner presentation, we introduce a macro Auth (Fig. 9) and leverage it in functionality descriptions.

Online Phase. In the online phase, the parties jointly compute a circuit, consuming correlated randomnesses. For instance, party P_I can input his secret value $x_I \in \mathbb{F}$ to the circuit, consuming an input mask $[\![r]\!]$ whose value r is only known to P_I: he adds $(x_I - r)$ to his share $[r]_I$ and broadcasts $(x_I - r)$. Then, all parties can update their MAC shares accordingly. To output a result of computations, parties can reconstruct their additive secret shares, and protocol Π_{MACCheck} (Fig. 10) enables parties to check the authentication of the reconstructed value without revealing the MAC key shares.

Additions of two shared values or multiplications between a shared value and a public value can be done locally, thanks to the linearity of authenticated shares. Additions between a shared value and a public value also can be easily done: a designated party adds the public value to his share, and all parties update their MAC shares accordingly. Finally, multiplications of two shared values can be carried out with renowned Beaver's trick [7], consuming a fresh authenticated triple.

Since most protocols following the SPDZ framework share the same online phase, we omit a formal description and security proof for the online phase. For a more detailed discussion, please refer to other papers under the SPDZ framework, e.g., [26, 40].

C.3 LowGear from Circuit-Private LHE

We give a description of the LowGear protocol of Overdrive [41]. While the original LowGear protocol is based on the particular BGV scheme [17] and relies on the noise-flooding technique to guarantee its circuit-privacy, we present in a bit more generic way with our syntax of circuit-private LHE.

Functionality $\mathcal{F}_{\mathsf{Prep}}$

Parameters:

- $\mathbb{F}$: a finite field
- $P_1, \cdots, P_n$: the participating parties
- $\mathcal{A} \subset \{1, \cdots, n\}$: the set of indices of the corrupted parties

Initialize: On input (Init) from all parties,

1. Sample $[\alpha]_i \xleftarrow{\$} \mathbb{F}$ for all $i \notin \mathcal{A}$.
2. Wait for inputs $[\alpha]_i \in \mathbb{F}$ for all $i \in \mathcal{A}$ from the adversary.
3. Set $\alpha = \sum_{i=1}^{n} [\alpha]_i$.
4. Output $[\alpha]_i$ to P_i, for all $i \notin \mathcal{A}$.

Input Mask Generation: On input (Input, P_I, N) from all parties,

1. If $I \notin \mathcal{A}$, sample $[\mathbf{r}]_i \xleftarrow{\$} \mathbb{F}^N$ for all $1 \leq i \leq n$.
   If $I \in \mathcal{A}$, wait for inputs $[\mathbf{r}]_i$ for all $1 \leq i \leq n$ from the adversary.
2. Run $\mathsf{Auth}([\mathbf{r}]_1, \cdots, [\mathbf{r}]_n)$.
3. Output $([\mathbf{r}]_1, \cdots, [\mathbf{r}]_n)$ to P_I.

Triple Generation: On input (Triple, N) from all parties,

1. Sample $[\mathbf{a}]_i, [\mathbf{b}]_i \xleftarrow{\$} \mathbb{F}^N$ for all $i \notin \mathcal{A}$.
2. Wait for inputs $([\mathbf{a}]_i, [\mathbf{b}]_i, [\mathbf{c}]_i) \in \mathbb{F}^{N \times 3}$ for all $i \in \mathcal{A}$ from the adversary.
3. Set $\mathbf{a} = \sum_{i=1}^{n} [\mathbf{a}]_i$, $\mathbf{b} = \sum_{i=1}^{n} [\mathbf{b}]_i$, and $\mathbf{c} = \mathbf{a} \odot \mathbf{b}$.
4. Sample uniform random $[\mathbf{c}]_i \in \mathbb{F}$ for all $i \notin \mathcal{A}$, subject to $\mathbf{c} = \sum_{i=1}^{n} [\mathbf{c}]_i$.
5. Run $\mathsf{Auth}([\mathbf{a}]_1, \cdots, [\mathbf{a}]_n)$, $\mathsf{Auth}([\mathbf{b}]_1, \cdots, [\mathbf{b}]_n)$, and $\mathsf{Auth}([\mathbf{c}]_1, \cdots, [\mathbf{c}]_n)$.

Fig. 8. Functionality $\mathcal{F}_{\mathsf{Prep}}$

Macro $\mathsf{Auth}(\ )$

When $\mathsf{Auth}([\mathbf{x}]_1, \cdots, [\mathbf{x}]_n)$ is called, where $[\mathbf{x}]_i$'s are in $\mathbb{F}^N$, do the following:

1. Set $\mathbf{x} = \sum_{i=1}^{n} [\mathbf{x}]_i$.
2. Wait for inputs $[\alpha\mathbf{x}]_i \in \mathbb{F}^N$ for all $i \in \mathcal{A}$ from the adversary.
3. Sample uniform random $[\alpha\mathbf{x}]_i \in \mathbb{F}^N$ for all $i \notin \mathcal{A}$, subject to $\alpha\mathbf{x} = \sum_{i=1}^{n} [\alpha\mathbf{x}]_i$.
4. Output $[\![\mathbf{x}]\!]_i = ([\mathbf{x}]_i, [\alpha\mathbf{x}]_i)$ to P_i, for all $i \notin \mathcal{A}$.

Fig. 9. Macro $\mathsf{Auth}(\ )$

Protocol Π_{MACCheck}

Each party P_i with input $(x, [\alpha x]_i, [\alpha]_i) \in \mathbb{F}^3$ does the following:

1. Commit to $\delta_i = [\alpha x]_i - [\alpha]_i \cdot x$ (using $\mathcal{F}_{\mathsf{Commit}}$). Wait until all parties commit.
2. Open δ_i (using $\mathcal{F}_{\mathsf{Commit}}$). Wait until all parties open.
3. Check $\sum_{i=1}^{n} \delta_i = 0$ and abort if the check fails.

Fig. 10. Protocol Π_{MACCheck}

High-Level Sketch. Recall that the main goal of the preprocessing is to generate random authenticated triples $(\llbracket a \rrbracket, \llbracket b \rrbracket, \llbracket c \rrbracket) = ([a], [b], [c], [\alpha a], [\alpha b], [\alpha c])$ satisfying $a \times b = c$. The LowGear protocol begins with each party P_i randomly sampling $[a]_i, [b]_i \xleftarrow{\$} \mathbb{F}$ which constitute secret share of random values $a = \sum [a]_i$ and $b = \sum [b]_i$. Here, we describe a high-level idea of how to get random secret share $[c] = [a \cdot b]$ from $[a]$ and $[b]$ using LHE. We can do the similar to get $[\alpha a]$, $[\alpha b]$, $[\alpha c]$ from $[a]$, $[b]$, $[c]$, and $[\alpha]$.

First, consider the following equality.

$$a \times b = \sum_{i=1}^{n} [a]_i \cdot \sum_{i=1}^{n} [b]_i = \sum_{i=1}^{n} [a]_i \cdot [b]_i + \sum_{i \neq j} [a]_i \cdot [b]_j$$

Having this equation in mind, we can reduce the task of generating random $[c]$ into a two-party protocol between P_i and P_j to securely secret share $[a]_i \cdot [b]_j$. In other words, it is sufficient to show how P_i and P_j can respectively get random s_{ij} and r_{ij} satisfying $s_{ij} + r_{ij} = [a]_i \cdot [b]_j$, without any leakage on $[a]_i$ and $[b]_j$. This is because then each party P_i can set $[c]_i = [a]_i \cdot [b]_i + \sum_{j \neq i} (s_{ij} + r_{ji})$.

Meanwhile, LHE naturally yields such two-party protocol as follows: (1) P_i sends $\mathtt{Enc}([a]_i)$ to P_j. (2) P_j samples $r_{ij} \xleftarrow{\$} \mathbb{F}$ and returns a ciphertext encrypting $([a]_i \cdot [b]_j - r_{ij})$ via homomorphic linear evaluation. Note that circuit-privacy of the underlying LHE is crucial here. Otherwise, the response $\mathtt{ct}'_{ij}$ may leak information about $[b]_j$ or r_{ij} (and thus $[c]_j$).

Full Protocol. Although the high-level idea is quite straightforward, the full protocol is much more complex to deal with every possible malicious behavior of adversaries. In particular, we must guarantee (i) that the same $[a]$ is used when computing $[c]$ and $[\alpha a]$ and (ii) well-formednesses of ciphertexts.

For requirement (i), the so-called *sacrifice* technique (Fig. 14) is employed. And for requirement (ii), whenever a ciphertext is sent from one party to another, zero-knowledge proof of plaintext knowledge (ZKPoPK) is accompanied. It proves that the ciphertext is honestly generated by the sender. The functionality $\mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}$ for ZKPoPK is formally described in Fig. 11. As zero-knowledge proofs for lattice-based encryptions usually come with unwanted *slackness*, we

define the functionality to allow such slackness and be parameterized by a relation $\mathbf{R}$.

A full description of the LowGear's preprocessing protocol Π_{Prep} is given in Fig. 14. For cleaner presentation, we first define functionality $\mathcal{F}_{\mathsf{Auth}}$ (Fig. 12), realize it as protocol Π_{Auth} (Fig. 13), and then describe Π_{Prep} in $\mathcal{F}_{\mathsf{Auth}}$-hybrid model.

Theorem 4. *Protocol Π_{Auth} implements functionality $\mathcal{F}_{\mathsf{Auth}}$ in the $(\mathcal{F}_{\mathsf{PKI}}, \mathcal{F}_{\mathsf{Rand}}, \mathcal{F}_{\mathsf{Commit}}, \mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}})$-hybrid model in the presence of actively corrupted majority, if the underlying circuit-private LHE is enhanced-CPA secure*[14] *and also supports ciphertexts satisfying $\mathbf{R}$. That is, if $\mathbf{R}(\mathtt{ct}, \mathtt{pk}, m) = \mathtt{true}$:*

1. $\mathtt{Dec}_{\mathtt{sk}} \circ \mathtt{Eval}_{\mathtt{pk}}(\mathtt{ct}, a, b) = a \cdot m + b$ *holds for all a and b,*
2. *and $\mathtt{Eval}_{\mathtt{pk}}(\mathtt{ct}, a, b)$ can be simulated from $\mathtt{ct}$, $\mathtt{pk}$, and $a \cdot m + b$.*

Proof. Implicit in [41].

Theorem 5. *Protocol Π_{Prep} implements functionality $\mathcal{F}_{\mathsf{Prep}}$ in the $(\mathcal{F}_{\mathsf{Auth}}, \mathcal{F}_{\mathsf{PKI}}, \mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}, \mathcal{F}_{\mathsf{Rand}})$-hybrid model in the presence of actively corrupted majority, if the underlying circuit-private LHE is enhanced-CPA secure and also supports ciphertexts satisfying $\mathbf{R}$.*

Proof. Implicit in [41].

Functionality $\mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}$

On input $(\mathsf{ZKPoPK}, \mathtt{pk}, m)$ from P_i and $(\mathsf{ZKPoPK}, \mathtt{pk})$ from P_j,

1. If P_i is not corrupted, compute $\mathtt{ct} \leftarrow \mathtt{Enc}_{\mathtt{pk}}(m)$.
   Otherwise, wait for input $\mathtt{ct}$ such that $\mathbf{R}(\mathtt{ct}, \mathtt{pk}, m) = \mathtt{true}$ from adversary.
2. Output $\mathtt{ct}$ to P_j.

Fig. 11. Functionality $\mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}$

D OLE Parameter Selection

D.1 Semi-honest OLE Parameter Selection

Parameter selection for our OLE protocol starts with security parameter λ and the plaintext modulus t, both of which are given by the application. The only

[14] *Enhanced CPA security* is a bit stronger version of IND-CPA security, introduced by Overdrive [41] to give a security reduction of LowGear protocol. As in Overdrive, we use RLWE parameters with large security margins to hinder attacks on enhanced-CPA security. Please refer to [41] for a detailed discussion.

Functionality $\mathcal{F}_{\mathsf{Auth}}$

Parameters: Same as in $\mathcal{F}_{\mathsf{Prep}}$

Initialize: On input (Init) from all parties, do the same as in $\mathcal{F}_{\mathsf{Prep}}$.

Authenticate: On inputs $(\mathsf{Auth}, [\mathbf{x}]_i)$ from each party P_i's where $[\mathbf{x}]_i \in \mathbb{F}^N$,

1. Run the macro $\mathsf{Auth}([\mathbf{x}]_1, \cdots, [\mathbf{x}]_n)$.

Fig. 12. Functionality $\mathcal{F}_{\mathsf{Auth}}$

Protocol Π_{Auth}

Parameters: Same as in $\mathcal{F}_{\mathsf{Auth}}$

Initialize: On input (Init) from all parties,

1. Parties jointly call $\mathcal{F}_{\mathsf{PKI}}$. Denote P_i's public and secret key by $\mathtt{pk}_i$ and $\mathtt{sk}_i$.
2. Each party P_i samples $[\alpha]_i \xleftarrow{\$} \mathbb{F}$.
3. Each party P_i sends $\mathtt{ct}_{ij} \leftarrow \mathtt{Enc}_{\mathtt{pk}_i}([\alpha]_i)$ to P_j via $\mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}$, for all $i \neq j$.

Authenticate: On inputs $(\mathsf{Auth}, [\mathbf{x}]_i)$ from each party P_i's where $[\mathbf{x}]_i \in \mathbb{F}^N$,

Multiplication

1. Each party P_i does the following:
   (a) Sample $[x_0]_i \xleftarrow{\$} \mathbb{F}$.
   (b) Parse $[\mathbf{x}]_i = ([x_1]_i, \cdots, [x_N]_i)$ and set $[\tilde{\mathbf{x}}]_i = ([x_0]_i, [x_1]_i, \cdots, [x_N]_i)$.
2. Each (ordered) pair of distinct parties (P_i, P_j) does the following:
   (a) P_j samples $\tilde{\mathbf{r}}_{ij} = (r_{ij0}, \cdots, r_{ijN}) \xleftarrow{\$} \mathbb{F}^{N+1}$.
   (b) P_j sends $\mathtt{ct}'_{ij} \leftarrow \mathtt{Eval}_{\mathtt{pk}_i}(\mathtt{ct}_{ij}; [\tilde{\mathbf{x}}]_j, -\tilde{\mathbf{r}}_{ij})$ to P_i.
   (c) P_i computes $\tilde{\mathbf{s}}_{ij} \leftarrow \mathtt{Dec}_{\mathtt{sk}_i}(\mathtt{ct}'_{ij})$ and sets $\tilde{\mathbf{s}}_{ij} = (s_{ij0}, \cdots, s_{ijN})$.
3. Each P_i computes $[\alpha x_k]_i = [\alpha]_i[x_k]_i + \sum_{j \neq i}(s_{ijk} + r_{jik})$, for all $0 \leq k \leq N$.

MAC Check

1. Parties jointly sample $\gamma_1, \cdots, \gamma_N \xleftarrow{\$} \mathbb{F}$ (using $\mathcal{F}_{\mathsf{Rand}}$).
2. Each party P_i does the following:
   (a) Compute and broadcast $[\hat{x}]_i = [x_0]_i + \sum_{j=1}^{N} \gamma_j [x_j]_i$.
   (b) Compute $\hat{x} = \sum_{k=1}^{n} [\hat{x}]_k$.
   (c) Compute $[\alpha\hat{x}]_i = [\alpha x_0]_i + \sum_{j=1}^{N} \gamma_j [\alpha x_j]_i$.
3. Parties jointly call Π_{MACCheck} with inputs $(\hat{x}, [\alpha\hat{x}]_i, [\alpha]_i)$, respectively.
4. Each party P_i stores $[\![x_j]\!]_i = ([x_j]_i, [\alpha x_j]_i)$ as the output, for all $1 \leq j \leq N$.

Fig. 13. Protocol Π_{Auth}

Protocol Π_{Prep}

Parameters: Same as in $\mathcal{F}_{\mathsf{Prep}}$

Initialize: On input (Init) from all parties,

1. Parties jointly initialize $\mathcal{F}_{\mathsf{Auth}}$ with inputs (Init).
2. Parties jointly call $\mathcal{F}_{\mathsf{PKI}}$. Denote P_i's public and secret key by pk_i and sk_i.

Input Mask Generation: On input (Input, P_I, N) from all parties,

1. For each $1 \le i \le n$, P_I samples $[\mathbf{r}]_i \xleftarrow{\$} \mathbb{F}^N$ and sends it to P_i.
2. All parties jointly call $\mathcal{F}_{\mathsf{Auth}}$ with respective inputs $(\mathsf{Auth}, [\mathbf{r}]_i)$ from P_i.

Triple Generation: On input (Triple, N) from all parties,

Multiplication & Authentication

1. Each party P_i samples $[\mathbf{a}]_i, [\mathbf{b}]_i, [\mathbf{b}']_i \xleftarrow{\$} \mathbb{F}^N$.
2. Each (ordered) pair of distinct parties (P_i, P_j) does the following:
   (a) P_i sends $\mathtt{ct}_{ij} \leftarrow \mathtt{Enc}_{\mathtt{pk}_i}([\mathbf{a}]_i)$ to P_j via $\mathcal{F}^{\mathbf{R}}_{\mathsf{ZKPoPK}}$.
   (b) P_j samples $\mathbf{r}_{ij} \xleftarrow{\$} \mathbb{F}^N$ and sends $\mathtt{ct}'_{ij} \leftarrow \mathtt{Eval}_{\mathtt{pk}_i}(\mathtt{ct}_{ij}; [\mathbf{b}]_j, -\mathbf{r}_{ij})$ to P_i.
   (c) P_i computes $\mathbf{s}_{ij} \leftarrow \mathtt{Dec}_{\mathtt{sk}_i}(\mathtt{ct}'_{ij})$.
3. Each party P_i computes $[\mathbf{c}]_i = [\mathbf{a}]_i \odot [\mathbf{b}]_i + \sum_{j \ne i}(\mathbf{s}_{ij} + \mathbf{r}_{ji})$.
4. Repeat Step 2 and 3 with $([\mathbf{b}']_i, \mathbf{r}'_{ij}, \mathbf{s}'_{ij})$ in place of $([\mathbf{b}]_i, \mathbf{r}_{ij}, \mathbf{s}_{ij})$ to get $[\mathbf{c}']_i$.
5. Parties jointly call $\mathcal{F}_{\mathsf{Auth}}$; each party P_i sends $(\mathsf{Auth}, ([\mathbf{a}]_i, [\mathbf{b}]_i, [\mathbf{c}]_i, [\mathbf{b}']_i, [\mathbf{c}']_i))$ and receives $[\![\mathbf{a}]\!]_i$, $[\![\mathbf{b}]\!]_i$, $[\![\mathbf{c}]\!]_i$, $[\![\mathbf{b}']\!]_i$, and $[\![\mathbf{c}']\!]_i$.

Sacrifice

1. Parties jointly sample $\gamma \xleftarrow{\$} \mathbb{F}$ (using $\mathcal{F}_{\mathsf{Rand}}$).
2. Each party P_i does the following:
   (a) Compute $([\mathbf{u}]_i, [\alpha\mathbf{u}]_i) = \gamma \cdot [\![\mathbf{b}]\!]_i - [\![\mathbf{b}']\!]_i$.
   (b) Broadcast $[\mathbf{u}]_i$ and compute $\mathbf{u} = \sum_{k=1}^{n} [\mathbf{u}]_k$.
   (c) Compute $([\mathbf{v}]_i, [\alpha\mathbf{v}]_i) = \gamma \cdot [\![\mathbf{c}]\!]_i - [\![\mathbf{c}']\!]_i - \mathbf{u} \odot \mathbf{a}$.
   (d) Broadcast $[\mathbf{v}]_i$ and compute $\mathbf{v} = \sum_{k=1}^{n} [\mathbf{v}]_k$.
   (e) Check $\mathbf{v} = \mathbf{0}$ and abort if the check fails.
3. Parties jointly sample $\gamma_{\mathbf{u}}, \gamma_{\mathbf{v}} \xleftarrow{\$} \mathbb{F}^N$ (using $\mathcal{F}_{\mathsf{Rand}}$).
4. Each party P_i computes:
   (a) $w = \langle \gamma_{\mathbf{u}}, \mathbf{u} \rangle + \langle \gamma_{\mathbf{v}}, \mathbf{v} \rangle$.
   (b) $[\alpha w]_i = \langle \gamma_{\mathbf{u}}, [\alpha\mathbf{u}]_i \rangle + \langle \gamma_{\mathbf{v}}, [\alpha\mathbf{v}]_i \rangle$.
5. Parties jointly call Π_{MACCheck} with inputs $(w, [\alpha w]_i, [\alpha]_i)$, respectively.
6. Each party P_i stores $([\![\mathbf{a}]\!]_i, [\![\mathbf{b}]\!]_i, [\![\mathbf{c}]\!]_i)$ as the output.

Fig. 14. Protocol Π_{Prep}

restriction on the plaintext modulus is that $t = 1 \pmod{2n}$ where n is the degree of the polynomial modulus. This condition is necessary to support packing plaintext values into elements in the ring R_t, as explained Section 2.4. The value of n is chosen to be the smallest power of 2 allowed by the ciphertext modulus; a larger ciphertext modulus requires a larger n to maintain security. Parameter selection begins with n as a small power of 2 (usually $n = 2^{12}$ at the start), and then if the resulting ciphertext modulus is too large, the parameters are recomputed with a larger n. In practice, we never consider n beyond 2^{14}.

Given the plaintext modulus t and the polynomial modulus degree n, the next parameters to select are the Gaussian variances σ^2 and τ^2. Note that the initial error variance is set to be ρ^2, where ρ is fixed to be 3.2 [2]. The remaining two variances must satisfy Inequality 2.

Since the Gaussian sampling is the most computationally intensive piece of the protocol, we minimize these variances by setting these two terms to be equal. In other words, we have $\sigma = \sqrt{2}\eta_\epsilon(t\mathbb{Z}^n)$ and $\tau = \sqrt{6}nB_\rho\eta_\epsilon(t\mathbb{Z}^n)$.

Once we have these values, the only remaining parameter is the cipehrtext modulus q. In order to guarantee correctness, we require that

$$(nB_\rho B_\sigma + B_\tau)(1 + n + nB_\rho) < \Delta/2 = q/2t.$$

If the resulting bound on q is larger than allowed by the current value of n, then we restart the parameter selection process with $n \leftarrow 2n$. We make use of the lattice estimator [3] to verify the largest q associated with n.

D.2 Malicious OLE Parameter Selection

In order to establish the parameters for the zero-knowledge proof, several factors should be considered, including the size of the challenge space, the indistinguishability of the simulators $\mathcal{S}_{\text{PoK1}}$ and $\mathcal{S}_{\text{PoK2}}$, and the gap between the honest language and the proven language. This gap is often referred to as the *soundness slack* and it represents how much cost do we pay in order to achieve security against a malicious adversary compared to an honest adversary. Firstly, we start from setting $\ell = \lceil \frac{128}{\log(2n)} \rceil$ so that $|C|^{-\ell} \geq 2^{-128}$ and the soundness security is guaranteed for both the proofs of the ciphertext knowledge.

On the other hand, we require $\rho_0 \geq 2\eta_\epsilon(\mathbb{Z}^n)$, $\sigma_0 \geq \eta_\epsilon(t\mathbb{Z}^n)$ and $\tau_0 \geq \eta_\epsilon(\mathbb{Z}^n)$ to achieve indistinguishability of the simulators $\mathcal{S}_{\text{PoK1}}$ and $\mathcal{S}_{\text{PoK2}}$ with respect to the real protocol, respectively. For the best performance, we initially set ρ_0 to be the lowest possible value, namely $\rho_0 = 2\eta_\epsilon(\mathbb{Z}^n)$.

Next, we determine the values ρ_1 and ρ_2 to minimize the soundness slack of PoK1. To measure the soundness slack, we adopt a methodology similar to [42] since our construction is similar to theirs. Leveraging their approach, we use $B'_\rho/B_{\rho_0} = 2(\rho_1 + \rho_2)/\rho_0$ as a measurement to the soundness slack. It is worth noting that when ρ_0 is fixed, the soundness slack is solely proportional to $\rho_1 + \rho_2$. Given $\frac{1}{\rho_0^2} = \frac{1}{\rho_1^2} + \frac{\ell}{\rho_2^2}$, it can be easily shown that $\rho_1 + \rho_2$ is minimized when $\rho_1 = \sqrt{\ell^{1/3} + 1} \cdot \rho_0$ and $\rho_2 = \ell^{1/3} \cdot \rho_1$. Now, similar to the semi-honest version,

the following inequalities should hold to guarantee the passive security and the correctness of our protocol.

$$\frac{1}{\sigma_1^2} + \frac{3n^2 B_{\rho_1}^2}{\tau_1^2} \leq \frac{1}{\eta_\epsilon(t\mathbb{Z}^n)^2},$$
$$(nB_{\rho_1}B_{\sigma_1} + B_{\tau_1})(1 + n + nB_{\rho_1}) < \Delta/2 = q/2t.$$

As discussed in the parameter selection for semi-honest OLE, we can set t, σ_1 and τ_1 in relation to ρ_1. With σ_1 and τ_1 fixed, we select the error parameters $\sigma_0, \sigma_2, \tau_0$ and τ_2 to minimize the soundness slack of our scheme. Notably, the honest language L_2 and the proven language L_2' of PoK2 have almost identical form to those of PoK1, allowing us to set $\sigma_1 = \sqrt{\ell^{1/3} + 1} \cdot \sigma_0, \sigma_2 = \ell^{1/3} \cdot \sigma_1, \tau_1 = \sqrt{\ell^{1/3} + 1} \cdot \tau_0$ and $\tau_2 = \ell^{1/3} \cdot \tau_1$. Similarly, it minimizes the soundness slack for PoK2. We remark that the conditions $\sigma_0 \geq \eta_\epsilon(\mathbb{Z}^n)$ and $\tau_0 \geq \eta_\epsilon(\mathbb{Z}^n)$ are satisfied naturally.