FHE Circuit Privacy Almost For Free

Florian Bourse*, Rafaël Del Pino†, Michele Minelli‡, and Hoeteck Wee§

ENS and PSL Research University, Paris, France

{fbourse,delpino,minelli,wee}@di.ens.fr

https://crypto.di.ens.fr/

Abstract. Circuit privacy is an important property for many applications of fully homomorphic encryption. Prior approaches for achieving circuit privacy rely on superpolynomial noise flooding or on bootstrapping. In this work, we present a conceptually different approach to circuit privacy based on a novel characterization of the noise distribution. In particular, we show that a variant of the GSW FHE for branching programs already achieves circuit privacy; this immediately yields a circuit-private FHE for NC^1 circuits under the standard LWE assumption with polynomial modulus-to-noise ratio. Our analysis relies on a variant of the discrete Gaussian leftover hash lemma which states that $\mathbf{e}^\intercal \mathbf{G}^{-1}(\mathbf{v}) + \textit{small noise}$ does not depend on $\mathbf{v}$. We believe that this result is of independent interest.

Keywords: Homomorphic Encryption, Circuit Privacy, Branching Program, Noise Flooding, Learning With Errors, Rerandomization

* CNRS. Supported by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 339563 CryptoCloud)

† INRIA. Supported by SAFEcrypto (H2020 ICT-644729)

‡ Supported by the Marie Skłodowska-Curie ITN ECRYPT-NET project (H2020 643161)

§ CNRS and Columbia University. Supported in part by the ERC Project aSCEND (H2020 639554) and NSF Award CNS-1445424.

1 Introduction

A fully homomorphic encryption (FHE) scheme is an encryption scheme which supports computation on encrypted data: given a ciphertext that encrypts some data μ, one can compute a ciphertext that encrypts $f(\mu)$ for any efficiently computable function f, without ever needing to decrypt the data or know the decryption key. FHE has numerous theoretical and practical applications, the canonical one being to the problem of outsourcing computation to a remote server without compromising one's privacy. In 2009, Gentry put forth the first candidate construction of FHE based on ideal lattices [Gen09]. Since then, substantial progress has been made [vDGHV10, SS10, SV10, BV11a, BV11b, BGV12, GHS12, GSW13, BV14, AP14], offering various improvements in conceptual and technical simplicity, efficiency, security guarantees, assumptions, etc; in particular, Gentry, Sahai and Waters presented a very simple FHE (hereafter the GSW cryptosystem) based on the standard learning with errors (LWE) assumption.

Circuit privacy. An additional requirement in many FHE applications is that the evaluated ciphertext should also hide the function f, apart from what is inevitably leaked through the outcome of the computation $f(\mu)$; we refer to this requirement as *circuit privacy* [SYY99, IP07]. In the context of outsourcing computation, a server may wish to hide its proprietary algorithm from the client. Circuit privacy is also a requirement when we use FHE for low-communication secure two-party computation. In all existing FHE schemes, there is a "noise" term in the ciphertext, which is necessary for security. The noise grows and changes as a result of performing homomorphic operations and, in particular, could leak information about the function f. The main challenge for achieving FHE circuit privacy lies precisely in avoiding the leakage from the noise term in the evaluated ciphertext.

Prior works. Prior works achieve circuit privacy by essentially canceling out the noise term in the evaluated ciphertext. There are two main approaches for achieving this. The first is "noise flooding" introduced in Gentry's thesis, where we add a much larger noise at the end of the computation; in particular, the noise that is added needs to be super-polynomially larger than the noise that accumulates amidst homomorphic operations, which in turn requires that we start with a super-polynomial modulus-to-noise ratio.[1] This is a fairly mild assumption for the early constructions of FHE schemes, which required a quasi-polynomial modulus-to-noise ratio just to support homomorphic operations for circuits in NC^1 (i.e., circuits of logarithmic depth). The second is to decrypt and re-encrypt the evaluated ciphertext, also known as bootstrapping in the FHE literature. This can be achieved securely without having to know the secret key in the clear in one of two ways: (i) with the use of garbled circuits [OPP14, GHV10], and (ii) via homomorphic evaluation given an encryption of the secret key under itself [DS16], which requires an additional assumption of circular security.

Both of the prior approaches have some theoretical and practical draw-backs, if we consider FHE for NC^1 circuits (the rest of the discussion also applies to leveled FHE for general circuits). First, recall that we now have FHE for NC^1 circuits under the LWE assumption with a polynomial modulus-to-noise ratio [BV14, AP14], and we would ideally like to achieve circuit privacy under the same assumption. Relying on noise flooding for circuit privacy would require quantitatively stronger assumptions with a super-polynomial modulus-to-noise ratio, which in turn impacts practical efficiency due to the use of larger parameters. Similarly, the use of bootstrapping for circuit privacy can also be computationally expensive (indeed, the bootstrapping operation is the computational bottleneck in existing FHE schemes, cf. [DM15, HS15]). Moreover, realizing bootstrapping via an encryption of the secret key requires an additional circular security assumption, which could in turn also entail the use of larger parameters in order to account for potential weaknesses introduced by circular security. Realizing bootstrapping via garbled circuits avoids the additional assumption, but is theoretically and practically unsatisfying as it requires encoding the algebraic structure in existing FHEs as boolean computation, and sacrifices the multi-hop property in that we can no longer perform further homomorphic computation on the evaluated ciphertexts.

[1] Recall that LWE hardness depends on the modulus-to-noise ratio: the smaller the ratio, the harder the problem.

1.1 Our results

Our main result is a circuit-private FHE for NC^1 circuits —and a circuit-private leveled FHE for general circuits— under the LWE assumption with a polynomial modulus-to-noise ratio, and whose efficiency essentially matches that of existing variants of the GSW cryptosystem in [BV14, AP14]; in other words, we avoid noise flooding or bootstrapping and obtain circuit privacy almost for free!
We obtain our main result via a conceptually different approach from prior works: instead of canceling out the noise term in the evaluated ciphertext, we directly analyze the *distribution* of the noise term (prior works on FHE merely gave a bound on the noise term). Concretely, we show that adding a small noise in each step of homomorphic evaluation in the GSW cryptosystem already hides the computation itself which yields circuit privacy. Along the way, we gain better insights into the algebraic structure and the noise distribution in GSW scheme and provide new tools for analyzing noise randomization which we believe could be of independent interest.

As an immediate corollary, we obtain a two-party protocol for secure function evaluation where Alice holds x, Bob holds a branching program f, and we want Alice to learn $f(x)$ while protecting the privacy of x and f to the largest extent possible, that is, Bob learns nothing about x and Alice learns nothing about f (apart from a bound on the size of f). Our protocol achieves semi-honest security under the standard LWE assumption with polynomial hardness, and where the total communication complexity and Alice's computation are poly-logarithmic in the size of f.

The core of our analysis is a variant of the Gaussian leftover hash lemma [AGHS13, AR13]: given a "small" vector $\mathbf{e}$ and any vector $\mathbf{v}$, we have

$$\mathbf{e}^\intercal \cdot \mathbf{G}^{-1}_{\text{rand}}(\mathbf{v}) + y \approx_s e'$$

where

- $\mathbf{G}^{-1}_{\text{rand}}(\mathbf{v})$ outputs a random short vector $\mathbf{x}$ satisfying $\mathbf{G}\mathbf{x} = \mathbf{v} \mod q$ according to a discrete Gaussian with parameter $r = \tilde{O}(1)$;
- both y and e' are drawn from discrete Gaussians with parameter $O(r \cdot \|\mathbf{e}\|)$ (the norm of e' will be slightly larger than that of y).

We stress that the distribution of e' is independent of $\mathbf{v}$ and that the norm of y, e' are polynomially related to that of $\|\mathbf{e}\|$. Indeed, a similar statement is true via noise flooding, where we pick y, e' to have norm super-polynomially larger than that of $\|\mathbf{e}\|$. Using this leftover hash lemma to hide the argument of $\mathbf{G}^{-1}_{\text{rand}}(\cdot)$ is new to this work and will be crucial in proving circuit privacy.

1.2 Technical overview

We proceed with a technical overview of our construction. We build up to our main construction in three steps.

Generating fresh LWE samples. How do we generate a fresh LWE sample from a large but bounded number of samples? That is, we need to randomize $(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)$. The first idea, going back to [Reg05, GPV08, ACPS09] is to choose $\mathbf{x}$ according to a discrete Gaussian with parameter $r = \tilde{O}(1)$ and a small "smoothing" noise y from a discrete Gaussian with parameter $O(r \cdot \|\mathbf{e}\|)$ and output

$$\mathbf{A}\mathbf{x}, (\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)\mathbf{x} + y$$

The vector $\mathbf{A}\mathbf{x}$ is statistically close to uniform (by leftover hash lemma), and the error $\mathbf{e}^\intercal\mathbf{x} + y$ in the resulting sample is statistically close to a discrete Gaussian with parameter $O(r \cdot \|\mathbf{e}\|)$. We stress that the norm of y is polynomially related to that of $\mathbf{e}$, which is better than naive noise flooding. One draw-back compared to noise flooding is that the error in the new sample leaks $\|\mathbf{e}\|$. In the case of generating fresh LWE samples, we just need to repeat the process to generate many more samples than what we started out with.

Randomizing GSW ciphertexts. Next, we note that the above idea can also be used to randomize GSW ciphertexts. Recall that a GSW encryption of a message μ is of the form

$$\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + \mu \mathbf{G} \in \mathbb{Z}_q^{n \times (n \log q)}$$

where $\mathbf{s} \in \mathbb{Z}_q^n$ is the secret key and $\mathbf{G}$ is the "powers of 2" gadget matrix. We can randomize $\mathbf{C}$ to be a fresh encryption of μ by computing

$$\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{G}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}$$

where $\mathbf{G}_{\text{rand}}^{-1}(\mathbf{G})$ is chosen according to a discrete Gaussian of parameter r satisfying $\mathbf{G} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{G}) = \mathbf{G}$ and $\mathbf{y}$ is again a small smoothing noise vector. Here, we need an extension of the previous lemma showing that each coordinate in $\mathbf{e}^\intercal \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{G}) + \mathbf{y}^\intercal$ is statistically close to a discrete Gaussian; this in turn follows from an extension of the previous lemma where the vector $\mathbf{x}$ is drawn from discrete Gaussian over the coset of a lattice (cf. Lemma 3.6). And again, the norm of $\mathbf{y}$ is polynomially related to that in $\mathbf{e}$, which is better than naive noise flooding.

Scaling GSW ciphertexts. More interesting, given a constant $a \in \{0, 1\}$, we can scale a GSW encryption of μ to obtain a fresh encryption of $a \cdot \mu$ while revealing no information about a beyond what is leaked in $a \cdot \mu$. In particular, if $\mu = 0$, then the resulting ciphertext should completely hide a. To achieve this, we simply proceed as before, except we use $\mathbf{G}_{\text{rand}}^{-1}(a \cdot \mathbf{G})$ so that $\mathbf{G} \cdot \mathbf{G}_{\text{rand}}^{-1}(a \cdot \mathbf{G}) = a \cdot \mathbf{G}$. Here, we crucially rely on the fact that the error $\mathbf{e}^\intercal \cdot \mathbf{G}_{\text{rand}}^{-1}(a \cdot \mathbf{G}) + \mathbf{y}^\intercal$ in the resulting ciphertext is independent of a.

Circuit-private homomorphic evaluation. The preceding construction extends to the setting where we are given a GSW encryption $\mathbf{C}'$ of a instead of a itself, so that we output

$$\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{C}') + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}$$

We can handle homomorphic encryption as in GSW; this then readily extends to a circuit-private homomorphic evaluation for branching programs, following [BV14, AP14].

Branching programs are a relatively powerful representation model. In particular, any logarithmic space or NC^1 computation can be carried out by a family of polynomial-size branching programs. Branching programs can also directly capture several representation models often used in practice such as decision trees, OBDDs, and deterministic finite automaton.

The key insight from Brakerski and Vaikuntanathan [BV14] is that when homomorphically evaluating a branching program, we will only need to perform homomorphic additions along with homomorphic multiplications of ciphertexts $\mathbf{V}_j, \mathbf{C}_i$ where $\mathbf{V}_j$ is the encryption of an intermediate computation and $\mathbf{C}_i$ is an encryption of the input variable x_i. To obtain decryption correctness with polynomial noise growth, they computed the product as

$$\mathbf{C}_i \cdot \mathbf{G}_{\text{det}}^{-1}(\mathbf{V}_j),$$

where $\mathbf{G}_{\text{det}}^{-1}(\cdot)$ denotes the deterministic binary decomposition, cleverly exploiting the asymmetric noise growth in GSW ciphertexts and the fact that the noise in $\mathbf{C}_i$ is smaller than that in $\mathbf{V}_j$. To obtain circuit privacy, we will compute the product as

$$\mathbf{C}_i \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_j) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}_j^\intercal \end{pmatrix}.$$

Note that we made two modifications:

- First, we switched to a randomized $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$. The use of a randomized $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ for homomorphic evaluation was first introduced in [AP14], but for the very different purpose of a mild improvement in the noise growth (i.e. *efficiency*); here, we crucially exploit randomization for *privacy*.
- Next, we introduced an additional Gaussian shift $\mathbf{y}_j^\intercal$.

Interestingly, it turns out that computing the product as $\mathbf{C}_i \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_j)$ instead of $\mathbf{V}_j \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{C}_i)$ is useful not only for polynomial noise growth, but also useful for circuit privacy. Roughly speaking, the former hides which $\mathbf{V}_j$ is used, which corresponds to hiding the intermediate states that lead to the final output state, which in turn hides the branching program.

We highlight a subtlety in the analysis: $\mathbf{V}_j$ could in principle encode information about $\mathbf{C}_i$, if the variable x_i has been read prior to reaching the intermediate state encoded in $\mathbf{V}_j$, whereas to apply our randomization lemma, we crucially rely on independence between $\mathbf{C}_i$ and $\mathbf{V}_j$. The analysis proceeds by a careful induction argument showing that $\mathbf{V}_j$ looks like a fresh GSW ciphertext independent of input ciphertexts $\mathbf{C}_1, \ldots, \mathbf{C}_\ell$ apart from some dependencies on the norm of the noise terms in the input ciphertexts (see Lemma 5.4 for a precise statement). These dependencies mean that homomorphic evaluation leaks the number of times each variable appears in the branching program, but that can be easily fixed by padding the branching program.

1.3 Discussions

One draw-back of our approach is that it is specific to the GSW cryptosystem and variants there-of, whereas previous approaches based on noise flooding and bootstrapping are fairly generic; another is that we need to pad the branching program so that each variable appears the same number of times. Nonetheless, we stress that the GSW cryptosystem turns out to be ubiquitous in many applications outside of FHE, including attribute-based encryption and fully homomorphic signatures [BGG+14, GVW15]. We are optimistic that the additional insights we gained into the noise distributions of GSW ciphertexts in this work will find applications outside of FHE.

We conclude with several open problems pertaining to FHE circuit privacy. The first is to achieve circuit privacy against malicious adversaries [OPP14]: namely, the result of a homomorphic evaluation should leak no information about the circuit f, even if the input ciphertexts are maliciously generated. Our analysis breaks down in this setting as it crucially uses fresh uniform randomness in the input ciphertexts for left-over hash lemma, and the fact that the noise in the input ciphertexts are small (but does not need to be discrete Gaussian). Another is to achieve circuit-private CCA1-secure FHE [LMSV12]; here, the approach in [DS16] is not applicable since giving out an encryption of the secret key violates CCA1-security. A third open problem is to extend the techniques in this work to other FHE schemes, such as those in [BV11a, DM15, HS15].

2 Preliminaries

In this section we clarify our notation and recall some definitions, problems and lemmas that we are going to use throughout the paper.

Notation. We denote the real numbers by $\mathbb{R}$, the integers by $\mathbb{Z}$, the integers modulo some q by $\mathbb{Z}_q$, and let $[N]$ indicate the integer numbers $\{1, \ldots, N\}$. Throughout the paper we use λ to denote the security parameter. We say that a function is *negligible*, and we denote it by $\text{negl}(\lambda)$, if it is a $f(\lambda) = o(\lambda^{-c})$ for every fixed constant c. We also say that a probability is *overwhelming* if it is $1 - \text{negl}(\lambda)$.
Vectors are denoted by lower-case bold letters (e.g., $\mathbf{v}$) and are always in column form ($\mathbf{v}^\intercal$ is a row vector), while matrices are indicated by upper-case bold letters. We let $(\mathbf{a}, \mathbf{b})$ denote the vector obtained by concatenating the two vectors, i.e. $\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix}$. We also write $(\mathbf{v}_1 \mid \mathbf{v}_2 \mid \ldots \mid \mathbf{v}_k)$ to denote the matrix whose columns are the vectors $\mathbf{v}_i$. Unless otherwise stated, the norm $\|\cdot\|$ considered in this paper is the ℓ_2 norm and log denotes the base-2 logarithm, while ln denotes the natural logarithm.
Given two distributions X, Y over a finite or countable domain D, their statistical distance is defined as $\Delta(X, Y) = \frac{1}{2}\sum_{v \in D} |X(v) - Y(v)|$. We say that two distributions are *statistically close* (denoted by $\approx_s$) if their statistical distance is $\text{negl}(\lambda)$. Given a set A, we will write $a \xleftarrow{\$} A$ to indicate that a is sampled from A uniformly at random. If $\mathcal{D}$ is a probability distribution, we will write $d \leftarrow \mathcal{D}$ to indicate that d is sampled according to the distribution $\mathcal{D}$. Following [MP12], we denote by $\mathbf{G}$ the gadget matrix, i.e. $\mathbf{G} = \mathbf{g}^\intercal \otimes \mathbf{I}_n$, where $\mathbf{g}$ is the vector $\left(1, 2, 4, \ldots, 2^{\lceil \log q \rceil - 1}\right)$, for given parameters n, q.

Lattices. A m-dimensional lattice Λ is a discrete additive subgroup of $\mathbb{R}^m$. For an integer $k < m$ and a rank k matrix $\mathbf{B} \in \mathbb{R}^{m \times k}$, $\Lambda(\mathbf{B}) = \{\mathbf{Bx} \in \mathbb{R}^m \mid \mathbf{x} \in \mathbb{Z}^k\}$ is the lattice generated by the columns of $\mathbf{B}$. We will let $\Lambda_q^\perp(\mathbf{B})$ denote $\{\mathbf{v} \in \mathbb{Z}^m \mid \mathbf{B}^\intercal \mathbf{v} = \mathbf{0} \mod q\}$.

Gaussian function. For any $\alpha > 0$, the spherical Gaussian function with parameter α (omitted if 1) is defined as $\rho_\alpha(\mathbf{x}) = \exp\left(-\pi \|\mathbf{x}\|^2 / \alpha^2\right)$, for any $\mathbf{x} \in \mathbb{R}^m$. Given a lattice $\Lambda \subseteq \mathbb{R}^m$, a parameter $r \in \mathbb{R}$ and a vector $\mathbf{c} \in \mathbb{R}^m$ the spherical Gaussian distribution with parameter r and support $\Lambda + \mathbf{c}$ is defined as

$$\mathcal{D}_{\Lambda+\mathbf{c},r}(\mathbf{x}) = \frac{\rho_r(\mathbf{x})}{\rho_r(\Lambda + \mathbf{c})}, \quad \forall \mathbf{x} \in \Lambda + \mathbf{c}$$

where $\rho_r(\Lambda + \mathbf{c})$ denotes $\sum_{\mathbf{x} \in \Lambda + \mathbf{c}} \rho_r(\mathbf{x})$. Note that $\rho_r(\mathbf{x}) = \rho\left(r^{-1}\mathbf{x}\right)$.

We now give an algorithm for the randomized bit decomposition $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$.

Definition 2.1 (The $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ algorithm, adapted from [MP12], [AP14, Claim 3.1]). *There is a randomized, efficiently computable function $\mathbf{G}_{\text{rand}}^{-1}(\cdot) : \mathbb{Z}_q^n \to \mathbb{Z}^m$ such that $\mathbf{x} \leftarrow \mathbf{G}_{\text{rand}}^{-1}(\mathbf{v})$ is drawn from a distribution close to a Gaussian with parameter $r = \tilde{O}(1)$ conditioned on $\mathbf{Gx} = \mathbf{v} \mod q$, i.e. $\mathbf{G}_{\text{rand}}^{-1}(\mathbf{v})$ outputs a sample from the distribution $\mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{G}_{\text{det}}^{-1}(\mathbf{v}),r}$ where $\mathbf{G}_{\text{det}}^{-1}(\cdot)$ denotes (deterministic) bit decomposition. We will also write $\mathbf{X} \leftarrow \mathbf{G}_{\text{rand}}^{-1}(\mathbf{M})$ to denote that the columns of the matrix $\mathbf{X} \in \mathbb{Z}^{m \times p}$ are obtained by applying the algorithm separately to each column of a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times p}$.*

In particular, using the exact sampler in [BLP+13, Section 5] (which is a variant of the algorithm presented in [GPV08]), $\mathbf{G}_{\text{rand}}^{-1}(\mathbf{v})$ outputs a sample from the discrete Gaussian

$$\mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{G}_{\text{det}}^{-1}(\mathbf{v}),r}$$

Next, we recall the definition of the *smoothing parameter* of a lattice from [MR04]. Intuitively, this parameter provides the width beyond which the discrete Gaussian measure on a lattice behaves like a continuous one.

Definition 2.2 (Smoothing parameter). *For a lattice $\Lambda \subseteq \mathbb{Z}^m$ and positive real $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ is the smallest real $r > 0$ such that $\rho_{1/r}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \varepsilon$, where $\Lambda^* := \{\mathbf{x} \in \mathbb{R}^m \mid \mathbf{x}^\intercal \Lambda \subseteq \mathbb{Z}\}$.*

We will also need the following probability results.

Lemma 2.3 (Simplified version of [Pei10, Theorem 3.1]). *Let $\varepsilon > 0$, $r_1, r_2 > 0$ be two Gaussian parameters, and $\Lambda \subseteq \mathbb{Z}^m$ be a lattice. If $\frac{r_1 r_2}{\sqrt{r_1^2 + r_2^2}} \geq \eta_\varepsilon(\Lambda)$, then*

$$\Delta(\mathbf{y}_1 + \mathbf{y}_2, \mathbf{y}') \leq 8\varepsilon$$

where $\mathbf{y}_1 \leftarrow \mathcal{D}_{\Lambda,r_1}$, $\mathbf{y}_2 \leftarrow \mathcal{D}_{\Lambda,r_2}$, and $\mathbf{y}' \leftarrow \mathcal{D}_{\Lambda,\sqrt{r_1^2+r_2^2}}$.

Lemma 2.4 ([AP14, Lemma 2.1]). *There exists a universal constant $C > 0$, such that*

$$\Pr\left[\|\mathbf{x}\| > Cr\sqrt{m}\right] \leq 2^{-\Omega(m)}$$

where $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m,r}$.

Next, we recall the LWE problem and its hardness assumption.

The LWE problem and assumption. The learning with errors (LWE) problem was introduced by Regev in [Reg05] as a generalization of "learning parity with noise". Let $q \geq 2$, n and $m = \text{poly}(n)$ be positive integers, and let χ be a probability distribution over $\mathbb{Z}_q$. We define the following advantage function for an adversary $\mathcal{A}$:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,q,\chi}} := \left|\Pr\left[\mathcal{A}\left(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathbf{A}, \mathbf{u}\right) = 1\right]\right|$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$. The LWE assumption asserts that for any PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,q,\chi}}$ is $\text{negl}(n)$.

Finally, we recall the definition of a homomorphic encryption scheme, evaluation correctness and semantic security.

Homomorphic encryption scheme. A homomorphic (secret-key) encryption scheme $\mathcal{E} = (\mathcal{E}.\mathsf{Setup}, \mathcal{E}.\mathsf{Encrypt}, \mathcal{E}.\mathsf{Decrypt}, \mathcal{E}.\mathsf{Eval})$ is a quadruple of PPT algorithms as follows:

- $\mathcal{E}.\mathsf{Setup}\left(1^\lambda\right)$: given the security parameter λ, outputs a secret key sk and an evaluation key evk
- $\mathcal{E}.\mathsf{Encrypt}\left(sk, \mu\right)$: using the secret key sk, encrypts a message $\mu \in \{0,1\}$ into a ciphertext c and outputs c
- $\mathcal{E}.\mathsf{Decrypt}\left(sk, c\right)$: using the secret key sk, decrypts a ciphertext c to recover a message $\mu \in \{0,1\}$
- $\mathcal{E}.\mathsf{Eval}\left(evk, f, c_1, \ldots, c_\ell\right)$: using the evaluation key evk, applies a function $f : \{0,1\}^\ell \to \{0,1\}$ to ciphertexts $c_1, \ldots, c_\ell$ and outputs a ciphertext c_f

Evaluation correctness. We say that the $\mathcal{E}.\mathsf{Eval}$ algorithm correctly evaluates all functions in $\mathcal{F}$ if, for any function $f \in \mathcal{F} : \{0,1\}^\ell \to \{0,1\}$ and respective inputs $x_1, \ldots, x_\ell \in \{0,1\}^\ell$ it holds that

$$\Pr\left[\mathcal{E}.\mathsf{Decrypt}\left(sk, \mathcal{E}.\mathsf{Eval}\left(evk, f, c_1, \ldots, c_\ell\right)\right) = f\left(x_1, \ldots, x_\ell\right)\right] = 1 - \text{negl}\left(\lambda\right)$$

where $sk \leftarrow \mathcal{E}.\mathsf{Setup}\left(1^\lambda\right)$ and $c_i \leftarrow \mathcal{E}.\mathsf{Encrypt}\left(sk, x_i\right)$.

Semantic security. A secret key encryption scheme $\mathcal{E}$ is said to be semantically secure (or IND-CPA secure) if any PPT adversary $\mathcal{A}$ cannot distinguish between encryptions of two known plaintexts. More formally, let $sk \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$ and $\mathcal{O}_b\left(\mu_0, \mu_1\right) = \mathcal{E}.\mathsf{Encrypt}\left(sk, \mu_b\right)$ for $b \in \{0,1\}$. Then $\mathcal{E}$ is IND-CPA secure if

$$\left|\Pr\left[\mathcal{A}^{\mathcal{O}_0}\left(1^\lambda\right) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}_1}\left(1^\lambda\right) = 1\right]\right| = \text{negl}\left(\lambda\right)$$

where the probability is taken over the internal coins of $\mathcal{E}.\mathsf{Setup}$, $\mathcal{E}.\mathsf{Encrypt}$ and $\mathcal{A}$.

3 Core Randomization Lemma

Note that throughout the rest of the paper we set q to be a power of 2, and $m = n \log q$. We discuss the use of a modulus q that is not a power of 2 in Section 5.4.
The goal of this Section is to establish the following lemma:

Lemma 3.1 (Core randomization lemma). *Let $\varepsilon, \varepsilon' > 0$, $r > \eta_\varepsilon(\Lambda_q^\perp\left(\mathbf{G}^\intercal\right))$ be a Gaussian parameter. For any $\mathbf{e} \in \mathbb{Z}_q^m$, $\mathbf{v} \in \mathbb{Z}_q^n$, if $r \geq \max\left(4\left(\left(1-\varepsilon\right)\left(2\varepsilon'\right)^2\right)^{-\frac{1}{m}}, \sqrt{5}(1+\|\mathbf{e}\|)\sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}}\right)$, then*

$$\Delta\left(\left(\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{e}^\intercal\mathbf{x} + y\right), \left(\mathbf{A}, \mathbf{u}, e'\right)\right) < \varepsilon' + 2\varepsilon$$

where $\mathbf{x} \leftarrow \mathbf{G}_{\text{rand}}^{-1}\left(\mathbf{v}\right)$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n-1)\times m}$, $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{n-1}$, $y \leftarrow \mathcal{D}_{\mathbb{Z},r}$ and $e' \leftarrow \mathcal{D}_{\mathbb{Z},r\sqrt{1+\|\mathbf{e}\|^2}}$.
Asymptotically, $r = \tilde{\Theta}(\|\mathbf{e}\|\sqrt{\lambda})$ is enough to obtain negligible statistical distance.

Remark 1 (on the necessity of randomization). We note here that the use of randomization in $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ and the shift are both necessary.

First, the shift is necessary for both distributions to have the same support. For example, $\mathbf{e}^{\intercal}\mathbf{G}_{\text{rand}}^{-1}((1,0,\ldots,0))$ and $\mathbf{e}^{\intercal}\mathbf{G}_{\text{rand}}^{-1}(\mathbf{0})$ might lie in two different cosets of the lattice $\mathbf{e}^{\intercal}\Lambda_q^{\perp}(\mathbf{G}^{\intercal})$, depending on the value of $\mathbf{e}$: if the first coordinate of $\mathbf{e}$ is odd and all the others are even, then $\mathbf{e}^{\intercal}\mathbf{G}_{\text{rand}}^{-1}((1,0,\ldots,0))$ will be odd, while $\mathbf{e}^{\intercal}\mathbf{G}_{\text{rand}}^{-1}(\mathbf{0})$ will be even, for a q even. The shift by a Gaussian over $\mathbb{Z}$ ensures that the support of the two distributions is $\mathbb{Z}$. Proving that $\mathbf{e}^{\intercal}\Lambda_q^{\perp}(\mathbf{G}^{\intercal}) = \mathbb{Z}$ with overwhelming probability over the choice of $\mathbf{e}$ is still an open question that would remove the necessity of the shift, thus proving circuit privacy for standard GSW only using randomized $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$.

Finally, the randomization of $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ is necessary for both distributions to have the same center. Using the same example, $\mathbf{e}^{\intercal}\mathbf{G}_{\text{det}}^{-1}((1,0,\ldots,0)) + y$ and $\mathbf{e}^{\intercal}\mathbf{G}_{\text{det}}^{-1}(\mathbf{0}) + y$ would be two Gaussians, centered respectively on e_1 (the first coordinate of $\mathbf{e}$) and on 0. Instead, using the randomized algorithm $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$, the center of both distributions will be 0.

3.1 Additional preliminaries

Before proving Lemma 3.1, we need to recall some additional results.

Lemma 3.2 ([MR07, Lemma 3.3]). *Let Λ be any rank-m lattice and ε be any positive real. Then*

$$\eta_{\varepsilon}(\Lambda) \leq \lambda_m(\Lambda) \cdot \sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}}$$

where $\lambda_m(\Lambda)$ is the smallest R such that the ball $\mathcal{B}_R$ centered in the origin and with radius R contains m linearly independent vectors of Λ.

Lemma 3.3 ([GPV08, Corollary 2.8]). *Let $\Lambda \subseteq \mathbb{Z}^m$ be a lattice, $0 < \varepsilon < 1$, $r > 0$. For any vector $\mathbf{c} \in \mathbb{R}^m$, if $r \geq \eta_{\varepsilon}(\Lambda)$, then we have*

$$\rho_r(\Lambda + \mathbf{c}) \in \left[\frac{1-\varepsilon}{1+\varepsilon},\ 1\right] \cdot \rho_r(\Lambda)$$

Lemma 3.4 ([Reg05, Claim 3.8]). *Let $\Lambda \subseteq \mathbb{Z}^m$ be any lattice, $\mathbf{c} \in \mathbb{R}^m$, $\varepsilon > 0$ and $r \geq \eta_{\varepsilon}(\Lambda)$. Then*

$$\rho_r(\Lambda + \mathbf{c}) \in \frac{r^m}{\det(\Lambda)}(1 \pm \varepsilon)$$

Generalized leftover hash lemma. We state here a simplified version of the generalized leftover hash lemma which is sufficient for our use. The min-entropy of a random variable X is defined as

$$H_{\infty}(X) = -\log\left(\max_x \Pr[X = x]\right)$$

Lemma 3.5 (Generalized leftover hash lemma [DRS04]).
Let $\mathbf{e}$ be any random variable over $\mathbb{Z}_q^m$ and $f : \mathbb{Z}_q^m \to \mathbb{Z}_q^k$. Then

$$\Delta((\mathbf{X}\mathbf{e}, \mathbf{X}, f(\mathbf{e})), (\mathbf{r}, \mathbf{X}, f(\mathbf{e}))) \leq \frac{1}{2}\sqrt{q^{n+k} \cdot 2^{-H_{\infty}(\mathbf{e})}}\ .$$

where $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^n$.

3.2 Proof of Lemma 3.1

We first prove that given $\mathbf{e}$, the new error term $\mathbf{e}^{\intercal}\mathbf{x} + y$ is indeed a Gaussian with parameter $r\sqrt{1 + \|\mathbf{e}\|^2}$. This proof is inspired by [AR13], which in turn is an improvement of [AGHS13], but it is different in two aspects: on one hand, in [AR13] the proof is done for the specific case where $\mathbf{x}$ is drawn from a Gaussian over a coset of $\mathbb{Z}^m$; on the other hand, they consider the more general case of an ellipsoidal Gaussian distribution.

Lemma 3.6 (adapted from [AR13, Lemma 3.3]). *Let $\varepsilon, r > 0$. For any $\mathbf{e} \in \mathbb{Z}^m$, $\mathbf{c} \in \mathbb{R}^m$, if $r \geq \sqrt{5}(1 + \|\mathbf{e}\|) \cdot \sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}}$, then*

$$\Delta\left(\mathbf{e}^\intercal \mathbf{x} + y, e'\right) < 2\varepsilon$$

where $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}$, $y \leftarrow \mathcal{D}_{\mathbb{Z},r}$, and $e' \leftarrow \mathcal{D}_{\mathbb{Z},r\sqrt{1+\|\mathbf{e}\|^2}}$.

Asymptotically, $r = \tilde{\Theta}(\|\mathbf{e}\|\sqrt{\lambda})$ is enough to obtain negligible statistical distance.

Proof. Let $\widehat{\mathbf{e}} = (\mathbf{e}, 1) \in \mathbb{Z}^{m+1}, \widehat{\mathbf{c}} = (\mathbf{c}, 0) \in \mathbb{Z}^{m+1}$ and $\widehat{\Lambda} = \Lambda_q^\perp(\mathbf{G}^\intercal) \times \mathbb{Z}$, we want to show that

$$\Delta\left(\widehat{\mathbf{e}}^\intercal \mathcal{D}_{\widehat{\Lambda}+\widehat{\mathbf{c}},r}, \mathcal{D}_{\mathbb{Z},\|\widehat{\mathbf{e}}\|r}\right) \leq 2\varepsilon$$

The support of $\widehat{\mathbf{e}}^\intercal \mathcal{D}_{\widehat{\Lambda}+\widehat{\mathbf{c}},r}$ is $\widehat{\mathbf{e}}^\intercal\widehat{\Lambda} + \widehat{\mathbf{e}}^\intercal\widehat{\mathbf{c}} = \mathbf{e}^\intercal\Lambda_q^\perp(\mathbf{G}^\intercal) + \mathbb{Z} + \mathbf{e}^\intercal\mathbf{c} = \mathbb{Z}$. Fix some $z \in \mathbb{Z}$. The probability mass assigned to z by $\widehat{\mathbf{e}}^\intercal \mathcal{D}_{\widehat{\Lambda}+\widehat{\mathbf{c}},r}$ is proportional to $\rho_r(\mathcal{L}_z)$, where

$$\mathcal{L}_z = \left\{\mathbf{v} \in \widehat{\Lambda} + \widehat{\mathbf{c}} : \widehat{\mathbf{e}}^\intercal\mathbf{v} = z\right\}$$

We define the lattice $\mathcal{L} = \left\{\mathbf{v} \in \widehat{\Lambda} : \widehat{\mathbf{e}}^\intercal\mathbf{v} = 0\right\}$; note that $\mathcal{L}_z = \mathcal{L} + \mathbf{w}_z$ for any $\mathbf{w}_z \in \mathcal{L}_z$. Let $\mathbf{u}_z = \frac{z}{\|\widehat{\mathbf{e}}\|^2 r}\widehat{\mathbf{e}}$, then $\mathbf{u}_z$ is clearly proportional to $\widehat{\mathbf{e}}$. Observe that $\mathbf{u}_z$ is orthogonal to $r^{-1}\mathcal{L}_z - \mathbf{u}_z$, indeed for any $\mathbf{t} \in r^{-1}\mathcal{L}_z$ we have $\widehat{\mathbf{e}}^\intercal(\mathbf{t} - \mathbf{u}_z) = 0$. From this we have $\rho(\mathbf{t}) = \rho(\mathbf{u}_z) \cdot \rho(\mathbf{t} - \mathbf{u}_z)$, and by summing for $\mathbf{t} \in r^{-1}\mathcal{L}_z$:

$$\rho(r^{-1}\mathcal{L}_z) = \rho(\mathbf{u}_z) \cdot \rho(r^{-1}\mathcal{L}_z - \mathbf{u}_z)$$

Observe that we have $r^{-1}\mathcal{L}_z - \mathbf{u}_z = r^{-1}(\mathcal{L} - \mathbf{c}')$ for some $\mathbf{c}'$ in the vector span of the lattice $\mathcal{L}$ (because $\mathcal{L}_z - r\mathbf{u}_z = \mathcal{L} + \mathbf{w}_z - r\mathbf{u}_z$ and $\widehat{\mathbf{e}}^\intercal(\mathbf{w}_z - r\mathbf{u}_z) = 0$). Thus using Lemmas 3.3 and 3.7 with $r \geq \sqrt{5}(1 + \|\mathbf{e}\|) \cdot \sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}} \geq \eta_\varepsilon(\mathcal{L})$, we obtain

$$\begin{aligned}
\rho(r^{-1}\mathcal{L}_z) &= \rho(\mathbf{u}_z) \cdot \rho_r(\mathcal{L} - \mathbf{c}') \\
&\in \left[\frac{1-\varepsilon}{1+\varepsilon},\ 1\right] \cdot \rho_r(\mathcal{L}) \cdot \rho(\mathbf{u}_z) \\
&= \left[\frac{1-\varepsilon}{1+\varepsilon},\ 1\right] \cdot \rho_r(\mathcal{L}) \cdot \rho\left(\frac{z}{\|\widehat{\mathbf{e}}\|^2 r}\widehat{\mathbf{e}}\right) \\
&= \left[\frac{1-\varepsilon}{1+\varepsilon},\ 1\right] \cdot \rho_r(\mathcal{L}) \cdot \rho_{\|\widehat{\mathbf{e}}\|r}(z)
\end{aligned}$$

This implies that the statistical distance between $\widehat{\mathbf{e}}^\intercal \mathcal{D}_{\widehat{\Lambda}+\widehat{\mathbf{c}},r}$ and $\mathcal{D}_{\mathbb{Z},\|\widehat{\mathbf{e}}\|r}$ is at most $1 - \frac{1-\varepsilon}{1+\varepsilon} \leq 2\varepsilon$. □

In order to conclude the previous proof, we now give a bound on the smoothing parameter of the lattice $\mathcal{L}$.

Lemma 3.7. *Let $\varepsilon > 0$. For any $\mathbf{e} \in \mathbb{Z}^m$, let $\mathcal{L}$ be as defined in Lemma 3.6. Then we have:*

$$\eta_\varepsilon(\mathcal{L}) \leq \sqrt{5}(1 + \|\mathbf{e}\|) \cdot \sqrt{\frac{\ln\left(2m\left(1 + 1/\varepsilon\right)\right)}{\pi}}$$

Proof. We use Lemma 3.2 to bound the smoothing parameter of $\mathcal{L}$. Since $\widehat{\Lambda} = \Lambda_q^\perp(\mathbf{G}^\intercal) \times \mathbb{Z}$ is of dimension $m + 1$ and $\mathcal{L}$ is the sublattice of $\widehat{\Lambda}$ made of the vectors that are orthogonal to $\mathbf{e}$, we have that $\mathcal{L}$ is of dimension m. We thus exhibit m independent short vectors of $\mathcal{L}$ to obtain an upper bound on $\lambda_m(\mathcal{L})$. We first define the matrix

$$\overline{\mathbf{B}} = \begin{pmatrix} 2 & & \\ -1 & \ddots & \\ & \ddots & \ddots \\ & & -1 \quad 2 \end{pmatrix} \in \mathbb{Z}^{(\log q)\times(\log q)}$$

and remark that it is a basis for the lattice $\Lambda_q^\perp(\mathbf{g}^\intercal)$. The lattice $\widehat{\Lambda}$ is then generated by the columns of the matrix:

$$\mathbf{B} = (\mathbf{b}_1 \mid \ldots \mid \mathbf{b}_{m+1}) = \left(\begin{array}{c|c} \mathbf{I}_n \otimes \overline{\mathbf{B}} & \mathbf{0} \\ \hline \mathbf{0}^\intercal & 1 \end{array}\right) \in \mathbb{Z}^{(m+1)\times(m+1)}$$

For $k \leq m$ let $\mathbf{u}_k = \mathbf{b}_k - \mathbf{b}_{m+1} \cdot \widehat{\mathbf{e}}^\intercal \mathbf{b}_k$, since $\widehat{\mathbf{e}}^\intercal \mathbf{b}_{m+1} = 1$ we directly have $\widehat{\mathbf{e}}^\intercal \mathbf{u}_k = 0$ and thus $\mathbf{u}_k \in \mathcal{L}$. The vectors $\mathbf{u}_1, \ldots, \mathbf{u}_m$ are linearly independent since $\text{span}(\mathbf{u}_1, \ldots, \mathbf{u}_m, \mathbf{b}_{m+1}) = \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_m, \mathbf{b}_{m+1}) = \mathbb{R}^{m+1}$ (which comes from the fact that $\mathbf{B}$ is a basis of an $(m+1)$-dimensional lattice). We now bound the norm of $\mathbf{u}_k$:

$$\begin{aligned} \|\mathbf{u}_k\| &\leq \|\mathbf{b}_k\| + \|\mathbf{b}_{m+1}\| \, \|\mathbf{e}\| \, \|\mathbf{b}_k\| \\ &= \sqrt{5}(1 + \|\mathbf{e}\|) \end{aligned}$$

Note that $|\widehat{\mathbf{e}}^\intercal \mathbf{b}_k| \leq \|\mathbf{e}\| \, \|\mathbf{b}_k\|$ since the last coefficient of $\mathbf{b}_k$ is 0. Finally we obtain $\lambda_m(\mathcal{L}) \leq \max_{k \leq m} \|\mathbf{u}_k\| \leq \sqrt{5}(1 + \|\mathbf{e}\|)$ and the result. □

The final proof of Lemma 3.1 will necessitate a call to the leftover hash lemma, so before continuing we analyze the min-entropy of $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}$.

Lemma 3.8. *Let $\varepsilon > 0$, $r \geq \eta_\varepsilon\left(\Lambda_q^\perp(\mathbf{G}^\intercal)\right)$. For any $\mathbf{c} \in \mathbb{R}^m$, we have*

$$H_\infty\left(\mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}\right) \geq \log(1-\varepsilon) + m\log(r) - m$$

Proof. For any $\mathbf{v} \in \Lambda_q^\perp(\mathbf{G}^\intercal) + \mathbf{c}$

$$\begin{aligned} \mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}(\mathbf{v}) &\leq \mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}(\mathbf{v}_0), \text{ for } \mathbf{v}_0 \text{ the point of } \Lambda_q^\perp(\mathbf{G}^\intercal) + \mathbf{c} \text{ closest to } \mathbf{0} \\ &= \frac{\rho_r(\mathbf{v}_0)}{\rho_r(\Lambda_q^\perp(\mathbf{G}^\intercal) + \mathbf{c})} \\ &\leq \frac{1}{\rho_r(\Lambda_q^\perp(\mathbf{G}^\intercal) + \mathbf{c})}, \text{ since } \rho_r(\mathbf{v}_0) < 1 \\ &\leq (1-\varepsilon)\frac{r^m}{\det\left(\Lambda_q^\perp(\mathbf{G}^\intercal)\right)}, \text{ by Lemma 3.4 since } r \geq \eta_\varepsilon\left(\Lambda_q^\perp(\mathbf{G}^\intercal)\right) \end{aligned}$$

The lattice $\Lambda_q^\perp(\mathbf{G}^\intercal)$ is generated by the basis $\mathbf{I}_n \otimes \overline{\mathbf{B}}$, with $\overline{\mathbf{B}}$ defined as above, which has determinant $\left(2^{\log q}\right)^n = 2^m$. The result follows:

$$H_\infty\left(\mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal)+\mathbf{c},r}\right) \geq \log(1-\varepsilon) + m\log(r) - m$$

□

We are now ready to prove Lemma 3.1.

Proof. The proof is done in two steps. First, by Lemma 3.8, we know that $\mathbf{x}$ has min entropy at least $\log(1-\varepsilon) + m\log(r) - m \geq (n+1)\log(q) - 2\log(\varepsilon') - 2$. Moreover, $\mathbf{e}^\intercal\mathbf{x} + y$ is in $\mathbb{Z}_q$. Applying the leftover hash lemma 3.5, we obtain

$$\Delta\left((\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{e}^\intercal\mathbf{x} + y), (\mathbf{A}, \mathbf{u}, \mathbf{e}^\intercal\mathbf{x} + y)\right) < \varepsilon'$$

where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{n-1}$. Now, using Lemma 3.6, we know that

$$\Delta\left(\mathbf{e}^\intercal\mathbf{x} + y, e'\right) < 2\varepsilon$$

Summing the two statistical distances concludes the proof. □

3.3 Rerandomizing LWE samples

We finally describe a simple application of Lemma 3.1. Generating fresh LWE samples for a fixed secret $\mathbf{s}$ from a bounded number of samples is very useful, for example to build a public key encryption scheme from a symmetric one. It has already been shown in the succession of papers [Reg05, GPV08, ACPS09] that multiplying a matrix of m LWE samples $(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)$ by a discrete Gaussian $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m,r}$ and adding another Gaussian term $y \leftarrow \mathcal{D}_{\mathbb{Z},r}$ to the error part yields a fresh LWE sample $(\mathbf{a}', \mathbf{s}^\intercal\mathbf{a}' + e')$ with a somewhat larger Gaussian noise e'. Here we have shown that picking $\mathbf{x}$ according to a discrete Gaussian distribution over a coset $\mathbf{c}$ of $\Lambda_q^\perp(\mathbf{G}^\intercal)$ is enough for this rerandomization process. Moreover, we show that the distribution of the final error is independent of the coset $\mathbf{c}$, which will come in handy for hiding homomorphic evaluations. We note that this could be extended to any other lattice with a small public basis (see the last paragraph of Section 5), but we mainly focus on $\Lambda_q^\perp(\mathbf{G}^\intercal)$ because this is sufficient for our use.

4 Basic GSW Cryptosystem

In this section, we present the Homomorphic Encryption scheme introduced by [GSW13], with notation inspired by [AP14]. We defer setting the parameters to Section 5.3. The scheme is composed of the following algorithms:

- $\mathsf{Setup}\left(1^\lambda\right)$: samples $\bar{\mathbf{s}} \xleftarrow{\$} \mathbb{Z}_q^{n-1}$ and returns the secret key $\mathbf{s} = \left(-\bar{\mathbf{s}}, 1\right) \in \mathbb{Z}_q^n$.
- $\mathsf{Encrypt}(\mathbf{s}, \mu)$: given the secret key $\mathbf{s} = \left(-\bar{\mathbf{s}}, 1\right)$ and a message $\mu \in \{0,1\}$, samples a matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n-1)\times m}$ and $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha}$. The algorithm then returns $\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \bar{\mathbf{s}}^\intercal\mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + \mu\,\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ as the ciphertext. Notice that $\mathbf{s}^\intercal\mathbf{C} = \mathbf{e}^\intercal + \mu\mathbf{s}^\intercal\mathbf{G}$, the last column of which is close to $\mu\,\frac{q}{2}$.
- $\mathsf{Decrypt}(\mathbf{s}, \mathbf{C})$: given a ciphertext $\mathbf{C}$ and the secret key $\mathbf{s}$, computes the inner product of $\mathbf{s}^\intercal$ and the last column of $\mathbf{C}$, and finally returns 0 if the norm of the result is smaller than $\frac{q}{4}$, otherwise it returns 1.

We omit the original Eval algorithm since our modified version, which guarantees circuit privacy, is presented in Section 5.1.
The IND-CPA security of this scheme comes directly from [GSW13] and the LWE assumption.

In order to shorten several formulas in the rest of the paper, we slightly abuse the notation and define a modified version of the encryption algorithm $\mathsf{Encrypt}_\gamma(\mathbf{s}, \mu)$, which is exactly the same as the previously defined $\mathsf{Encrypt}(\mathbf{s}, \mu)$, except that $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\gamma}$. We implicitly use $\mathsf{Encrypt}(\mathbf{s}, \mu)$ to denote $\mathsf{Encrypt}_\alpha(\mathbf{s}, \mu)$.

Extension to public key setting. This scheme can be easily adapted to the public key setting. We now describe $\mathsf{Setup}_{\mathsf{pub}}$ and $\mathsf{Encrypt}_{\mathsf{pub}}$, as the other algorithms are identical to the private key setting.

- $\mathsf{Setup}_{\mathsf{pub}}\left(1^\lambda\right)$: given the security parameter λ, samples $\bar{\mathbf{s}} \xleftarrow{\$} \mathbb{Z}_q^{n-1}$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n-1)\times m}$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha}$. The algorithm returns the secret key $\mathbf{s} = \left(-\bar{\mathbf{s}}, 1\right) \in \mathbb{Z}_q^n$ and the public key $\widehat{\mathbf{A}} = \begin{pmatrix} \mathbf{A} \\ \bar{\mathbf{s}}^\intercal\mathbf{A} + \mathbf{e}^\intercal \end{pmatrix}$.
- $\mathsf{Encrypt}_{\mathsf{pub}}\left(\widehat{\mathbf{A}}, \mu\right)$: given the public key $\widehat{\mathbf{A}}$ and a message $\mu \in \{0,1\}$, samples a matrix $\mathbf{R} \xleftarrow{\$} \{-1,0,1\}^{m\times m}$. The algorithm then sets $\mathbf{C} = \widehat{\mathbf{A}}\mathbf{R} + \mu\,\mathbf{G}$ and returns $\mathbf{C} \in \mathbb{Z}_q^{n\times m}$ as the ciphertext. Notice that $\mathbf{s}^\intercal\mathbf{C} = \mathbf{e}^\intercal\mathbf{R} + \mu\mathbf{s}^\intercal\mathbf{G}$ the last column of which is close to $\mu\,\frac{q}{2}$.

Basic homomorphic operations. The homomorphic operations are done as follows:

- Homomorphic addition: $\mathbf{C}_1 \boxplus \mathbf{C}_2 = \mathbf{C}_1 + \mathbf{C}_2$
- Homomorphic multiplication: $\mathbf{C}_1 \boxdot \mathbf{C}_2 \leftarrow \mathbf{C}_1 \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{C}_2)$

where the $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ algorithm is the randomized bit decomposition described in Definition 2.1.

From now on and for readability, we will assume a correct choice of parameters has been made. This setting is discussed in Section 5.3.

4.1 Rerandomizing and scaling GSW ciphertexts

Here we describe our new technique to rerandomize GSW ciphertexts. This method allows the scaling of GSW ciphertexts, which will be used in our circuit evaluation procedure.

We recall the form of a GSW ciphertext

$$\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \bar{\mathbf{s}}^\intercal \mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} + \mu \mathbf{G}$$

Using the rerandomization of LWE samples presented in Section 3, it is possible to generate a fresh encryption of 0 by computing $\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V})$, where $\mathbf{C}$ is an encryption of 0 and $\mathbf{V}$ is any matrix in $\mathbb{Z}_q^{n \times m}$.

Lemma 4.1. *Let $r > 0$. For any $\mathbf{V} \in \mathbb{Z}_q^{n \times m}$, if $r = \Omega\left(\alpha\sqrt{\lambda\, m \log m}\right)$, with α being the Gaussian parameter of fresh encryptions, then*

$$\left(\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}, \mathbf{C}\right) \approx_s (\mathbf{C}', \mathbf{C})$$

where $\mathbf{C} = \begin{pmatrix} \mathbf{A} \\ \bar{\mathbf{s}}^\intercal \mathbf{A} + \mathbf{e}^\intercal \end{pmatrix} \leftarrow \mathsf{Encrypt}(\mathbf{s}, 0)$, $\mathbf{C}' \leftarrow \mathsf{Encrypt}_\gamma(\mathbf{s}, 0)$, with $\gamma = r\sqrt{1 + \|\mathbf{e}\|^2}$.

Proof. Fix $\mathbf{v} \in \mathbb{Z}_q^m$ and $\mathbf{e}$ such that $\|\mathbf{e}\| \leq C\alpha\sqrt{m}$, where C is as in Lemma 2.4. Then by applying Lemma 3.1 with $r = \Omega\left(\alpha\sqrt{\lambda\, m \log m}\right)$ and $\varepsilon' = \varepsilon = 2^{-\lambda}$ we have

$$\Delta\left((\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{e}^\intercal \mathbf{x} + y), (\mathbf{A}, \mathbf{u}, e')\right) < 3 \cdot 2^{-\lambda}$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n-1) \times m}$, $\mathbf{x} \leftarrow \mathbf{G}_{\text{rand}}^{-1}(\mathbf{v})$ and $y \leftarrow \mathcal{D}_{\mathbb{Z},r}$. From this we obtain that for $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha}$:

$$\begin{aligned}
&\Delta\left((\mathbf{A}, \mathbf{e}, \mathbf{A}\mathbf{x}, \mathbf{e}^\intercal \mathbf{x} + y), (\mathbf{A}, \mathbf{e}, \mathbf{u}, e')\right) \\
&= \sum_{\mathbf{w} \in \mathbb{Z}^m} \Delta\left((\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{w}^\intercal \mathbf{x} + y), (\mathbf{A}, \mathbf{u}, w')\right) \cdot \Pr[\mathbf{e} = \mathbf{w}] \\
&\leq \sum_{\|\mathbf{w}\| < C\alpha\sqrt{m}} 3 \cdot 2^{-\lambda} \Pr[\mathbf{e} = \mathbf{w}] + \sum_{\|\mathbf{w}\| \geq C\alpha\sqrt{m}} \Pr[\mathbf{e} = \mathbf{w}] \\
&\leq 3 \cdot 2^{-\lambda} + \Pr\left[\|\mathbf{e}\| \geq C\alpha\sqrt{m}\right] \\
&\leq 3 \cdot 2^{-\lambda} + 2^{-\Omega(\lambda)}
\end{aligned}$$

In the left operand of the third equation we bound the statistical distance by $3 \cdot 2^{-\lambda}$ and in the right operand we bound it by 1. To obtain the last inequality we use Lemma 2.4 and have $\Pr[\|\mathbf{e}\| > C\alpha\sqrt{m}] \leq 2^{-\Omega(m)} \leq 2^{-\Omega(\lambda)}$ since $m \geq \lambda$. By rewriting this distance we have for any $\mathbf{v} \in \mathbb{Z}_q^m$

$$\left(\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{v}) + \begin{pmatrix} \mathbf{0} \\ y \end{pmatrix}, \mathbf{C}\right) \approx_s \left(\begin{pmatrix} \mathbf{u} \\ \bar{\mathbf{s}}^\intercal \mathbf{u} + e' \end{pmatrix}, \mathbf{C}\right)$$

By writing $\mathbf{V} = (\mathbf{v}_1 \mid \ldots \mid \mathbf{v}_m)$ and $\mathbf{y} = (y_1, \ldots, y_m)$, we have

$$\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix} = \left(\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{v}_1) + \begin{pmatrix} \mathbf{0} \\ y_1 \end{pmatrix} \mid \ldots \mid \mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{v}_m) + \begin{pmatrix} \mathbf{0} \\ y_m \end{pmatrix}\right)$$

We define the distributions $(D_i)_{0 \leq i \leq m}$ in which the first i columns of $\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}$ are replaced with "fresh" $\begin{pmatrix} \mathbf{u} \\ \bar{\mathbf{s}}^\intercal \mathbf{u} + e' \end{pmatrix}$ and we obtain through a hybrid argument that

$$\Delta\left(\left(\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}, \mathbf{C}\right), \left(\begin{pmatrix} \mathbf{A}' \\ \bar{\mathbf{s}}^\intercal \mathbf{A}' + \mathbf{e}'^\intercal \end{pmatrix}, \mathbf{C}\right)\right) \leq m(3 \cdot 2^{-\lambda} + 2^{-\Omega(\lambda)})$$

□

As a direct corollary we remark that the scaling of a GSW encryption $\mathbf{C}$ of μ by a bit a, defined as $\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(a \cdot \mathbf{G}) + \begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal \end{pmatrix}$, where $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z}^m,r}$, does not depend on a, but only on $a\mu$.

5 Our Scheme: Circuit-Private Homomorphic Evaluation for GSW

In this section, we prove that a slight modification of the GSW encryption scheme is enough to guarantee circuit privacy, i.e. that an evaluation of any branching program does not reveal anything more than the result of the computation and the length of the branching program, as long as the secret key holder is honest.
First, we state our definition of circuit privacy, similar to [IP07, Definition 7], which is stronger than the one given in [Gen09, Definition 2.1.6] in the sense that it is simulation based, but weaker in the sense that we leak information about the length of the branching program.

Definition 5.1 (Simulation-based circuit privacy). *We say that a homomorphic encryption scheme $\mathcal{E}$ is circuit private if there exists a PPT algorithm* Sim *such that for any branching program Π of length $L = \mathrm{poly}(\lambda)$ on ℓ variables, any $x_1, \ldots, x_\ell \in \{0,1\}^\ell$, the following holds:*

$$\begin{aligned} &\left(\mathcal{E}.\mathsf{Eval}\left(evk, \Pi, (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\right), \mathbf{C}_1, \ldots, \mathbf{C}_\ell, 1^\lambda, \mathbf{s}\right) \\ \approx_s\ &\left(\mathsf{Sim}\left(1^\lambda, \Pi\left(x_1, \ldots, x_\ell\right), 1^L, (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\right), \mathbf{C}_1, \ldots, \mathbf{C}_\ell, 1^\lambda, \mathbf{s}\right) \end{aligned}$$

where $\mathbf{s} \leftarrow \mathcal{E}.\mathsf{Setup}\left(1^\lambda\right)$, $\mathbf{C}_i \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathbf{s}, x_i)$ *for* $i \in [\ell]$.

We can now state our main theorem:

Theorem 5.2 (Main theorem). *There exists a fully homomorphic encryption scheme for branching programs that is circuit private and whose security is based on the* LWE *assumption with polynomial noise-to-modulus ratio.*

5.1 Homomorphic evaluation for branching programs

We first recall the branching program evaluation algorithm given in [BV14] and describe our modified version.

Permutation branching programs. A permutation branching program Π of length L and width W with input space $\{0,1\}^\ell$ is a sequence of L tuples of the form $\left(\mathrm{var}(t), \pi_{t,0}, \pi_{t,1}\right)$ where

- $\mathrm{var} : [L] \to [\ell]$ is a function that associates the t-th tuple with an input bit $x_{\mathrm{var}(t)}$
- $\pi_{t,0}, \pi_{t,1} : [W] \to [W]$ are permutations that dictate the t-th step of the computation.

On input $(x_1, \ldots, x_\ell)$, Π outputs 1 iff

$$\pi_{L,x_{\mathrm{var}(L)}}(\cdots(\pi_{1,x_{\mathrm{var}(1)}}(1))\cdots) = 1.$$

Following [BV14, IP07], we will evaluate Π recursively as follows. We associate each $t \in [L]$ with the characteristic vector $\mathbf{v}_t \in \{0,1\}^W$ of the current "state", starting with $\mathbf{v}_0 = (1, 0, \ldots, 0)$. We can then compute the w-th entry of $\mathbf{v}_t$ (denoted by $\mathbf{v}_t[w]$) as follows: for all $t \in [L], w \in [W]$,

$$\begin{aligned} \mathbf{v}_t[w] &= \mathbf{v}_{t-1}\left[\pi^{-1}_{t,x_{\mathrm{var}(t)}}(w)\right] \\ &= x_{\mathrm{var}(t)} \cdot \mathbf{v}_{t-1}\left[\pi^{-1}_{t,1}(w)\right] + \left(1 - x_{\mathrm{var}(t)}\right) \cdot \mathbf{v}_{t-1}\left[\pi^{-1}_{t,0}(w)\right] \end{aligned} \tag{5.1}$$

Our branching program evaluation. Here we present our $\mathsf{Eval}\left(\Pi, (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\right)$ algorithm (note that it does not require any evaluation key), which homomorphically evaluates a branching program Π over ciphertexts $\mathbf{C}_1, \ldots, \mathbf{C}_\ell$. The first state vector is encrypted without noise: the initial encrypted state vector is $\mathbf{V}_0 = (\mathbf{G}, \mathbf{0}, \ldots, \mathbf{0})$, i.e. $\mathbf{V}_0[1] = \mathbf{G}$ and $\mathbf{V}_0[i] = \mathbf{0}$, for $2 \le i \le W$. Note that $\mathbf{G}$ and $\mathbf{0}$ are noiseless encryptions of 1 and 0, respectively. The encrypted state vector is then computed at each step by homomorphically applying (5.1) and adding a noise term: for $t \in [L]$ and $w \in [W]$

$$\begin{aligned} \mathbf{V}_t[w] \leftarrow\ & \mathbf{C}_{\mathrm{var}(t)} \cdot \mathbf{G}^{-1}_{\mathrm{rand}}\left(\mathbf{V}_{t-1}\left[\pi^{-1}_{t,1}(w)\right]\right) \\ & + \left(\mathbf{G} - \mathbf{C}_{\mathrm{var}(t)}\right) \cdot \mathbf{G}^{-1}_{\mathrm{rand}}\left(\mathbf{V}_{t-1}\left[\pi^{-1}_{t,0}(w)\right]\right) + \boxed{\begin{pmatrix} \mathbf{0} \\ \mathbf{y}^\intercal_{t,w} \end{pmatrix}} \end{aligned} \tag{5.2}$$

where $\mathbf{y}_{t,w} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$. The output of the evaluation algorithm is $\mathbf{V}_L[0] \in \mathbb{Z}_q^{n \times m}$.

Remark 2 (Comparison with [BV14, AP14]). The differences between our homomorphic evaluation procedure and the previous ones are as follows:

- We added an additional Gaussian noise to the computation, as captured in the boxed term;
- [BV14] uses the deterministic $\mathbf{G}^{-1}_{\text{det}}(\cdot)$ whereas [AP14] introduced the randomized $\mathbf{G}^{-1}_{\text{rand}}(\cdot)$ for efficiency. Here, we crucially exploit the randomized $\mathbf{G}^{-1}_{\text{rand}}(\cdot)$ for privacy.

Simulator. Towards proving circuit privacy, we need to specify a simulator Sim. We first describe a simulator that is given access to the number of times each variable is used and prove that its output distribution is statistically close to the result of Eval (Lemma 5.5). We can then pad the branching program so that each variable is used the same number of times. Given the security parameter λ, the length L of the branching program Π, the number of times τ_i that Π uses the i-th variable, the final value x_f of the evaluation of Π on input $(x_1, \ldots, x_\ell)$, the ciphertexts $\mathbf{C}_i$ encrypting x_i for $i \in [\ell]$, Sim mimics the way error grows in the states of Eval by doing τ_i dummy steps of computation with the i-th variable. This gives a new encryption $\widehat{\mathbf{A}}_f$ of 0 with the same noise distribution as the ciphertext output by the Eval procedure. Sim then adds the message part x_f to this ciphertext and outputs $\mathbf{C}_f = \widehat{\mathbf{A}}_f + x_f\mathbf{G}$.
In other words,

$$\mathsf{Sim}\left(1^\lambda, x_f, (1^{\tau_1}, \ldots, 1^{\tau_\ell}), (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\right) \leftarrow \sum_{i=1}^{\ell}\sum_{t=1}^{\tau_i}\left(\mathbf{C}_i \cdot \left(\mathbf{G}^{-1}_{\text{rand}}(\mathbf{0}) - \mathbf{G}^{-1}_{\text{rand}}(\mathbf{0})\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_t^\intercal\end{pmatrix}\right) + x_f\mathbf{G}$$

where $\mathbf{y}_t \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$ for $t \in [L]$.
We note that the sum of $2\tau_i$ samples $\mathbf{G}^{-1}_{\text{rand}}(\mathbf{0})$ can be sampled at once using the $\mathbf{G}^{-1}_{\text{rand}}(\cdot)$ algorithm with a larger parameter $r\sqrt{2\tau_i}$, and the sum of τ_i samples from $\mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$ is close to a sample from $\mathcal{D}_{\mathbb{Z}^m, r\sqrt{2\tau_i}}$.

5.2 Proof of circuit privacy

We proceed to establish circuit privacy in two steps. We first analyze how the ciphertext distribution changes in a single transition, and then proceed by induction to reason about homomorphic evaluation of the entire branching program.

Step 1. We begin with the following simple identity, which is useful for analyzing the output of (5.2):

Lemma 5.3. *For any $\mathbf{V}_0, \mathbf{V}_1 \in \mathbb{Z}_q^{n\times m}$ and any $\mathbf{C} = \widehat{\mathbf{A}} + x\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ where $x \in \{0,1\}$, we have:*

$$\mathbf{C}\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) + (\mathbf{G}-\mathbf{C})\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0) = \widehat{\mathbf{A}}\cdot\left(\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) - \mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)\right) + \mathbf{V}_x$$

Roughly speaking, this says that if $\mathbf{V}_0, \mathbf{V}_1, \mathbf{C}$ are respectively GSW encryptions of v_0, v_1, x, then

$$\mathbf{V}' \leftarrow \mathbf{C}\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) + (\mathbf{G}-\mathbf{C})\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)$$

is a GSW encryption of v_x. Note that, by using the GSW ciphertext randomization from Lemma 4.1, we have:

$$\left(\mathbf{C}\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) + (\mathbf{G}-\mathbf{C})\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}^\intercal\end{pmatrix}, \mathbf{C}\right) \approx_s (\mathbf{C}', \mathbf{C})$$

where $\mathbf{C} = \begin{pmatrix}\mathbf{A}\\ \bar{\mathbf{s}}^\intercal\mathbf{A} + \mathbf{e}^\intercal\end{pmatrix} + x\mathbf{G} \leftarrow \mathsf{Encrypt}(\mathbf{s}, x)$, $\mathbf{V}_i \leftarrow \mathsf{Encrypt}_\gamma(\mathbf{s}, v_i)$ for $i \in \{0,1\}$, $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$, and $\mathbf{C}' \leftarrow \mathsf{Encrypt}_\zeta(\mathbf{s}, v_x)$ where $\zeta = \sqrt{\gamma^2 + 2r^2(1+\|\mathbf{e}\|^2)}$, i.e. $\mathbf{V}'$ looks like a fresh encryption of v_x with a slightly larger noise.

Proof. Proving the first identity just requires performing the calculations:

$$\begin{aligned}
&\mathbf{C}\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) + (\mathbf{G}-\mathbf{C})\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)\\
=&\left(\widehat{\mathbf{A}} + x\mathbf{G}\right)\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) + \left((1-x)\mathbf{G} - \widehat{\mathbf{A}}\right)\cdot\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)\\
=&\widehat{\mathbf{A}}\cdot\left(\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) - \mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)\right) + x\mathbf{V}_1 + (1-x)\mathbf{V}_0\\
=&\widehat{\mathbf{A}}\cdot\left(\mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_1) - \mathbf{G}^{-1}_{\text{rand}}(\mathbf{V}_0)\right) + \mathbf{V}_x
\end{aligned}$$

To show the second part of the lemma we first observe that by applying Lemma 2.3 we have

$$\begin{pmatrix}\mathbf{0}\\ \mathbf{y}^\intercal\end{pmatrix} \approx_s \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_1^\intercal\end{pmatrix} - \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_0^\intercal\end{pmatrix}$$

where $\mathbf{y}_b \leftarrow \mathcal{D}_{\mathbb{Z}^m,r}, b \in \{0,1\}$, using Lemma 4.1 we also have:

$$\left(\widehat{\mathbf{A}} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_b) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_b^\intercal\end{pmatrix}, \mathbf{C}\right) \approx_s (\mathbf{C}_b, \mathbf{C})$$

where $\mathbf{C}_b \leftarrow \mathsf{Encrypt}_\zeta(\mathbf{s}, 0)$ with $\zeta = r\sqrt{1+\|\mathbf{e}\|^2}$, for $b \in \{0,1\}$. We now have

$$\left(\mathbf{C} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_1) + (\mathbf{G} - \mathbf{C}) \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_0), \mathbf{C}\right) \approx_s (\mathbf{C}_1 - \mathbf{C}_0 + \mathbf{V}_x, \mathbf{C})$$

By additivity of variance on independent variables, we obtain that $\mathbf{C}' = \mathbf{C}_1 - \mathbf{C}_0 + \mathbf{V}_x$ looks like a fresh encryption of $0 - 0 + v_x = v_x$ with parameter $\sqrt{\gamma^2 + 2r^2(1+\|\mathbf{e}\|^2)}$. □

Step 2. We now prove that, at each step of the evaluation, each entry of the encrypted state $\mathbf{V}_t$ looks like a fresh GSW encryption of the corresponding entry of the state $\mathbf{v}_t$, even given the GSW encryptions of the input bits, except for a small correlation in the noise.

Lemma 5.4 (Distribution of the result of Eval). *For any branching program Π of length L on ℓ variables, we define $\tau_{t,i}$ to be the number of times the i-th variable has been used after t steps of the evaluation, i.e. $\tau_{t,i} = \left|\text{var}^{-1}(i) \cap [t]\right|$, for i in $[\ell]$ and $t \in [L]$.*
For any $x_1, \ldots, x_\ell \in \{0,1\}^\ell$, any $\bar{\mathbf{s}} \in \mathbb{Z}_q^{n-1}$, at each step $t \in [L]$, for all indexes $w \in [W]$, the following holds:

$$\left(\mathbf{V}_t[w], (\mathbf{C}_i)_{i\in[\ell]}\right) \approx_s \left(\mathbf{C}'_{t,w}, (\mathbf{C}_i)_{i\in[\ell]}\right)$$

where $\mathbf{C}_i = \begin{pmatrix}\mathbf{A}_i\\ \bar{\mathbf{s}}^\intercal\mathbf{A}_i + \mathbf{e}_i^\intercal\end{pmatrix} + x_i\mathbf{G} \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$ *for* $i \in [\ell]$, $\mathbf{C}'_{t,w} \leftarrow \mathsf{Encrypt}_{r_t}(\mathbf{s}, \mathbf{v}_t[w])$ *for* $(t,w) \in [L] \times [W]$ *and* $r_t = r\sqrt{2\sum_{i=1}^{\ell}\tau_{t,i}\left(1+\|\mathbf{e}_i\|^2\right)}$. *In particular, $\mathbf{C}'_{t,w}$ is independent of w, and of the matrices $\mathbf{A}_i$, for $i \in [\ell]$.*

Proof. We prove this lemma by induction on $t \in [L]$. At step $t > 1$, for index $w \in [W]$ we use a series of hybrid distributions $\mathcal{H}_{t,w,k}$ for $0 \le k \le 2$ to prove that $\left(\mathbf{V}_t[w], (\mathbf{C}_i)_{i\in[\ell]}\right) \approx_s \left(\mathbf{C}'_{t,w}, (\mathbf{C}_i)_{i\in[\ell]}\right)$. In particular $\mathcal{H}_{t,w,0} = \left(\mathbf{V}_t[w], (\mathbf{C}_i)_{i\in[\ell]}\right)$, and $\mathcal{H}_{t,w,2} = \left(\mathbf{C}'_{t,w}, (\mathbf{C}_i)_{i\in[\ell]}\right)$.

Hybrid $\mathcal{H}_{t,w,0}$. Let $w_b = \pi_{t,b}^{-1}(w)$ for $b \in \{0,1\}$. We write w_β to denote $w_{x_{\text{var}(t)}}$, i.e. w_0 or w_1, depending on the value of the variable which is used at time t.

$$\begin{aligned}\mathcal{H}_{t,w,0} &= \left(\mathbf{V}_t[w], (\mathbf{C}_i)_{i\in[\ell]}\right)\\ &= \left(\mathbf{C}_{\text{var}(t)} \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_{t-1}[w_1]) + \left(\mathbf{G} - \mathbf{C}_{\text{var}(t)}\right) \cdot \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_{t-1}[w_0]) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,w}^\intercal\end{pmatrix}, (\mathbf{C}_i)_{i\in[\ell]}\right)\end{aligned}$$

where $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$ and $\mathbf{y}_{t,w} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$.

Hybrid $\mathcal{H}_{t,w,1}$. We set

$$\mathcal{H}_{t,w,1} = \left(\mathbf{C}_{\text{var}(t)} \cdot \mathbf{G}_{\text{rand}}^{-1}\left(\mathbf{C}'_{t-1,w_1}\right) + \left(\mathbf{G} - \mathbf{C}_{\text{var}(t)}\right) \cdot \mathbf{G}_{\text{rand}}^{-1}\left(\mathbf{C}'_{t-1,w_0}\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,w}^\intercal\end{pmatrix}, (\mathbf{C}_i)_{i\in[\ell]}\right)$$

where $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$, $\mathbf{y}_{t,w} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$ and $\mathbf{C}'_{t-1,w_b} \leftarrow \mathsf{Encrypt}_{r_{t-1}}(\mathbf{s}, \mathbf{v}_{t-1}[w_b])$ for $b \in \{0,1\}$.
By induction hypothesis we have $\mathcal{H}_{t-1,w_b,0} \approx_s \mathcal{H}_{t-1,w_b,2}$ for $b \in \{0,1\}$, i.e.

$$\left(\mathbf{V}_{t-1}[w_b], (\mathbf{C}_i)_{i\in[\ell]}\right) \approx_s \left(\mathbf{C}'_{t-1,w_b}, (\mathbf{C}_i)_{i\in[\ell]}\right)$$

where $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$ and $\mathbf{C}'_{t-1,w_b} \leftarrow \mathsf{Encrypt}_{r_{t-1}}(\mathbf{s}, \mathbf{v}_{t-1}[w_b])$ for $b \in \{0,1\}$.
We use the fact that applying a function to two distributions does not increase their statistical distance to obtain $\mathcal{H}_{t,w,0} \approx_s \mathcal{H}_{t,w,1}$.

Hybrid $\mathcal{H}_{t,w,2}$. Let

$$\mathcal{H}_{t,w,2} = \left(\mathbf{C}', (\mathbf{C}_i)_{i\in[\ell]}\right)$$

with $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$, $\mathbf{C}' \leftarrow \mathsf{Encrypt}_\zeta(\mathbf{s}, \mathbf{v}_{t-1}[w_\beta])$ and $\zeta = \sqrt{r_{t-1}^2 + 2r^2\left(1 + \left\|\mathbf{e}_{\mathrm{var}(t)}\right\|^2\right)}$.
By Lemma 5.3 we have:

$$\left(\mathbf{C}_{\mathrm{var}(t)} \cdot \mathbf{G}_{\mathrm{rand}}^{-1}\left(\mathbf{C}'_{t-1,w_1}\right) + \left(\mathbf{G} - \mathbf{C}_{\mathrm{var}(t)}\right) \cdot \mathbf{G}_{\mathrm{rand}}^{-1}\left(\mathbf{C}'_{t-1,w_0}\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,w}^\intercal\end{pmatrix}, (\mathbf{C}_i)_{i\in[\ell]}\right) \approx_s \left(\mathbf{C}', (\mathbf{C}_i)_{i\in[\ell]}\right)$$

where $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$, $\mathbf{y}_{t,w} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{2}}$, $\mathbf{C}'_{t-1,w_b} \leftarrow \mathsf{Encrypt}_{r_{t-1}}(\mathbf{s}, \mathbf{v}_{t-1}[w_b])$ for $b \in \{0,1\}$ and $\mathbf{C}' \leftarrow \mathsf{Encrypt}_\zeta(\mathbf{s}, \mathbf{v}_{t-1}[w_\beta])$. Note that $\mathbf{v}_{t-1}[w_\beta] = \mathbf{v}_t[w]$ and $r_t = \sqrt{r_{t-1}^2 + 2r^2\left(1 + \left\|\mathbf{e}_{\mathrm{var}(t)}\right\|^2\right)} = \zeta$ from which we have that $\mathbf{C}'$ and $\mathbf{C}'_{t,w}$ are identically distributed, and directly $\mathcal{H}_{t,w,1} \approx_s \mathcal{H}_{t,w,2}$.

We note that this recursive formula does not apply to step $t = 0$, we thus use $t = 1, w \in [W]$ as the base case. We only describe the steps that differ from the case $t > 1$.

Hybrid $\mathcal{H}_{1,w,1}$. We have $\mathbf{G}_{\mathrm{rand}}^{-1}(\mathbf{V}_0[w_b]) = \mathbf{G}_{\mathrm{rand}}^{-1}(\mathbf{v}_0[w_b] \cdot \mathbf{G})$ for $b \in \{0,1\}$. Notice that we now have exactly $\mathcal{H}_{1,w,1} = \mathcal{H}_{1,w,0}$.

Hybrids $\mathcal{H}_{1,w,2}$. The proof for $\mathcal{H}_{1,w,1} \approx_s \mathcal{H}_{1,w,2}$ is identical to the one of Lemma 5.3 except for the fact that the ciphertext $\mathbf{V}_x$ from the proof is now a plaintext $\mathbf{v}_0[w_\beta]$. The resulting ciphertext $\mathbf{C}'_{1,w}$ is now only the sum of two encryptions of 0 and $\mathbf{v}_0[w_\beta]$ and has a Gaussian parameter $r\sqrt{2\left(1 + \left\|\mathbf{e}_{\mathrm{var}(1)}\right\|^2\right)} = r_1$. This implies $\mathcal{H}_{1,w,1} \approx_s \mathcal{H}_{1,w,2}$.

□

We now proceed to prove circuit privacy. We will first prove the following lemma, which states that the Eval algorithm presented in Section 5.1 only leaks the final result of the evaluation and the number of times each variable is used.

Lemma 5.5. *Let $\mathcal{E}$ be the scheme defined in Section 4 with evaluation defined as in this section, and Sim be the corresponding simulator. Then for any branching program Π of length $L = \mathrm{poly}(\lambda)$ on ℓ variables, such that the i-th variable is used τ_i times, and any $x_1, \ldots, x_\ell \in \{0,1\}^\ell$, the following holds:*

$$\begin{aligned}&\left(\mathcal{E}.\mathsf{Eval}\left(\Pi, (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\right), \mathbf{C}_1, \ldots, \mathbf{C}_\ell, 1^\lambda, \mathbf{s}\right)\\ \approx_s\ &\left(\mathsf{Sim}\Big(1^\lambda, \Pi\left(x_1, \ldots, x_\ell\right), (1^{\tau_1}, \ldots, 1^{\tau_\ell}), (\mathbf{C}_1, \ldots, \mathbf{C}_\ell)\Big), \mathbf{C}_1, \ldots, \mathbf{C}_\ell, 1^\lambda, \mathbf{s}\right)\end{aligned}$$

where $\mathbf{s} \leftarrow \mathcal{E}.\mathsf{Setup}\left(1^\lambda\right)$, $\mathbf{C}_i \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathbf{s}, x_i)$ for i in $[\ell]$.

Proof. As shown in Lemma 5.4, the final result of the homomorphic evaluation of the branching program Π is of the form

$$\mathbf{V}_L[0] \approx_s \begin{pmatrix}\mathbf{A}\\ \bar{\mathbf{s}}^\intercal\mathbf{A} + \mathbf{f}^\intercal\end{pmatrix} + x_f\mathbf{G}$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n-1)\times m}$, $\mathbf{f} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r_L}$ and $r_L = r\sqrt{2\sum_{i=1}^{\ell}\left(1 + \|\mathbf{e}_i\|^2\right)\tau_i}$.
Now we prove that the output of Sim is statistically close to the same distribution. This proof follows from the fact that scaling GSW ciphertexts yields a result which is independent of the argument of $\mathbf{G}_{\mathrm{rand}}^{-1}(\cdot)$. Let $\mathbf{A}_{i,t}, \mathbf{A}'_{i,t} \xleftarrow{\$} \mathbb{Z}_q^{(n-1)\times m}$, $\mathbf{f}_{i,f}, \mathbf{f}'_{i,t} \leftarrow \mathcal{D}_{\mathbb{Z}^m, r\sqrt{1+\|\mathbf{e}_i\|}}$, then the joint distribution of the output of Sim and

ciphertexts $(\mathbf{C}_i)_{i\in[\ell]}$ is

$$\begin{aligned}\left(\mathcal{S}, (\mathbf{C}_i)_{i\in[\ell]}\right) = & \left(\sum_{i=1}^{\ell} \mathbf{C}_i \sum_{t=1}^{\tau_i} \left(\mathbf{G}_{\text{rand}}^{-1}(\mathbf{0}) - \mathbf{G}_{\text{rand}}^{-1}(\mathbf{0})\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_t^\intercal\end{pmatrix} + x_f\mathbf{G}, (\mathbf{C}_i)_{i\in[\ell]}\right)\\ \approx_s & \left(\sum_{i=1}^{\ell}\sum_{t=1}^{\tau_i}\begin{pmatrix}\mathbf{A}_{i,t}\\ \bar{\mathbf{s}}^\intercal\mathbf{A}_{i,t}+\mathbf{f}_{i,t}\end{pmatrix} + \begin{pmatrix}\mathbf{A}'_{i,t}\\ \bar{\mathbf{s}}^\intercal\mathbf{A}'_{i,t}+\mathbf{f}'_{i,t}\end{pmatrix}, (\mathbf{C}_i)_{i\in[\ell]}\right)\end{aligned}$$

by Lemma 3.1

$$\approx_s \left(\begin{pmatrix}\mathbf{A}\\ \bar{\mathbf{s}}^\intercal\mathbf{A}+\mathbf{f}^\intercal\end{pmatrix}, (\mathbf{C}_i)_{i\in[\ell]}\right)$$

by Lemma 2.3 and summing uniform variables.

The result is the same as the joint distribution of the output of Eval and ciphertexts $(\mathbf{C}_i)_{i\in[\ell]}$, thus concluding the proof. □

We are now ready to prove Theorem 5.2.

Proof (Main theorem). Theorem 5.2 follows from Lemma 5.5 by tweaking the Eval algorithm of $\mathcal{E}$: it is sufficient that this algorithm pads the branching program Π so that each variable is used L times. This padding is done by using the identity permutation for all steps after the L-th. After this proof, we discuss more efficient ways to pad branching program evaluations. It is easy to see that this step is enough to reach the desired circuit privacy property: the only information leaked besides the final result is $\tau_i = L$. □

Padding branching program evaluations. In order to pad a branching program Π that uses the i-th variable τ_i times to one that uses the i-th variable L times, we add $L - \tau_i$ steps, using the identity permutation at each one of these. Given $\mathbf{V}_L[0]$ the final result of the computation, this padding corresponds to steps $t \in [L+1, 2L-\tau_i]$ defined as follows:

$$\mathbf{V}_t[0] \leftarrow \mathbf{V}_{t-1}[0] + \mathbf{C}_i\left(\mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_{t-1}[0]) - \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_{t-1}[0])\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,0}^\intercal\end{pmatrix}$$

Using the same proof as Lemma 5.5 the final output will be

$$\begin{aligned}\mathbf{V}_{2L-\tau_i}[0] &\leftarrow \mathbf{V}_L[0] + \mathbf{C}_i\sum_{t=L}^{2L-\tau_i-1}\left(\mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_t[0]) - \mathbf{G}_{\text{rand}}^{-1}(\mathbf{V}_t[0])\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,0}^\intercal\end{pmatrix}\\ &\approx_s \mathbf{V}_L[0] + \mathbf{C}_i\sum_{t=L}^{2L-\tau_i-1}\left(\mathbf{G}_{\text{rand}}^{-1}(\mathbf{0}) - \mathbf{G}_{\text{rand}}^{-1}(\mathbf{0})\right) + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,0}^\intercal\end{pmatrix}\end{aligned}$$

Observe that by using Lemma 2.3 we have that

$$\begin{aligned}\sum_{t=L}^{2L-\tau_i-1}\left(\mathbf{G}_{\text{rand}}^{-1}(\mathbf{0}) - \mathbf{G}_{\text{rand}}^{-1}(\mathbf{0})\right) &\approx_s \mathcal{D}_{\Lambda_q^\perp(\mathbf{G}^\intercal),r_f}\\ \sum_{t=L}^{2L-\tau_i-1}\begin{pmatrix}\mathbf{0}\\ \mathbf{y}_{t,0}^\intercal\end{pmatrix} &\approx_s \mathcal{D}_{\mathbb{Z}^m,r_f}\end{aligned}$$

Where $r_f = r\sqrt{2(L-\tau_i)}$. We can thus do all the steps at once by outputting $\mathbf{V}_L[0] + \mathbf{C}_i\cdot\mathbf{X} + \begin{pmatrix}\mathbf{0}\\ \mathbf{y}_f^\intercal\end{pmatrix}$, where $\mathbf{X} \leftarrow \mathcal{D}^m_{\Lambda_q^\perp(\mathbf{G}^\intercal),r_f}$ and $\mathbf{y}_f \leftarrow \mathcal{D}_{\mathbb{Z}^m,r_f}$. We note that $\mathbf{X}$ can be sampled using the $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ algorithm with parameter r_f instead of r.

5.3 Setting the parameters

In this section we show that, for appropriate values of the parameters, the output of the homomorphic evaluation $\mathbf{V}_L[0]$ decrypts to $\Pi(x_1,\ldots,x_\ell)$ with overwhelming probability and guarantees circuit privacy.

We first recall the bounds on the parameters needed for both correctness and privacy. Let $n = \Theta(\lambda)$, $q = \text{poly}(n)$, $m = n \log q$, α be the Gaussian parameter of fresh encryptions, r be the parameter of $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$. Let $B = \Theta(\alpha\sqrt{m})$ be a bound on the norm of the error in fresh encryptions (using a tail cutting argument we can show that $B = C\alpha\sqrt{m}$ is sufficient to have a bound with overwhelming probability), $L_{\max} = \text{poly}(n)$ be a bound on the size of the branching programs we consider and $\ell_{\max} = \text{poly}(n)$ an upper bound on their number of variables. Let $\varepsilon = O(2^{-\lambda})$ and $\varepsilon' = O(2^{-\lambda})$.
We have the following constraints:

- $\alpha = \Omega(\sqrt{m})$ for the hardness of $\mathsf{LWE}_{n-1,q,\mathcal{D}_{\mathbb{Z},\alpha}}$
- $r \geq \sqrt{\frac{5\ln(2m(1+1/\varepsilon))}{\pi}}$ for the correctness of $\mathbf{G}_{\text{rand}}^{-1}(\cdot)$ sampling
- $r \geq 4\left((1-\varepsilon)(2\varepsilon')^2\right)^{-\frac{1}{m}}$ for the leftover hash lemma
- $r \geq \sqrt{5}(1+B)\sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}}$ for Lemma 3.7
- $q = \Omega\left(\sqrt{m}r\alpha\left(m\, L_{\max}\, \ell_{\max}\right)^{1/2}\right)$ for the correctness of decryption

We can thus set the parameters as follows:

- $n = \Theta(\lambda)$,
- $L_{\max} = \text{poly}(n)$,
- $\ell_{\max} = \text{poly}(n)$,
- $\alpha = \Theta(\sqrt{n})$,
- $r = \tilde{\Theta}(n)$,
- $q = \tilde{\Theta}\left(n^{5/2} \cdot L_{\max} \cdot \ell_{\max}\right)$, a power of 2.

Note that the ciphertext size grows with $\log L_{\max}$. Correctness follows directly.

Lemma 5.6 (Correctness). *For any branching program Π of length L on ℓ variables, any $x_1, \ldots, x_\ell \in \{0,1\}^\ell$, the result of the homomorphic evaluation $\mathbf{C}_f = \mathsf{Eval}(\Pi, (\mathbf{C}_1, \ldots, \mathbf{C}_\ell))$ decrypts to $\Pi(x_1, \ldots, x_\ell)$ with overwhelming probability, where $\mathbf{C}_i \leftarrow \mathsf{Encrypt}(\mathbf{s}, x_i)$ for $i \in [\ell]$ and $\mathbf{s} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$.*

Proof. Lemma 5.4 shows that the noise distribution of the output $\mathbf{C}_f$ of Eval has parameter $r_f = r\sqrt{2\sum_{i=1}^{\ell} \tau_i \left(1 + \|\mathbf{e}_i\|^2\right)}$, that is $r\sqrt{2L\sum_{i=1}^{\ell}\left(1 + \|\mathbf{e}_i\|^2\right)}$ because of the padding we applied to Π. We have $r_f \leq r\sqrt{2L\ell(1 + C^2\alpha^2 m)}$ with C the universal constant defined in Lemma 2.4, Using the bounds $L_{\max}$ and $\ell_{\max}$ we have $r_L = \tilde{O}\left(r\alpha(m\, L_{\max}\, \ell_{\max})^{1/2}\right)$. Finally, by a tail cutting argument, $q = \tilde{\Theta}(r_L\sqrt{n}) = \tilde{\Theta}\left(n^{5/2} L_{\max} \ell_{\max}\right)$ is enough for decryption to be correct with overwhelming probability. □

5.4 Arbitrary modulus and random trapdoor matrix

In this paragraph we show how to instantiate our proofs in a more generic setting.

Our GSW ciphertext rerandomization can be straightforwardly adapted to any matrix $\mathbf{H}$ and modulus q, as long as the lattice $\Lambda_q^\perp(\mathbf{H}^\intercal)$ has a small public basis, i.e. a small public trapdoor. Observe that the conditions needed to apply GSW ciphertext rerandomization are given in Lemma 3.7, which bounds the smoothing parameter of the lattice

$$\mathcal{L} = \left\{\mathbf{v} \in \Lambda_q^\perp(\mathbf{H}^\intercal) \times \mathbb{Z} : \widehat{\mathbf{e}}^\intercal \mathbf{v} = 0\right\}$$

and in Lemma 3.8 which gives the min-entropy of a Gaussian over $\Lambda_q^\perp(\mathbf{H}^\intercal)$.
Let $\beta \geq \|\mathbf{t}_i\|$, where $\mathbf{T} = \{\mathbf{t}_1, ..., \mathbf{t}_m\}$ is the public trapdoor of $\mathbf{H}$ (i.e. $\mathbf{T}$ is a small basis of $\Lambda_q^\perp(\mathbf{H}^\intercal)$), we show that the previous two lemmas can be proven for $\mathbf{H}$ and the parameter r only grows by a factor β. First, observe that Lemma 3.7 aims to find m small independent vectors in $\mathcal{L}$. By noticing that

$$\mathcal{L} = \left\{(\mathbf{v}, -\mathbf{v}^\intercal \mathbf{e}) : \mathbf{v} \in \Lambda_q^\perp(\mathbf{H}^\intercal)\right\}$$

we can exhibit m small vectors $\mathbf{u}_i = (\mathbf{t}_i, -\mathbf{t}_i^\intercal \mathbf{e}), i \in [m]$ which are of norm

$$\|\mathbf{u}_i\| \leq \|\mathbf{t}_i\| (1 + \|\mathbf{e}\|) \leq \beta(1 + \|\mathbf{e}\|)$$

This bound is the one we obtain in Lemma 3.7 for $\Lambda_q^\perp(\mathbf{G}^\intercal)$ where $\|\mathbf{T}\| = \sqrt{5}$.
Second, we show that the bound on the min-entropy of Lemma 3.8 can be expressed as a function of β, simply by using the fact that $\det(\mathbf{T}) \leq \|\mathbf{T}\|^m = \beta^m$. From this we have the following bound on the min-entropy:

$$H_\infty \left(\mathcal{D}_{\Lambda_q^\perp(\mathbf{H}^\intercal)+\mathbf{c},r} \right) \geq \log(1-\varepsilon) + m \log(r) - m \log(\beta)$$

This bound is slightly worse that the one we obtain in Lemma 3.8 for $\mathbf{G}$ (where we had 2 instead of β). However this is not a problem as it is a weaker bound than the one obtained in Lemma 3.7.
By using these two lemmas we can rerandomize GSW ciphertexts and ensure circuit privacy for arbitrary modulus q, and any matrix $\mathbf{H}$ with public trapdoor by setting the Gaussian parameter of $\mathbf{H}^{-1}(\cdot)$ to $r = \tilde{\Theta}(\beta n)$.

5.5 Extension to general circuits

We can realize circuit-private FHE for general circuits via bootstrapping as in [OPP14, GHV10] by combining a FHE for general circuits with decryption in NC^1 with our circuit-private FHE for NC^1 circuits. The ensuing scheme however will not satisfy the multi-hop requirement.

Acknowledgements

We thank Vinod Vaikuntanathan for insightful discussions.

References

ACPS09. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, August 2009.

AGHS13. Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Discrete gaussian leftover hash lemma over infinite domains. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 97–116. Springer, Heidelberg, December 2013.

AP14. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 297–314. Springer, Heidelberg, August 2014.

AR13. Divesh Aggarwal and Oded Regev. A note on discrete gaussian combinations of lattice vectors. *CoRR*, abs/1308.2405, 2013.

BGG+14. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Heidelberg, May 2014.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012.

BLP+13. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013.

BV11a. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 97–106. IEEE Computer Society Press, October 2011.

BV11b. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, Heidelberg, August 2011.

BV14. Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In Moni Naor, editor, *ITCS 2014*, pages 1–12. ACM, January 2014.

DM15. Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 617–640. Springer, Heidelberg, April 2015.

DRS04. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer, Heidelberg, May 2004.

DS16. Léo Ducas and Damien Stehlé. Sanitization of FHE ciphertexts. In *EUROCRYPT*, 2016. To appear. Also, eprint 2016/164.

Gen09. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

GHS12. Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867. Springer, Heidelberg, August 2012.

GHV10. Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-Hop homomorphic encryption and rerandomizable Yao circuits. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 155–172. Springer, Heidelberg, August 2010.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

GVW15. Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 469–477. ACM Press, June 2015.

HS15. Shai Halevi and Victor Shoup. Bootstrapping for HElib. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 641–670. Springer, Heidelberg, April 2015.

IP07. Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 575–594. Springer, Heidelberg, February 2007.

LMSV12. Jake Loftus, Alexander May, Nigel P. Smart, and Frederik Vercauteren. On CCA-secure somewhat homomorphic encryption. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 55–72. Springer, Heidelberg, August 2012.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

MR04. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004.

MR07. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, April 2007.

OPP14. Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 536–553. Springer, Heidelberg, August 2014.

Pei10. Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 80–97. Springer, Heidelberg, August 2010.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

SS10. Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 377–394. Springer, Heidelberg, December 2010.

SV10. Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 420–443. Springer, Heidelberg, May 2010.

SYY99. Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for NC1. In *40th FOCS*, pages 554–567. IEEE Computer Society Press, October 1999.

vDGHV10. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 24–43. Springer, Heidelberg, May 2010.