Privately Constrained PRFs from DCR: Puncturing and Bounded Waring Rank

Amik Raj Behera[1], Pierre Meyer[2], Claudio Orlandi[2], Lawrence Roy[2], and Peter Scholl[2]

[1] University of Copenhagen, Copenhagen, Denmark ambe@di.ku.dk
[2] Aarhus University, Aarhus, Denmark
{pierre.meyer,orlandi,peter.scholl}@cs.au.dk, ldr709@gmail.com

Abstract. A *privately constrained pseudorandom function (pCPRF)* is a PRF with the additional property that one can derive a constrained key that allows evaluating the PRF only on inputs satisfying a constraint predicate C, without revealing C itself or leaking information about the PRF's output on inputs that do not satisfy the constraint.

Existing privately constrained PRFs face significant limitations: either (1) they rely on assumptions known to imply fully-homomorphic encryption or indistinguishability obfuscation, (2) they support only highly restricted classes of constraints—for instance, no known group-based pCPRF even supports the simple class of *puncturing* constraints (where the constrained key permits evaluation on all but one point while hiding the punctured point), or (3) they are limited to polynomial-size input domains. A long-standing open question has been whether one can construct a privately constrained PRF from group-based assumptions for more expressive classes of constraints. In this work, we present a pCPRF based on the decisional composite residuosity (DCR) assumption that supports a highly expressive class of predicates, namely constraints with polynomially bounded Waring rank, which notably includes puncturing.

From a technical perspective, our work follows the general template of Couteau, Meyer, Passelègue, and Riahinia (Eurocrypt'23), who constructed a pCPRF from group-based homomorphic secret-sharing but were limited to inner-product constraints in the constraint-hiding setting. Leveraging novel techniques for computing with *distributed discrete logarithms (DDLog)*, we enable the non-interactive authentication of powers of linear combinations of shares of some value. This, in turn, allows us to express constraints with polynomially bounded Waring rank.

Our construction is single-key, selectively secure, and supports an exponential-size domain.

Table of Contents

1 Introduction

Pseudorandom functions (PRFs) are a fundamental tool in both the theory and practice of cryptography. A constrained PRF (CPRF) allows a master secret PRF key to be used to derive a constrained key that permits evaluation of the PRF only on a restricted set of inputs. This restriction is determined by a predicate C, such that the constrained key allows evaluation at all points where $C(x) = 0$, while for any x where $C(x) = 1$, the PRF output remains pseudorandom.[3]

Some CPRFs support *private constraints* (or are constraint-hiding), meaning that the constrained key reveals no information about the specific constraint used, beyond the allowed class of constraints.

Originally, constrained PRFs were proposed for applications such as identity-based key exchange and broadcast encryption [BW13]. Later, they played a key role in the punctured programming technique [SW14], which enabled many applications of indistinguishability obfuscation. CPRFs have also been used to construct watermarkable PRFs, searchable encryption, private keyword search, and symmetric-key deniable encryption [BLW17].

One of the most fundamental constraint classes is *puncturing*, where $C(x) = 1$ if and only if $x = x^\star$. In this case, a constrained key—called a punctured key—allows evaluation of the PRF at all points except for $x^\star$. Puncturable PRFs can be built using the GGM construction [GGM84], where a PRF is constructed from a tree of length-doubling pseudorandom generators (PRGs). However, in GGM-style constructions [BW13, KPTZ13, BGI14], the punctured key reveals the punctured point $x^\star$, meaning that the construction is not private.

Privately puncturable PRFs were first constructed by Boneh et al. [BLW17] using multilinear maps or indistinguishability obfuscation, achieving constrained PRFs for general polynomial-size circuits. Later constructions obtained privately constrained PRFs for circuits under the learning with errors (LWE) assumption [BTVW17, CC17] and were further extended in various ways, such as enabling programmability [PS18] and adaptive security [HKKW19, DKN+20]. Alternative constructions support weaker classes of predicates or security properties while avoiding the full power of LWE. These include constructions based on DDH [AMN+18], DCR [CMPR23], and even solely one-way functions [DKN+20, Ser24].

Notably, however, none of these constructions achieve a *private* puncturable PRF without relying on either LWE or indistinguishability obfuscation. The only exception is the work of Boyle et al. [BGIK22], which is based solely on one-way functions but requires the PRF domain to be polynomially sized (as generating a constrained key requires iterating over the entire domain). Thus, this construction is more akin to a "privately programmable incremental PRG." This leaves the following challenging open problem:

Is it possible to construct a privately puncturable PRF with a superpolynomial domain size without relying on LWE or indistinguishability obfuscation?

Our Contributions. We construct the first privately puncturable PRF from the decisional composite residuosity (DCR) assumption that supports an exponentially large domain, resolving the question posed above. In fact, we construct a private CPRF for a broader class of functions—namely, functions with polynomially bounded Waring rank—that includes puncturing.

A function has polynomially bounded Waring rank if it can be expressed as the sum of at most polynomially many powers of linear functions, and has polynomial degree. This class

[3] The convention of using $C(x) = 0$ instead of $C(x) = 1$ for inputs that satisfy the constraint simplifies the exposition of our construction.

encompasses many useful functions, including puncturing, as well as all log-local functions, inner product membership [BCM+24], and more.

Our private CPRF construction follows the homomorphic secret-sharing-based template for building private CPRFs introduced by Couteau, Meyer, Passelègue, and Riahinia [CMPR23]. In essence, the CPRF key and constrained keys are defined as the values needed for non-interactively computing (subtractive) secret shares of $C(x) \cdot F_K(x)$, where C represents the constraint and F is a PRF. Notably, the CPRF outputs are the *shares* of this value, and the PRF F is not the PRF being constrained, but rather a tool for ensuring that when $C(x) = 1$, the shares remain pseudorandom. Specifically, when $C(x) = 0$, both parties hold the same shares (i.e., the constrained key produces the same output as the master key), whereas when $C(x) = 1$, the PRF F ensures that the two shares are pseudorandom.

The construction of [CMPR23] supports only constraints that can be expressed as inner products, as it relies on a special form of "offline-online" homomorphic secret sharing—a two-party, authenticated secret-sharing scheme where one party's share x_0 of an input x can be generated independently of x, and the other share x_1 is later generated using x_0 and x. Due to this restriction, offline-online shared values support only linear homomorphism, limiting their CPRF construction to inner-product constraints.[4] We overcome this limitation via a new technique based on distributed discrete logarithms (DDLog) and novel methods for homomorphically combining authenticated secret shares. This technique may be of independent interest and lead to further applications.

Comparison. There is a vast body of literature on constructing constrained PRFs under various cryptographic assumptions. In Table 1, we summarize prior and concurrent results that rely on group-based assumptions or one-way functions, as these are most relevant to our work. The most expressive prior constructions are those of [AMN+18], which is based on a variant of DDH in pairing-free groups, and [CMPR23], which relies on DCR or class group assumptions. Both of these constructions support arbitrary NC1 constraints; however, neither achieves constraint hiding.

Previous private CPRFs with superpolynomial domain sizes support constraints such as inner products, bit-fixing, or t-CNF for constant $t = O(1)$. Despite their apparent generality, none of these function classes are expressive enough to capture puncturing, which is inherently a high-degree constraint.

Concurrent Work. Ishai, Li, and Lin [ILL24] concurrentlydeveloped new CPRF constructions extending the CMPR template. Their work introduces techniques for succinct partial garbling, allowing them to construct CPRFs for any constraint expressible by a circuit. However, because their approach relies on partial garbling, their construction does not achieve privacy. Thus, our results and those of [ILL24] extend the CMPR template in different directions. An interesting open question is whether both approaches can be combined to achieve a private CPRF for general circuits.

2 Technical Overview

2.1 The CMPR Template and Its Limitations.

Couteau, Meyer, Passelègue, and Riahinia [CMPR23] provided a template for building (privately) constrained PRFs using homomorphic secret sharing (HSS) techniques. The key idea is that the task can be viewed as enabling two parties—the master key holder and the constrained key holder—to compute pseudorandom shares of the value $C(x) \cdot \mathsf{PRF}_k(x)$ for any

[4] [CMPR23] also obtained a CPRF for NC1 circuits, but this construction is not private.

Table 1: Constructions of CPRFs using group-based assumptions or one-way functions. All selectively secure constructions can be made adaptively secure in the random oracle model.

	Private	Constraints	Adaptive	Domain Size	Assumption
[BW13], [KPTZ13], [BGI14]	✗	Prefix/puncturing	✗	exp	OWF
[BW13]	✗	Left-right	✓	exp	Bilinear DDH + ROM
[AMN+18]	✗	NC1	✗	exp	L-DDHI
[CMPR23]	✗	NC1	✗	exp	DCR/class groups
[ILL24]	✗	P/Poly	✗	exp	DCR/class groups
[AMN+18]	✓	Bit-fixing	✗	exp	DDH
[DKN+20]	✓	t-CNF	✓	exp	OWF
[CMPR23]	✓	Inner product	✗	exp	DCR/class groups
[Ser24]	✓	Inner product	✓	exp	ROM
[Ser24]	✓	Inner product	✗	exp	DDH
[Ser24]	✓	Inner product	✗	poly	OWF
[BGIK22]	✓	Puncturing	✗	poly	OWF
Ours	✓	Bounded Waring Rank (incl. Puncturing)	✗	exp	DCR

input x of the CPRF. If $C(x) = 0$ (meaning the input x is authorized), both shares are equal, whereas if $C(x) = 1$, they are offset by $\mathsf{PRF}_k(x)$. Provided the constrained key does not reveal k, this ensures that unauthorized evaluations remain masked.

The main challenge in instantiating this template is designing a method for the parties to compute pseudorandom subtractive shares of $\mathsf{sk} \cdot C(x)$, where sk is the secret key of an appropriately chosen encryption scheme. In this work, we use the Damgård-Jurik-ElGamal scheme, whose security follows from DCR. This must be done while satisfying the following constraints:

1. The master key sk must be sampled independently of C.
2. The constrained key ck should hide sk.
3. (*Constraint privacy*) ck should hide C.

Going from shares of $\mathsf{sk} \cdot C(x)$ to shares of $C(x) \cdot \mathsf{PRF}_k(x)$, thereby obtaining a (privately) constrained PRF, is then handled by the CMPR template.

2.2 Our Solution: Authenticating Powers of Shares

We write $\langle x \rangle$ for *subtractive shares* of x, and we denote by $\langle x \rangle_\sigma$ the share held by party P_σ. We call $\langle \mathsf{sk} \cdot x \rangle$ an *authenticated share* of x. Similarly, we refer to $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot x)$ as an *authenticated encryption* of x.

Now assume that two parties hold shares and authenticated shares of each input bit (*i.e.* $\langle x_i \rangle_\sigma$ and $\langle \mathsf{sk} \cdot x_i \rangle_\sigma$), as well as encryptions and authenticated encryptions of the first party's input shares (*i.e.* $\mathsf{Enc}_{\mathsf{sk}}(\langle x_i \rangle_0)$, $\mathsf{Enc}_{\mathsf{sk}}(\langle \mathsf{sk} \cdot x_i \rangle_0)$, and $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot \langle x_i \rangle_0)$). We now explain how they can locally compute shares of $\mathsf{sk} \cdot f(x)$, where f is an arbitrary function with polynomially bounded Waring rank.

Circuit Randomization Step. Our goal is to express the constraint function f in a form that allows efficient computation over secret shares. To achieve this, we use the Waring rank decomposition.

Let f be a degree-d polynomial over $\{0,1\}^m$ with Waring rank at most r. [5] By definition of Waring rank, there exist affine functions $f_1, \ldots, f_r$, scalars $\alpha_1, \ldots, \alpha_r$, and degrees $d_1, \ldots, d_r \leq d$ such that:

$$\begin{aligned} f(x) &= \sum_{i=1}^{r} \alpha_i \cdot [f_i(x)]^{d_i} && \text{(by definition of Waring rank)} \\ &= \sum_{i=1}^{r} \alpha_i \cdot [\underbrace{f_i'(\langle x \rangle_1)}_{:= f_i(\langle x \rangle_1) - f_i(0,\ldots,0)} + f_i(-\langle x \rangle_0)]^{d_i} && \text{(by affinity of the } f_i\text{)} \\ &= \sum_{i=1}^{r} \sum_{j=0}^{d_i} \alpha_i \binom{d_i}{j} [f_i'(\langle x \rangle_1)]^j \cdot [f_i(-\langle x \rangle_0)]^{d_i - j} && \text{(by the binomial theorem)} \end{aligned}$$

Thus, computing subtractive shares of $f(x)$ (resp. $\mathsf{sk} \cdot f(x)$) reduces to computing subtractive shares of $[L(\langle x \rangle_1)]^j \cdot [A(\langle x \rangle_0)]^i$ (resp. $\mathsf{sk} \cdot [L(\langle x \rangle_1)]^j \cdot [A(\langle x \rangle_0)]^i$), where L is linear and A is affine.[6]

Authenticating Powers of linear combination of shares. The main technical contribution of this work is a new method that enables non-interactive evaluation of powers of authenticated shares. From a technical perspective, both our work and the concurrent work of [ILL24] are inspired by the techniques for non-interactive computation on shares presented in [MORS24] in the context of arithmetic garbling, but extend them in different directions.

Before detailing how the parties obtain shares for each term, recall that DDLog enables them to non-interactively compute shares of the product of a secret-shared value and an encrypted value. Specifically, if the parties hold $\langle X \rangle$, $\langle \mathsf{sk} \cdot X \rangle$, and $\mathsf{Enc}_{\mathsf{sk}}(Y)$, they can use DDLog to derive $\langle X \cdot Y \rangle$ without additional interaction.
At a high level, our approach consists of the following steps: The first step is to obtain shares of $[L(\langle x \rangle_1)]^j$ and $\mathsf{sk} \cdot [L(\langle x \rangle_1)]^j$, which the parties compute inductively.

Initialisation step ($j = 1$): This step requires computing shares of $L(\langle x \rangle_1)$ and $\mathsf{sk} \cdot L(\langle x \rangle_1)$. The first part is straightforward, as the parties already hold tautological shares of $\langle x \rangle_1$—meaning that party 1 knows $\langle x \rangle_1$ "in the clear," and party 0 can simply set their own share to 0. Since subtractive shares are compatible with linear transformations:

$$\langle L(\langle x \rangle_1) \rangle_\sigma \leftarrow \begin{cases} L(\langle x \rangle_1) & \text{if } \sigma = 1 \\ 0 & \text{if } \sigma = 0 \end{cases}$$

To compute shares of $\mathsf{sk} \cdot L(\langle x \rangle_1)$, we observe that:

$$\begin{aligned} \mathsf{sk} \cdot L(\langle x \rangle_1) - \mathsf{sk} \cdot L(\langle x \rangle_0) &= \mathsf{sk} \cdot L(x) \\ &= L(\mathsf{sk} \cdot x) \\ &= L(\langle \mathsf{sk} \cdot x \rangle_1) - L(\langle \mathsf{sk} \cdot x \rangle_0) \ . \end{aligned} \tag{1}$$

This allows the parties to compute the desired shares as:

$$\langle \mathsf{sk} \cdot L(\langle x \rangle_1) \rangle_\sigma \leftarrow \begin{cases} L(\langle \mathsf{sk} \cdot x \rangle_1) & \text{if } \sigma = 1 \\ L(\langle \mathsf{sk} \cdot x \rangle_0) - \mathsf{sk} \cdot L(\langle x \rangle_0) & \text{if } \sigma = 0. \end{cases}$$

[5] Formally, we use the mixed Waring rank notion here. Later in the paper, we prove that the standard and mixed Waring rank notions are polynomially related, meaning our results extend to both cases.

[6] Remember that all functions here take as input vectors of bits, e.g., $f(x) = f(x_1, \ldots, x_n)$, and $L(\langle x \rangle_1) = L(\langle x_1 \rangle_1, \ldots, \langle x_n \rangle_1)$, etc.

Induction step: Observe that the first part (computing shares of $[L(\langle x\rangle_1)]^j$) follows the same tautological approach as before. For the second part (computing shares of $\mathsf{sk}\cdot[L(\langle x\rangle_1)]^j$), consider the identity:

$$\begin{aligned}
\mathsf{sk}\cdot[L(\langle x\rangle_1)]^j &= (\mathsf{sk}\cdot L(\langle x\rangle_1))\cdot[L(\langle x\rangle_1)]^{j-1} \\
&= \big(\mathsf{sk}\cdot L(\langle x\rangle_0) + L(\langle \mathsf{sk}\cdot x\rangle_1) - L(\langle \mathsf{sk}\cdot x\rangle_0)\big)\cdot[L(\langle x\rangle_1)]^{j-1} \qquad \text{(by eq. (1))} \\
&= \underbrace{\mathsf{sk}\cdot[L(\langle x\rangle_1)]^{j-1}\cdot L(\langle x\rangle_0)}_{\text{Term 1}} + \underbrace{L(\langle \mathsf{sk}\cdot x\rangle_1)\cdot[L(\langle x\rangle_1)]^{j-1}}_{\text{Term 2}} \\
&\quad - \underbrace{[L(\langle x\rangle_1)]^{j-1}\cdot L(\langle \mathsf{sk}\cdot x\rangle_0)}_{\text{Term 3}}
\end{aligned}$$

The parties obtain shares of each term as follows:

1. **First term.** The parties hold shares of $[L(\langle x\rangle_1)]^{j-1}$ and $\mathsf{sk}\cdot[L(\langle x\rangle_1)]^{j-1}$ (by the induction hypothesis), as well as $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk}\cdot\langle x\rangle_0)$. Using the linear homomorphic properties[7] of Enc, they compute $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk}\cdot L(\langle x\rangle_0))$. By combining $\langle[L(\langle x\rangle_1)]^{j-1}\rangle$, $\langle\mathsf{sk}\cdot[L(\langle x\rangle_1)]^{j-1}\rangle$, and the derived ciphertext with DDLog, they directly obtain shares of Term 1.
2. **Second term.** Since party 1 already knows $\langle\mathsf{sk}\cdot x\rangle_1$ and $\langle x\rangle_1$, they can compute the second term locally. The parties then hold tautological shares:

$$\langle L(\langle\mathsf{sk}\cdot x\rangle_1)\cdot[L(\langle x\rangle_1)]^{j-1}\rangle_\sigma \leftarrow \begin{cases} L(\langle\mathsf{sk}\cdot x\rangle_1)\cdot[L(\langle x\rangle_1)]^{j-1} & \text{if } \sigma = 1 \\ 0 & \text{if } \sigma = 0 \end{cases}$$

3. **Third term.** Using the same approach as in the first term, the parties already hold shares of $[L(\langle x\rangle_1)]^{j-1}$ and $\mathsf{sk}\cdot[L(\langle x\rangle_1)]^{j-1}$ (from the induction hypothesis), as well as $\mathsf{Enc}_{\mathsf{sk}}(\langle\mathsf{sk}\cdot x\rangle_0)$. They can derive a common ciphertext $\mathsf{Enc}_{\mathsf{sk}}(L(\langle\mathsf{sk}\cdot x\rangle_0))$ via linear homomorphism and then use DDLog to compute shares of Term 3.

Sharing Mixed Term Products. Using linear homomorphism, the parties can derive the same ciphertext $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk}\cdot A(\langle x\rangle_0))$. By leveraging their shares of $[L(\langle x\rangle_1)]^j$ and $\mathsf{sk}\cdot[L(\langle x\rangle_1)]^j$ (as computed in the previous steps), the parties can recursively apply DDLog—specifically, with $X_i \leftarrow [L(\langle x\rangle_1)]^j\cdot[A(\langle x\rangle_0)]^{i-1}$ and $Y \leftarrow \mathsf{sk}\cdot A(\langle x\rangle_0)$—to obtain shares of $\mathsf{sk}\cdot[L(\langle x\rangle_1)]^j\cdot[A(\langle x\rangle_0)]^i$.

2.3 Waring Rank

We now analyze the expressiveness of the class of constraints that our technique can handle. As already mentioned, our technique allows us to evaluate constraints with polynomially bounded Waring rank: The Waring rank of a polynomial f of degree d is the smallest number of linear polynomials whose $d_i < d$ powers sum to f. More formally, for some scalars α_i and linear polynomials L_i, it holds that

$$f(x) = \sum_{i=1}^{r} \alpha_i (L_i(x))^{d_i},$$

where r is the Waring rank of f. For the special case of $d = 2$, the Waring rank equals the rank of the corresponding symmetric matrix representation of f.

[7] It is important that the homomorphic evaluation be a deterministic procedure, ensuring both parties hold the *same* ciphertext.

Functions with polynomially bounded Waring rank include log-local functions, since they can be decomposed into sums of products of logarithmically many variables, and each such product can be handled by lemma 13. Another example is inner product membership, which follows from the same techniques we describe below for puncturing.

Applications of Waring Rank. Originally studied in invariant theory during the 19th century [IK99, Introduction], Waring rank has since found applications in several areas of theoretical computer science. In algebraic complexity theory, it characterizes one of the simplest yet non-trivial complexity classes, namely homogeneous depth-3 diagonal circuits [DGI+24, Lan17, EGdOW18]. The problem of computing the matrix multiplication exponent ω, which determines the fastest known algorithm for matrix multiplication, has been shown to be closely related to Waring rank [CHI+18]. More recently, Pratt [Pra19] established an intriguing connection between algebraic algorithms for graph problems and upper bounds on the Waring rank of certain polynomials. Our work introduces a new connection between cryptography and Waring rank, showing how this algebraic notion captures a class of constraints for pCPRFs.

Waring Rank of Puncturing. Note that the puncturing constraint can be expressed as a polynomial with polynomially bounded degree and polynomially bounded Waring rank. Intuitively, $C_{x^\star}(x) = 1$ (remember we want the constraint to be 1 for the punctured point) iff all the bits of x and $x^\star$ are the same. Thus, we can define a polynomial of degree n (for n-bit inputs), call it $A_{x^\star}$, that counts the number of common bits between x and $x^\star$. The puncturing constraint can then be rewritten as a product of $(A_{x^\star}(x) - i)$ for all $i = 0, \ldots, n-1$, resulting in 0 iff the number of common bits is between 0 and n these terms. This polynomial can then be normalized to ensure that $C_x(x^\star) \in \{0, 1\}$.

Rounding Rational Shares Normalizing the puncturing polynomial requires division, so the coefficients α_i will not be rationals. This is typical for expressing functions in Waring rank form – the expressive power is not great unless you have a sufficiently large field. Fortunately, we can work over $\mathbb{Q}$, and convert to integer shares as a final step. That is, if $y_0, y_1 \in \mathbb{Q}$ are rational shares of an integer $y_1 - y_0 = y$, then $\lfloor y_0 \rfloor, \lfloor y_1 \rfloor$ will be integer shares of the same y. Therefore, as long as f outputs integers, we can tolerate fractional coefficients in its definition.

3 Preliminaries

3.1 Damgård-Jurik-ElGamal Cryptosystem

Definition 1 (Decision Composite Residuosity Assumption (DCR), [Pai99]). *Let* $\mathsf{RSA.Gen}$ *be a polynomial-time algorithm which, on input a security parameter λ, outputs (N, p, q) where p and q are λ-bit primes and $N = pq$. Let λ be a security parameter. We say that the* Decision Composite Residuosity (DCR) *problem is hard relative to modulus-sampling algorithm* $\mathsf{RSA.Gen}$ *if*

$$\left\{ (N, x) \colon \begin{array}{r} (N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda) \\ x \xleftarrow{\$} (\mathbb{Z}/N^2\mathbb{Z})^\times \end{array} \right\} \overset{\mathrm{c}}{\approx} \left\{ (N, x^N \bmod N^2) \colon \begin{array}{r} (N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda) \\ x \xleftarrow{\$} (\mathbb{Z}/N^2\mathbb{Z})^\times \end{array} \right\} .$$

Theorem 2. *Assuming DCR, Damgård–Jurik–ElGamal encryption fig. 1 is a public key encryption scheme satisfying correctness and KDM security for affine functions of the key. Specifically, the following properties hold:*

Correctness: $\mathsf{DJE.Dec}_{\mathsf{sk}}(\mathsf{DJE.Enc}_{\mathsf{pk}}(x)) = x$, *for any* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{DJE.KeyGen}$, *and any* $x \in \mathbb{Z}/N^{\zeta}\mathbb{Z}$.

KDM Security: *For all* p.p.t. *adversaries* Adv, *the oracles* $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}$ *and* $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}$ *are indistinguishable in the following experiment*

$$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^{\lambda})$$
$$\text{output } \mathsf{Adv}^{\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R/\$}(\mathsf{pk})}$$

where these oracles are defined as

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}(x, y)$:
$(k, N) \leftarrow \mathsf{sk}$
$z \leftarrow x \cdot k + y$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}(x, y)$:
$(k, N) \leftarrow \mathsf{sk}$
$z \xleftarrow{\$} \mathbb{Z}/N^{\zeta}\mathbb{Z}$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

Proof. This is very similar to many previous encryption schemes used in HSS and proven KDM secure in that context. See, e.g., the Damgård–Jurik instantiation of the NIDLS framework [ADOS22]), though details differ, as we do not require p and q to be safe primes. For completeness, we present a full proof in appendix A.2. □

DJE Damgård-Jurik-ElGamal Cryptosystem

`Requires:`

- $\zeta \geq 1$ is a parameter defining the plaintext size.
- Group isomorphism $\exp\colon (\mathbb{Z}/N^{\zeta}\mathbb{Z})^{+} \to 1 + N(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})$ and its inverse $\log\colon 1 + N(\mathbb{Z}/N^{\zeta+1}\mathbb{Z}) \to (\mathbb{Z}/N^{\zeta}\mathbb{Z})^{+}$, as defined as in [RS21]:

$$\exp(x) = \sum_{k=0}^{\zeta} \frac{(Nx)^k}{k!} \qquad \text{and} \qquad \log(1+Nx) = \sum_{k=1}^{\zeta} \frac{(-N)^{k-1}x^k}{k}$$

$\mathsf{DJE.KeyGen}(1^{\lambda})$:

1. Sample $(N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^{\lambda})$ [a]
2. Sample $k \xleftarrow{\$} [0, N)$
3. Sample $g \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$
4. Compute $h \leftarrow g^{-k}$
5. Output $(\mathsf{sk} = (k, N), \mathsf{pk} = (g, h, N))$

[a] Note that p and q are unused, so N could instead be a CRS.

$\mathsf{DJE.Enc}_{\mathsf{pk}}(x)$:

1. Parse $\mathsf{pk} = (g, h, N)$
2. Sample $r \xleftarrow{\$} [0, N)$
3. Compute $c_0 \leftarrow g^r$
4. Compute $c_1 \leftarrow h^r \cdot \exp(x)$
5. Output $c = (c_0, c_1)$

$\mathsf{DJE.Dec}_{\mathsf{sk}}(c = (c_0, c_1))$:

1. Parse $\mathsf{sk} = (k, N)$
2. Assert $c_0^k \cdot c_1 \equiv 1 \mod N$
3. Output $x \leftarrow \log(c_0^k \cdot c_1)$

Fig. 1: The Damgård-Jurik-ElGamal cryptosystem.

3.2 Distributed Discrete Logarithm

DDLOG Damgård-Jurik Distance Function [RS21]

$\mathsf{DDLog}_N(h \in \mathbb{Z}/N^{\zeta+1}\mathbb{Z})$:
Compute and output $z \leftarrow \log\left(\frac{h}{h \bmod N}\right) \in \mathbb{Z}/N^{\zeta}\mathbb{Z}$

Fig. 2: [RS21]'s distributed discrete logarithm for the Damgård-Jurik cryptosystem [DJ01].

Lemma 3 (Distributed Decryption). *If we have shares over $\mathbb{Z}$ of $\langle x\rangle_1 - \langle x\rangle_0 = x$ and $\langle k \cdot x\rangle_1 - \langle k \cdot x\rangle_0 = k \cdot x$, then*

$$\mathsf{DDLog}_N(c_0^{\langle k\cdot x\rangle_1} c_1^{\langle x\rangle_1}) - \mathsf{DDLog}_N(c_0^{\langle k\cdot x\rangle_0} c_1^{\langle x\rangle_0}) \equiv x \cdot y \mod N^{\zeta}$$

always holds, for every choice of plaintext size $\zeta \geq 1$, key pair $(\mathsf{sk} = (k, N), \mathsf{pk}) \in \mathsf{Supp}(\mathsf{DJE.KeyGen}(1^{\lambda}))$, plaintext $y \in \mathbb{Z}/N^{\zeta}\mathbb{Z}$, ciphertext $(c_0, c_1) \in \mathsf{Supp}(\mathsf{DJE.Enc}_{\mathsf{pk}}(y))$, and scalar $x \in \mathbb{Z}/N^{\zeta}\mathbb{Z}$.

Proof. Taking the ratio of the inputs to DDLog, we get

$$\frac{c_0^{\langle k\cdot x\rangle_1} c_1^{\langle x\rangle_1}}{c_0^{\langle k\cdot x\rangle_0} c_1^{\langle x\rangle_0}} = c_0^{k\cdot x} c_1^{x} = \exp(\mathsf{DJE.Dec}_{\mathsf{sk}}(c_0, c_1))^x = \exp(x \cdot y).$$

The second and third equalities are from the definition of DJE.Dec and the correctness of DJE, respectively. The inputs to the DDLogs are therefore multiplicative shares of $\exp(x \cdot y)$, which, by [RS21, Theorem 18], makes the outputs of the DDLogs additive shares over $\mathbb{Z}/N^{\zeta}\mathbb{Z}$ of $x \cdot y$. □

Lemma 4 (Adapted from [RS21, Lemma 19]). *For all moduli $M > 1$ and all modulo M shares $\langle x\rangle_0, \langle x\rangle_1 \in \mathbb{Z}/M\mathbb{Z}$ of some $x \in \mathbb{Z}$, we have*

$$\Pr_{r \xleftarrow{\$} \mathbb{Z}/M\mathbb{Z}}\Big[(\langle x\rangle_1 + r) \bmod M - (\langle x\rangle_0 + r) \bmod M = x\Big] = \max\left(1 - \frac{|x|}{N^{\zeta}}, 0\right).$$

3.3 Privately Constrained Pseudorandom Function

We use the simulation-based definition of Peikert-Shiehian [PS18], which simultaneously captures privacy of the constraining function, pseudorandomness on unauthorised inputs, and correctness of constrained evaluation on authorised inputs.

Definition 5 (Privately Contrained Pseudorandom Function). *A* privately constrained PRF *(pCPRF) with input space $\{0,1\}^n$ and output space $\{0,1\}^{\mu}$ and supporting as constraints a class of class of circuits $\mathcal{C}$ (where each circuit in $\mathcal{C}$ is from $\{0,1\}^n$ to $\{0,1\}$) is a tuple of algorithms* $\mathsf{PCPRF} = (\mathsf{PCPRF.KeyGen}, \mathsf{PCPRF.Eval}, \mathsf{PCPRF.Constrain}, \mathsf{PCPRF.CEval})$ *with the following syntax and properties:*

- $\mathsf{PCPRF.KeyGen}(1^{\lambda}) \to \mathsf{msk}$*: On input a security parameter 1^{λ}, the key generation algorithm* KeyGen *produces a master secret key* msk*.*
- $\mathsf{PCPRF.Eval}(1^{\lambda}, \mathsf{msk}, x) \to y$*: On input a security parameter 1^{λ}, a master secret key* msk*, and an input $x \in \{0,1\}^n$, the evaluation algorithm* Eval *output an element $y \in \{0,1\}^{\mu}$.*

– $\mathsf{PCPRF.Constrain}(1^\lambda, \mathsf{msk}, C) \to \mathsf{ck}$*: On input a security parameter* 1^λ*, a master secret key* msk*, and a constraining circuit* $C \in \mathcal{C}$*, the constraining algorithm* $\mathsf{Constrain}$ *outputs a constrained key* ck*.*

– $\mathsf{PCPRF.CEval}(1^\lambda, \mathsf{ck}, x) \to y$*: On input a security parameter* 1^λ*, a constrained secret key* ck*, and an input* $x \in \{0,1\}^n$*, the constrained evaluation algorithm* CEval *output an element* $y \in \{0,1\}^\mu$*.*

– **Single-key selective PCPRF security.** *A privately constrained PRF is* single-key selectively secure *if for any* p.p.t. *adversary* $\mathcal{A}$ *(which we require to never repeat a query) and any* $m \in \{0,1\}^\lambda$ *there exists a* p.p.t. *simulator* $\mathcal{S}$ *such that:*

$$\left\{\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^m)\right\}_{\lambda\in\mathbb{N}} \overset{\mathrm{c}}{\approx} \left\{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^m)\right\}_{\lambda\in\mathbb{N}}$$

where Real *and* Ideal *are the respective views of* $\mathcal{A}$ *in the experiments of fig. 3.*

Experiment Privately Constrained PRF Security

$\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^m)$	$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^m)$
$C \xleftarrow{\$} \mathcal{A}(1^\lambda, 1^m)$	$C \xleftarrow{\$} \mathcal{A}(1^\lambda, 1^m)$
$\mathsf{msk} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$	
$\mathsf{ck} \xleftarrow{\$} \mathsf{Constrain}(1^\lambda, \mathsf{msk}, C)$	$\mathsf{ck} \xleftarrow{\$} \mathcal{S}(1^\lambda, 1^m)$
Send ck to $\mathcal{A}$	Send ck to $\mathcal{A}$
Repeat until $\mathcal{A}$ halts:	Repeat until $\mathcal{A}$ halts:
Wait to receive x from $\mathcal{A}$	Wait to receive x from $\mathcal{A}$
$y \xleftarrow{\$} \mathsf{Eval}(1^\lambda, \mathsf{msk}, x)$	If $C(x) = 0$: $y \xleftarrow{\$} \mathsf{CEval}(1^\lambda, \mathsf{ck}, x)$
	If $C(x) = 1$: $y \xleftarrow{\$} \{0,1\}^\mu$
Send y to $\mathcal{A}$	Send y to $\mathcal{A}$

Fig. 3: Real and Ideal worlds of PCPRF security (definition 5).

4 Share Conversion Procedures

In this section, we present "homomorphic share conversion" algorithms based on distributed discrete logarithms, which allow two parties holding appropriately encoded representation (namely, shares, authenticated shares, encryptions) of some inputs to compute shares of some function of these inputs. We emphasise that we make no security claims in this section, and only show that these algorithms output (not necessarily secret) shares of the correct result.

1. ***Multiplication.*** Section 4.1 (and specifically fig. 4) recalls from [BGI16] (with the "DCR variants" of [OSY21, RS21]) how to compute $\langle x \cdot y \rangle$ given $\langle x \rangle$, $\langle k \cdot x \rangle$, $\mathsf{Enc}_k(y)$.
2. ***Bounded Waring Rank.*** Section 4.2 (and specifically fig. 5) is the main technical contribution of this paper and shows how to compute $\langle (1, \mathsf{sk}) \cdot f(x) \rangle$, where f is any (public) polynomial with bounded[8] degree and Waring rank, given $\langle x_i \rangle$, $\langle \mathsf{sk} \cdot x_i \rangle$, $\mathsf{Enc}_{\mathsf{sk}}(\langle x_i \rangle_0)$, $\mathsf{Enc}_{\mathsf{sk}}(\langle \mathsf{sk} \cdot x_i \rangle_0)$, $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot \langle x_i \rangle_0)$.

[8] The bound can be an arbitrary polynomial in the security parameter.

3. ***RMS Programs with Leading Multiplication.*** Section 4.3 (and specifically fig. 6) recalls from [CMPR23] how to compute $\langle X \cdot \mathsf{RMS}(y_1, \ldots, y_1) \rangle$ for any RMS program RMS, given $\langle X \rangle$, $\langle \mathsf{sk} \cdot X \rangle$, $\mathsf{Enc}_{\mathsf{sk}}(y_i)$, and $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot y_i)$.

4.1 Homomorphic Share Conversion: Multiplication

Algorithm Homomorphic Share Conversion: Multiplication

$\mathsf{Mult}(\mathsf{sh}, \mathsf{auth\text{-}sh}, \mathsf{ct}, r)$: // Given $\langle x \rangle$, $\langle k \cdot x \rangle$, $\mathsf{Enc}_k(y)$, $r \in [N^\zeta]$ compute $\langle x \cdot y \rangle$ (randomised by r)

1. Parse ct as $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1)$
2. Output $\mathsf{DDLog}(\mathsf{ct}_0^{\mathsf{auth\text{-}sh}} \cdot \mathsf{ct}_1^{\mathsf{sh}}) + r \bmod N^\zeta$

Fig. 4: Homomorphic multiplication of secret shares, from DDLog "*à la* ElGamal", adapted from [BGI16, OSY21].

Lemma 6 (Correctness of share conversion procedure for multiplications). *Let $\lambda \in \mathbb{N}^\star$ be a security parameter and let $B = \lambda^{\mathcal{O}(1)}$. For all $(x, y) \in [0, B)^2$ such that $xy \leq B$ and for all $\mathsf{ct} \in \mathsf{Supp}(\mathsf{DJE.Enc}_{\mathsf{pk}}(y))$,*

$$\Pr\left[y_1 - y_0 = x \cdot y \colon (y_0, y_1) \xleftarrow{\$} \mathsf{Sam}(1^\lambda, x, y, \mathsf{ct})\right] \geq 1 - B/N^{\zeta - 1}$$

where the sampler Sam is defined as follows (where Mult is the deterministic algorithm of fig. 4 and DJE is the cryptosystem of fig. 2):
$\mathsf{Sam}(1^\lambda, x, y, \mathsf{ct})$:

1. $((k, N), \mathsf{pk}) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)$
2. $\mathsf{sh}_0 \xleftarrow{\$} [0, 2^\lambda B)$
3. $\mathsf{sh}_1 \leftarrow x + \mathsf{sh}_0$
4. $\mathsf{sh}_0^{\mathsf{auth}} \xleftarrow{\$} [0, N \cdot 2^\lambda B)$
5. $\mathsf{sh}_1^{\mathsf{auth}} \leftarrow k \cdot x + \mathsf{sh}_0^{\mathsf{auth}}$
6. $r \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$
7. *For* $\sigma \in \{0, 1\}$, $y_\sigma \leftarrow \mathsf{Mult}(\mathsf{sh}_\sigma, \mathsf{sh}_\sigma^{\mathsf{auth}}, \mathsf{ct}, r)$
8. *Output* (y_0, y_1)

Lemma 6 follows immediately from lemma 18.

4.2 Homomorphic Share Conversion: Bounded Waring Rank

Algorithm Homomorphic Share Conversion: Bounded Waring Rank

// Compute $\langle f(x_1, \ldots, x_n) \rangle_\sigma$ and $\langle k \cdot f(x_1, \ldots, x_n) \rangle_\sigma$.

`Requires:`

1. r and d are fixed bounds on the Waring rank and the degree of functions which can be evaluated homomorphically
2. DJE is the cryptosystem of fig. 1
3. Mult is the algorithm of fig. 4
4. HEval is the linearly homomorphic evaluation algorithm for the Damgård-Jurik-ElGamal encryption scheme

$\mathsf{ShareConv}(\sigma,$ // The party index
$k,$ // The key (only needed for $\sigma = 0$)
$(\alpha_i, f_i, d_i)_{i\in[r]},$ //$f = \sum_{i=1}^{r} \alpha_i f_i^{d_i}$
$(\mathsf{sh}_{\sigma,i})_{i\in[m]},$ // $\langle x_i \rangle_\sigma$ (λ-bit shares over $\mathbb{Z}$)
$(\mathsf{sh}^{\mathsf{auth}}_{\sigma,i})_{i\in[m]},$ // $\langle k \cdot x_i \rangle_\sigma$ ($(\lambda \cdot \log N)$-bit shares over $\mathbb{Z}$)
$(c_i = (c_{i,0}, c_{i,1}))_{i\in[m]},$ // An encryption under (k, N) of $\mathsf{sh}_{0,i}$
$(c^{\mathsf{auth}}_i = (c^{\mathsf{auth}}_{i,0}, c^{\mathsf{auth}}_{i,1}))_{i\in[m]},$ // An encryption of $\mathsf{sh}^{\mathsf{auth}}_{0,i}$
$(c^{\mathsf{ext\text{-}auth}}_i = (c^{\mathsf{ext\text{-}auth}}_{i,0}, c^{\mathsf{ext\text{-}auth}}_{i,1}))_{i\in[m]},$ // Encryption of $k \cdot \mathsf{sh}_{0,i}$
$(c_{\mathsf{cst}}, c^{\mathsf{auth}}_{\mathsf{cst}}),$ // Encryptions of 1 and k
$\mathsf{crs})$: // A random string in $(\mathbb{Z}/N^\zeta\mathbb{Z})^{4\cdot rd}$

1. For $i \in [r]$, $f'_i \leftarrow f_i - f_i(0, \ldots, 0)$ // Linear part of affine function f_i

2. Parse crs as $\mathsf{crs} = (\mathsf{crs}_{i,j,b})_{i\in[r],j\in[d],b\in[4]} \in [N^\zeta]^{4rd}$

3. Initialise table TAB_σ, indexed by $[r] \times [d] \times [d]$
   // $\mathsf{TAB}_\sigma[i][j][\ell]$ will be used to store $\langle [f'_i(\langle x \rangle_1)]^j \cdot [f_i(-\langle x \rangle_0)]^\ell \rangle_\sigma$

4. Initialise table $\mathsf{TAB}^{\mathsf{auth}}_\sigma$, indexed by $[r] \times [d] \times [d]$
   // $\mathsf{TAB}^{\mathsf{auth}}_\sigma[i][j][\ell]$ will be used to store $\langle k \cdot [f'_i(\langle x \rangle_1)]^j \cdot [f_i(-\langle x \rangle_0)]^\ell \rangle_\sigma$

5. **Compute the $\mathsf{Enc}(k \cdot f'_i(\langle x \rangle_0))$ (for $i \in [r]$).**

$$\mathsf{ct}'_i \leftarrow \mathsf{DJE.HEval}(f'_i, c^{\mathsf{ext\text{-}auth}}_1, \ldots, c^{\mathsf{ext\text{-}auth}}_n)$$

6. **Compute the $\mathsf{Enc}(f'_i(\langle k \cdot x \rangle_0))$ (for $i \in [r]$).**

$$\mathsf{ct}''_i \leftarrow \mathsf{DJE.HEval}(f'_i, c^{\mathsf{auth}}_1, \ldots, c^{\mathsf{auth}}_n)$$

7. **Compute the $\mathsf{Enc}(f_i(-\langle x \rangle_0))$ (for $i \in [r]$).**

$$\mathsf{ct}'''_i \leftarrow \mathsf{DJE.HEval}(g_i, c_1, \ldots, c_n, c_{\mathsf{cst}})$$

   where $g_i \colon (X_1, \ldots, X_n, X_{m+1}) \mapsto f'_i(-X_1, \ldots, -X_n) + f_i(0, \ldots, 0) \cdot X_{m+1}$

8. **Compute the $\mathsf{Enc}(k \cdot f_i(-\langle x \rangle_0))$ (for $i \in [r]$).**

$$\mathsf{ct}''''_i \leftarrow \mathsf{DJE.HEval}(g_i, c^{\mathsf{ext\text{-}auth}}_1, \ldots, c^{\mathsf{ext\text{-}auth}}_n, c^{\mathsf{auth}}_{\mathsf{cst}})$$

9. **Compute the $\langle k \cdot [f'_i(\langle x \rangle_1)]^j \rangle_\sigma$ (for $1 \le j \le i \le r$).**
   For $i = 1, \ldots, r$ do (in parallel):
   (a) For $j = 1, \ldots, d_i$ do (in parallel):

$$\mathsf{TAB}_\sigma[i][j][0] \leftarrow \begin{cases} [f'_i(\langle x_1 \rangle_1, \ldots, \langle x_n \rangle_1)]^j & \text{if } \sigma = 1 \\ 0 & \text{if } \sigma = 0 \end{cases}$$

   (b) Set $\mathsf{TAB}^{\mathsf{auth}}_\sigma[i][1][0]$ to

$$\begin{cases} f'_i(\mathsf{sh}^{\mathsf{auth}}_{1,1}, \ldots, \mathsf{sh}^{\mathsf{auth}}_{1,m}) & \text{if } \sigma = 1 \\ f'_i(\mathsf{sh}^{\mathsf{auth}}_{0,1}, \ldots, \mathsf{sh}^{\mathsf{auth}}_{0,m}) - k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) & \text{if } \sigma = 0 \end{cases}$$

(c) For $j = 2, \ldots, d_i$ (sequentially), set $\mathsf{TAB}_\sigma^{\mathsf{auth}}[i][j][0]$ to

$$\begin{cases} f_i'(\mathsf{sh}_{1,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}}) \cdot [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\ +\mathsf{Mult}(\mathsf{TAB}_1[i][j-1][0], \mathsf{TAB}_1^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}_i', \mathsf{crs}_{i,j,1}) \\ -\mathsf{Mult}(\mathsf{TAB}_1[i][j-1][0], \mathsf{TAB}_1^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}_i'', \mathsf{crs}_{i,j,2}) & \text{if } \sigma = 1 \\ \\ \mathsf{Mult}(\mathsf{TAB}_0[i][j-1][0], \mathsf{TAB}_0^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}_i', \mathsf{crs}_{i,j,1}) \\ -\mathsf{Mult}(\mathsf{TAB}_0[i][j-1][0], \mathsf{TAB}_0^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}_i'', \mathsf{crs}_{i,j,2}) & \text{if } \sigma = 0 \end{cases}$$

10. **Compute the $\langle \boldsymbol{k} \cdot [\boldsymbol{f_i'}(\langle \boldsymbol{x} \rangle_1)]^{\boldsymbol{j}} \cdot [\boldsymbol{f_i}(-\langle \boldsymbol{x} \rangle_0)]^{\boldsymbol{\ell}} \rangle_{\boldsymbol{\sigma}}$ (for $\boldsymbol{0 \le j < i \le r}$ and $\boldsymbol{\ell \in [d_i]}$).**
For $i = 1, \ldots, r$ do (in parallel):
(a) Set $\mathsf{TAB}_\sigma[i][0][d_i]$ to

$$\begin{cases} 0 & \text{if } \sigma = 1 \\ [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i} & \text{if } \sigma = 0 \end{cases}$$

(b) Set $\mathsf{TAB}_\sigma^{\mathsf{auth}}[i][0][d_i]$ to

$$\begin{cases} 0 & \text{if } \sigma = 1 \\ k \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i} & \text{if } \sigma = 0 \end{cases}$$

(c) For $j = 1, \ldots, d_i$ do (in parallel):
For $\ell = 1, \ldots, d_i - j$ do (sequentially):

$$\begin{aligned} \mathsf{TAB}_\sigma[i][j][\ell] &\leftarrow \mathsf{Mult}(\mathsf{TAB}_\sigma[i][j][\ell-1], \mathsf{TAB}_\sigma^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i''', \mathsf{crs}_{i,j,3}) \\ \mathsf{TAB}_\sigma^{\mathsf{auth}}[i][j][\ell] &\leftarrow \mathsf{Mult}(\mathsf{TAB}_\sigma[i][j][\ell-1], \mathsf{TAB}_\sigma^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i'''', \mathsf{crs}_{i,j,4}) \end{aligned}$$

11. **Compute $\langle \boldsymbol{f}(\boldsymbol{x}) \rangle_{\boldsymbol{\sigma}}$ and $\langle \boldsymbol{k} \cdot \boldsymbol{f}(\boldsymbol{x}) \rangle_{\boldsymbol{\sigma}}$.**

$$y_\sigma \leftarrow \sum_{i=1}^{r} \sum_{j=0}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot \mathsf{TAB}_\sigma[i][j][d_i - j]$$

$$y_\sigma^{\mathsf{auth}} \leftarrow \sum_{i=1}^{r} \sum_{j=0}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot \mathsf{TAB}_\sigma^{\mathsf{auth}}[i][j][d_i - j]$$

12. Output $(\lfloor y_\sigma \rfloor, \lfloor y_\sigma^{\mathsf{auth}} \rfloor)$

Fig. 5: Share conversion precedure for homomorphically evaluating functions of bounded Waring rank.

Core Lemma 1 (Correctness of share conversion procedure for bounded Waring rank)**.** *Let*

$\lambda \in \mathbb{N}^*$ be a security parameter, let $(\lambda_{\mathsf{DCR}}, \zeta)$ be parameters for the Damgård-Jurik-ElGamal cryptosystem, let r, d be fixed polynomials in the security parameter, and let $f = \sum_{i=1}^{r} \alpha_i \cdot f_i^{d_i}$ be a integer-valued degree-d polynomial with Waring rank at most r (where $f_1, \ldots, f_r$ are affine functions and $\alpha_1 \ldots, \alpha_r \in \mathbb{Q}$), and consider the deterministic algorithm $\mathsf{ShareConv}$ *of fig. 5 parameterised by Damgård-Jurik parameter ζ. Let β be a bound on the coefficients of the f_i. For any inputs $x_1, \ldots, x_n \in [0, B)^m$,*

$$\Pr\left[y_1 - y_0 = \mathsf{sk} \cdot f(x_1, \ldots, x_n) \colon (\mathsf{sk}, y_0, y_1) \xleftarrow{\$} \mathsf{Sam}(1^\lambda, (x_i)_{i=1}^m)\right] \geq 1 - \frac{4rd^2(nB2^\lambda\beta)^d}{(2^{\lambda_{\mathsf{DJE}}-1})^{\zeta-1}}$$

when $f(x_1, \ldots, x_n) \in \mathbb{Z}$, where the sampler Sam *is defined as:*

$\mathsf{Sam}(1^\lambda, (x_i)_{i\in[m]})$:

1. $((k, N), \mathsf{pk}) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)$
2. $\mathsf{sh}_{0,i} \xleftarrow{\$} [0, 2^\lambda B)$
3. $\mathsf{sh}_{1,i} \leftarrow x_i + \mathsf{sh}_{0,i}$
4. $\mathsf{sh}_{0,i}^{\mathsf{auth}} \xleftarrow{\$} [0, N \cdot 2^\lambda B)$
5. $\mathsf{sh}_{1,i}^{\mathsf{auth}} \leftarrow k \cdot x_i + \mathsf{sh}_{0,i}$
6. $c_i \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sh}_{0,i})$
7. $c_i^{\mathsf{auth}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sh}_{0,i}^{\mathsf{auth}})$
8. $c_i^{\mathsf{ext\text{-}auth}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sk} \cdot \mathsf{sh}_{0,i})$
9. $\mathsf{crs} \xleftarrow{\$} (\mathbb{Z}/N^\zeta\mathbb{Z})^{4rd}$
10. *For* $\sigma \in \{0, 1\}$,
    $y_\sigma \leftarrow \mathsf{ShareConv}(\sigma, (\alpha_i, f_i, d_i)_{i\in[r]}, (\mathsf{sh}_{\sigma,i})_{i\in[m]}, (\mathsf{sh}_{\sigma,i}^{\mathsf{auth}})_{i\in[m]}, (c_i)_{i\in[m]}, (c_i^{\mathsf{auth}})_{i\in[m]}, (c_i^{\mathsf{ext\text{-}auth}})_{i\in[m]}, \mathsf{crs})$
11. *Output* (sk, y_0, y_1)

We refer to appendix B.1 for the proof of core lemma 1.

4.3 Homomorphic Share Conversion: RMS Programs with Leading Multiplication

Definition 7 (Restricted-Multiplication Straight-line Programs, adapted from [BGI16]). A (single-output) *restricted-multiplication straight-line (RMS) program* is an arbitrary sequence of instructions, each matching one of the following four:

- *Loading an input into memory:* $(\text{“Load”}, \hat{y}_j, \hat{w}_i)$ // Instructing to load input $\hat{w}_i$ into memory value/variable $\hat{y}_j$
- *Add values in memory:* $(\text{“Add”}, \hat{y}_k, \hat{y}_i, \hat{y}_j)$ // Instructing to store the sum of memory values/variables $\hat{y}_i$ and $\hat{y}_j$ in memory value/variable $\hat{y}_k$
- *Multiply value in memory by an input value:* $(\text{“Multiply”}, \hat{y}_k, \hat{w}_i, \hat{y}_j)$ // Instructing to store the product of input $\hat{w}_i$ and memory value/variable $\hat{y}_j$ in memory value/variable $\hat{y}_k$
- *Output value from memory:* $(\text{Output}, j, \hat{y}_i)$ // Instructing to output the memory value $\hat{y}_i$

We say a set of inputs is admissible to an RMS program with respect to bound B if all memory values remain bounded by B.

Algorithm Homomorphic Share Conversion: RMS Programs with Leading Multiplication, adapted from [CMPR23]

// Given $\langle X \rangle$, $\langle \mathsf{sk} \cdot X \rangle$, $\mathsf{Enc}_{\mathsf{sk}}(y_i)$, and $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot y_i)$, compute $\langle X \cdot \mathsf{RMS}(y_1, \ldots, y_1) \rangle$.
`Requires`: Mult is the algorithm of fig. 4. The scheme is additionally parameterised by parameters N, ζ.

$\mathsf{ShareConvExtRMS}(\sigma,$ // Party index
$\mathsf{pk},$ // Pk for the enc. underlying DDLog
$\Pi,$ // An RMS program
$\mathsf{sh}_\sigma,$ // $\langle X \rangle_\sigma$
$\mathsf{sh}^{\mathsf{auth}}_\sigma,$ // $\langle \mathsf{sk} \cdot X \rangle_\sigma$
$(c_i)_{i\in[m]},$ // $\mathsf{Enc}_{\mathsf{sk}}(y_i)$ for $i \in [m]$
$(c^{\mathsf{auth}}_i)_{i\in[m]},$ // $\mathsf{Enc}_{\mathsf{sk}}(\mathsf{sk} \cdot y_i)$ for $i \in [m]$
$\mathsf{crs})$ // A $(M \cdot \zeta \cdot \log N)$-bit string

1. Parse crs as $(\mathsf{crs}_j)_{j\in[M]} \in (\mathbb{Z}/N^\zeta\mathbb{Z})^M$
2. Parse Π as a sequence of RMS instructions $\mathsf{op}_1, \ldots, \mathsf{op}_M$.
3. Initialise a vector $\mathsf{MemShares}_\sigma$ indexed by $[M]$.
   // Used to store in position ℓ the memory value resulting from op_ℓ.
4. Initialise a vector $\mathsf{MemShares}^{\mathsf{auth}}_\sigma$ indexed by $[M]$.
   // Used to store in position ℓ the authenticated memory value resulting from op_ℓ.
5. For $\ell = 1, \ldots, M$:
   - If op_ℓ is of the form ("Load", $\hat{y}_j, \hat{w}_i$):

$$\mathsf{MemShares}_\sigma[j] \leftarrow \mathsf{Mult}(\mathsf{sh}_\sigma, \mathsf{sh}^{\mathsf{auth}}_\sigma, c_i, \mathsf{crs}_\ell)$$
$$\mathsf{MemShares}^{\mathsf{auth}}_\sigma[j] \leftarrow \mathsf{Mult}(\mathsf{sh}_\sigma, \mathsf{sh}^{\mathsf{auth}}_\sigma, c^{\mathsf{auth}}_i, \mathsf{crs}_\ell)$$

   - If op_ℓ is of the form ("Add", $\hat{y}_q, \hat{y}_i, \hat{y}_j$):

$$\mathsf{MemShares}_\sigma[q] \leftarrow \mathsf{MemShares}_\sigma[i] + \mathsf{MemShares}_\sigma[j]$$
$$\mathsf{MemShares}^{\mathsf{auth}}_\sigma[q] \leftarrow \mathsf{MemShares}^{\mathsf{auth}}_\sigma[i] + \mathsf{MemShares}^{\mathsf{auth}}_\sigma[j]$$

   - If op_ℓ is of the form ("Multiply", $\hat{y}_q, \hat{w}_i, \hat{y}_j$):

$$\mathsf{MemShares}_\sigma[q] \leftarrow \mathsf{Mult}(\mathsf{MemShares}_\sigma[j], \mathsf{MemShares}^{\mathsf{auth}}_\sigma[j], c_i, \mathsf{crs}_\ell)$$
$$\mathsf{MemShares}^{\mathsf{auth}}_\sigma[q] \leftarrow \mathsf{Mult}(\mathsf{MemShares}_\sigma[j], \mathsf{MemShares}^{\mathsf{auth}}_\sigma[j], c^{\mathsf{auth}}_i, \mathsf{crs}_\ell)$$

$$\mathsf{MemShares}_\sigma[q] \leftarrow \mathsf{DDLog}_{\mathsf{pk}}\big((c_{\mathsf{PRF},i}.\mathsf{fst})^{\mathsf{MemShares}^{\mathsf{auth}}_\sigma[j]} \cdot (c_{\mathsf{PRF},i}.\mathsf{snd})^{\mathsf{MemShares}_\sigma[j]}\big)$$
$$\mathsf{MemShares}^{\mathsf{auth}}_\sigma[q] \leftarrow \mathsf{DDLog}_{\mathsf{pk}}\big((c^{\mathsf{auth}}_i.\mathsf{fst})^{\mathsf{MemShares}^{\mathsf{auth}}_0[j]} \cdot (c^{\mathsf{auth}}_i.\mathsf{snd})^{\mathsf{MemShares}_\sigma[j]}\big)$$

   - If op_ℓ is of the form (Output, $\hat{y}_i$)

$$y_0 \leftarrow \mathsf{MemShares}_\sigma[i]$$

6. Output y_0

Fig. 6: Share conversion procedure for RMS programs with leading term.

Lemma 8 (Correctness of share conversion for RMS with leading multiplication, adapted from [CMPR23]). *Let $\lambda \in \mathbb{N}^*$ be a security parameter, let $\beta, B = \lambda^{\mathcal{O}(1)}$ be bounds, let Π be a B-bounded size-M RMS program on n inputs, and consider the deterministic algorithm* ShareConvExtRMS *of fig. 6 parameterised by Damgård-Jurik parameter ζ. For every*

$X \in [0, \beta)$ *and all admissible inputs* $x_1, \ldots, x_n \in [0, B)$,

$$\Pr\left[y_1 - y_0 = X \cdot \Pi(x_1, \ldots, x_n) \colon (y_0, y_1) \xleftarrow{\$} \mathsf{Sam}(1^\lambda, (x_i)_{i=1}^n)\right] \geq 1 - \frac{2M\beta B}{N^{\zeta-1}} \ ,$$

where the sampler Sam *is defined as:*

$\mathsf{Sam}(1^\lambda, (x_i)_{i\in[n]})$:

1. $((k, N), \mathsf{pk}) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)$
2. $\mathsf{sh}_0 \xleftarrow{\$} [0, 2^\lambda B)$
3. $\mathsf{sh}_1 \leftarrow X + \mathsf{sh}_{0,i}$
4. $\mathsf{sh}_0^{\mathsf{auth}} \xleftarrow{\$} [0, N \cdot 2^\lambda B)$
5. $\mathsf{sh}_1^{\mathsf{auth}} \leftarrow k \cdot X + \mathsf{sh}_{0,i}$
6. $c_i \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(x_i)$
7. $c_i^{\mathsf{auth}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sk} \cdot x_i)$
8. $\mathsf{crs} \xleftarrow{\$} (\mathbb{Z}/N^\zeta\mathbb{Z})^M$
9. *For* $\sigma \in \{0, 1\}$,
   $y_\sigma \leftarrow \mathsf{ShareConvExtRMS}(\sigma, \mathsf{pk}, \Pi, \mathsf{sh}_\sigma, \mathsf{sh}_\sigma^{\mathsf{auth}}, (c_i)_{i\in[n]}, (c_i^{\mathsf{auth}})_{i\in[n]}, \mathsf{crs})$
10. *Output* (y_0, y_1)

Lemma 8 follows from [CMPR23] (without using the formalism of staged HSS).

5 Privately Constrained PRF for Bounded Dual Waring Rank

We are finally ready to present our construction of a privately constrained PRF for constraints with polynomially bounded degree and Waring rank. As the main intuition behind the constructions and its security were already provided in the introduction, we only provide the formal protocol description and formal theorem statement here.

PCPRF Bounded Dual Waring-rank constrained PRF

`Requires:`

- r and d are fixed bounds on the Waring rank and the degree of the dual constraints
- DJE is the cryptosystem of fig. 1
- $\mathsf{ShareConv}$ is the algorithm of fig. 5
- $\mathcal{C} \subseteq \{0,1\}^n \to \{0,1\}$ is a class of constraints whose dual have Waring rank at most r and degree at most d; namely $\mathsf{Encode} \colon \mathcal{C} \to \{0,1\}^m$ is an encoding function, $\mathcal{C}^\perp \coloneqq \{C_x \colon \{0,1\}^m \to \{0,1\}, \widehat{C} \mapsto C(x)\}$ is the dual class of $\mathcal{C}$, and $\mathsf{Decompose}$ is a deterministic function mapping each $C_x \in \mathcal{C}^\perp$ to one of its Waring-rank-r decompositions $(\alpha_i, d_i, f_i)_{i\in[r]}$.
- $\mathsf{PRF} \colon \{0,1\}^n \times \{0,1\}^\lambda \to [0, 2^\mu)$ is a PRF which can be expressed RMS program of polynomial size M.
- $\mathsf{ShareConvExtRMS}$ is the algorithm of fig. 6.

$\mathsf{CPRF.KeyGen}(1^\lambda)$:

1. $(\mathsf{sk} = (k, N), \mathsf{pk} = (g, h, N)) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)$
2. $k_{\mathsf{PRF}} = (k_{\mathsf{PRF},1}, \ldots, k_{\mathsf{PRF},\lambda}) \xleftarrow{\$} \{0,1\}^\lambda$
3. $c_{\mathsf{cst}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(1)$
4. $c_{\mathsf{cst}}^{\mathsf{auth}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(k))$
5. For $i \in [m]$:
   - $\mathsf{sh}_{0,i} \xleftarrow{\$} [0, 2^\lambda)$ // $\langle \widehat{C}_i \rangle_0$ (λ-bit share over $\mathbb{Z}$)
   - $\mathsf{sh}_{0,i}^{\mathsf{auth}} \xleftarrow{\$} [0, N2^\lambda)$ // $\langle k \cdot \widehat{C}_i \rangle_0$ ($(\lambda + \log N)$-bit share over $\mathbb{Z}$)
   - $c_i \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sh}_{0,i})$ // $\mathsf{Enc}_{\mathsf{pk}}(\langle \widehat{C}_i \rangle_0)$
   - $c_i^{\mathsf{auth}} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sh}_{0,i}^{\mathsf{auth}})$ // $\mathsf{Enc}_{\mathsf{pk}}(\langle k \cdot \widehat{C}_i \rangle_0)$

– $c_i^{\mathsf{ext\text{-}auth}} \xleftarrow{\$} \mathsf{DJE.Enc_{pk}}(k \cdot \mathsf{sh}_{0,i})$ // $\mathsf{Enc_{pk}}(k \cdot \langle \widehat{C}_i \rangle_0)$

6. For $i \in [\lambda]$:
   – $c_{\mathsf{PRF},i} \xleftarrow{\$} \mathsf{DJE.Enc_{pk}}(k_{\mathsf{PRF},i})$
   – $c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}} \xleftarrow{\$} \mathsf{DJE.Enc_{pk}}(k \cdot k_{\mathsf{PRF},i})$
7. $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2) \xleftarrow{\$} (\mathbb{Z}/N^{\zeta}\mathbb{Z})^{4 \cdot rd} \times (\mathbb{Z}/N^{\zeta}\mathbb{Z})^{M}$ // A uniformly random string.
8. Output $\mathsf{msk} \leftarrow (\ \mathsf{sk}, k_{\mathsf{PRF}}, c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}},$
   $(\mathsf{sh}_{0,i}, \mathsf{sh}_{0,i}^{\mathsf{auth}}, c_i, c_i^{\mathsf{auth}}, c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]},$
   $(c_{\mathsf{PRF},i}, c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}})_{i \in [\lambda]},$
   $\mathsf{crs})$
9. Output msk

$\mathsf{CPRF.Eval}(1^{\lambda}, \mathsf{msk}, x)$:

1. Parse $\mathsf{sk} \leftarrow (k, \mathsf{pk})$
2. $(\alpha_i, d_i, f_i)_{i \in [r]} \leftarrow \mathsf{Decompose}(C_x)$
3. Parse $\mathsf{msk} = (\ \mathsf{sk}, k_{\mathsf{PRF}}, c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}},$
   $(\mathsf{sh}_{0,i}, \mathsf{sh}_{0,i}^{\mathsf{auth}}, c_i, c_i^{\mathsf{auth}}, c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]},$
   $(c_{\mathsf{PRF},i}, c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}})_{i \in [\lambda]},$
   $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2))$
4. $(\overline{y}_0, \overline{y}_0^{\mathsf{auth}}) \leftarrow \mathsf{ShareConv}(\ 0, k,$
   $(\alpha_i, f_i, d_i)_{i \in [r]},$
   $(\mathsf{sh}_{0,i})_{i \in [m]},$
   $(\mathsf{sh}_{0,i}^{\mathsf{auth}})_{i \in [m]},$
   $(c_i)_{i \in [m]},$
   $(c_i^{\mathsf{auth}})_{i \in [m]},$
   $(c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]},$
   $(c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}}),$
   $\mathsf{crs}_1)$
5. $y_0 \leftarrow \mathsf{ShareConvExtRMS}(0, N, \mathsf{PRF}, \overline{y}_0, \overline{y}_0^{\mathsf{auth}}, (c_i)_{i \in [m]}, (c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]}, \mathsf{crs}_2)$
6. Output $y_0 \bmod 2^{\mu}$

$\mathsf{CPRF.Constrain}(1^{\lambda}, \mathsf{msk}, C)$:

1. $\widehat{C} = (\widehat{C}_1, \ldots, \widehat{C}_m) \leftarrow \mathsf{Encode}(C)$
2. Parse $\mathsf{msk} = (\ \mathsf{sk}, k_{\mathsf{PRF}}, c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}},$
   $(\mathsf{sh}_{0,i}, \mathsf{sh}_{0,i}^{\mathsf{auth}}, c_i, c_i^{\mathsf{auth}}, c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]},$
   $(c_{\mathsf{PRF},i}, c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}})_{i \in [\lambda]},$
   $\mathsf{crs})$
3. For $i \in [m]$,
   – $\mathsf{sh}_{1,i} \leftarrow \widehat{C}_i + \mathsf{sh}_{0,i}$
   – $\mathsf{sh}_{1,i}^{\mathsf{auth}} \leftarrow \mathsf{sk} \cdot \widehat{C}_i + \mathsf{sh}_{0,i}^{\mathsf{auth}}$
4. $\mathsf{ck} \leftarrow (\ \mathsf{pk}, c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}},$
   $(\mathsf{sh}_{1,i}, \mathsf{sh}_{1,i}^{\mathsf{auth}}, c_i, c_i^{\mathsf{auth}}, c_i^{\mathsf{ext\text{-}auth}})_{i \in [m]},$
   $(c_{\mathsf{PRF},i}, c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}})_{i \in [\lambda]},$
   $\mathsf{crs})$
5. Output ck

$\mathsf{CPRF.CEval}(1^{\lambda}, \mathsf{ck}, x)$:

1. Parse $\mathsf{ck} = (\ \mathsf{pk}, c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}},$
$(\mathsf{sh}_{1,i}, \mathsf{sh}_{1,i}^{\mathsf{auth}}, c_i, c_i^{\mathsf{auth}}, c_i^{\mathsf{ext\text{-}auth}})_{i\in[m]},$
$(c_{\mathsf{PRF},i}, c_{\mathsf{PRF},i}^{\mathsf{ext\text{-}auth}})_{i\in[\lambda]},$
$\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2))$
2. $(\overline{y}_1, \overline{y}_1^{\mathsf{auth}}) \leftarrow \mathsf{ShareConv}(\ 1, \bot,$
$(\alpha_i, f_i, d_i)_{i\in[r]},$
$(\mathsf{sh}_{1,i})_{i\in[m]},$
$(\mathsf{sh}_{1,i}^{\mathsf{auth}})_{i\in[m]},$
$(c_i)_{i\in[m]},$
$(c_i^{\mathsf{auth}})_{i\in[m]},$
$(c_i^{\mathsf{ext\text{-}auth}})_{i\in[m]},$
$(c_{\mathsf{cst}}, c_{\mathsf{cst}}^{\mathsf{auth}}),$
$\mathsf{crs}_1)$
3. $y_1 \leftarrow \mathsf{ShareConvExtRMS}(1, N, \mathsf{PRF}, \overline{y}_1, \overline{y}_1^{\mathsf{auth}}, (c_i)_{i\in[m]}, (c_i^{\mathsf{ext\text{-}auth}})_{i\in[m]}, \mathsf{crs}_2)$
4. Output $y_1 \bmod 2^\mu$

Fig. 7: Privately Constrained PRF for any class of constraints whose dual have (polynomially) bounded Waring rank and degree.

Main Theorem 1 (Privately Constrained PRF for Bounded Dual Waring Rank from DCR)**.** *Let $\mathcal{C}$ be any class of constraints whose duals all have Waring rank and algebraic degree upper bounded by some polynomials in the security parameter. The construction of fig. 7 is a single-key, selectively secure privately constrained PRF for $\mathcal{C}$.*

We refer to appendix B.2 for the proof of main theorem 1.

Corollary 9 (Privately Punctured PRF from DCR). *Assuming DCR, there exists a single-key, selectively secure, privately punctured PRF.*

Proof. As already discussed in the technical overview, the puncturing constraint can be expressed with a polynomial of bounded degree and Waring rank. More formally, the class of puncturing constraints is[9]

$$\mathcal{C} \coloneqq \{C_{x^\star} \colon \{0,1\}^n \to \{0,1\}, x \mapsto x = x^\star \mid x^\star \in \{0,1\}^n\}$$

and it coincides with its own dual class

$$\mathcal{C}^\perp \coloneqq \{C_x \colon \{0,1\}^n \to \{0,1\}, x^\star \mapsto x = x^\star \mid x \in \{0,1\}^n\}$$

We show that the puncturing constraint C_x can be expressed as a polynomial of degree n in the bits of $x^\star$: We can define a function A_x that counts the number of common bits between $x, x^\star$ and rewrite the puncturing constraint as a the product of these terms, and then normalize it by the max value of this polynomial to ensure that $C_x(x^\star) = 1$. More

[9] Remember we want $C(x) = 0$ for authorized inputs, and $C(x) = 1$ for constrained inputs.

formally:

$$\begin{aligned} C_x(x^\star) &= \left[\sum_{i=1}^{n}(x_i = x_i^\star) = n\right] \\ &= \left[\prod_{i=0}^{n-1}(A_x(x^\star) - i) = 0\right] \qquad \text{where } A_x\colon \{0,1\}^n \to \mathbb{Z},\ (X_i)_{i\in[n]} \mapsto \sum_{i=1}^{n}(x_i = X_i) \\ &= \left[\sum_{i=0}^{n}\alpha_i \cdot A_x(x^\star)^i\right]/(n!) \\ &= \sum_{i=0}^{n}\frac{\alpha_i}{n!}\cdot A_x(x^\star)^i \end{aligned}$$

where the $(\alpha_i)_{i=0}^n$ are the coefficients[10] of the degree-n polynomial $\prod_{i=0}^{n-1}(X-i)$. In this form, the dual of puncturing clearly has Waring rank at most n. □

6 Waring Rank

In this section, we provide some background on Waring rank and prove some results that are useful for our construction. We will relax the homogeneity condition and define Waring rank for polynomials of degree *at most* d.

Definition 10 (Waring Rank and its variants). *Let $f \in \mathbb{Q}[X_1,\ldots,X_N]$ be a polynomial of degree $d \in \mathbb{N}$. In the following, let $f_1,\ldots,f_r,g_1,\ldots,g_r \in \mathbb{Q}[X_1,\ldots,X_N]$ denote affine functions[11] and $\alpha_1,\ldots,\alpha_r \in \mathbb{Q}$.*

- *The* Waring rank *of f is defined to be the smallest integer $r \in \mathbb{N}$ such that there exists affine functions $f_1,\ldots,f_r$ and coefficients $\alpha_1,\ldots,\alpha_r$ satisfying*

$$f(X_1,\ldots,X_N) \;=\; \sum_{i=1}^{r}\alpha_i\,(f_i(X_1,\ldots,X_N))^d.$$

  We will refer to the set $\{(f_1,\alpha_1),\ldots,(f_r,\alpha_r)\}$ as a Waring decomposition of f of rank r.
- *The* Mixed-degree Waring rank *of f is defined to be the smallest integer $r \in \mathbb{N}$ such that there exists affine functions $f_1,\ldots,f_r$ and coefficients $\alpha_1,\ldots,\alpha_r$ satisfying*

$$f(X_1,\ldots,X_N) \;=\; \sum_{i=1}^{r}\alpha_i\,(f_i(X_1,\ldots,X_N))^{d_i}, \qquad \text{where each } 0 \le d_i \le d.$$

  We will refer to the set $\{(f_1,d_1,\alpha_1),\ldots,(f_r,d_r,\alpha_r)\}$ as a Mixed Waring decomposition of f of rank r.
- *The* Split Waring Rank *of f is defined to be the smallest integer $r \in \mathbb{N}$ such that there exists affine functions $f_1,\ldots,f_r$ and $g_1,\ldots,g_r$ and coefficients $\alpha_1,\ldots,\alpha_r$ satisfying*

$$f(X_1,\ldots,X_N) \;=\; \sum_{i=1}^{r}\alpha_i\,(f_i(X_1,\ldots,X_{N/2}))^{d_i}\cdot(g_i(X_{N/2+1},\ldots,X_N))^{e_i},$$

  where for each $i \in [r]$, $d_i,e_i \ge 0$ and $(d_i+e_i) \le d$. We will refer to the set $\{(f_1,g_1,d_1,e_1,\alpha_1),\ldots,(f_r,g_r,d_r,e_r,\alpha_r)\}$ as a Split Waring decomposition of f of rank r.

[10] Note that these coefficients can be computed in polynomial time.

[11] We allow affine (instead of linear) functions here because we are concerned with representing inhomogeneous polynomials 7.

An example. We give two simple examples of Waring decomposition for concreteness.

$$X_1X_2 = \frac{1}{4}(X_1+X_2)^2 - \frac{1}{4}(X_1-X_2)^2$$

$$\begin{aligned} X_1X_2X_3 = \frac{1}{24}(X_1+X_2+X_3)^3 - \frac{1}{24}(X_1+X_2-X_3)^3 \\ -\frac{1}{24}(X_1-X_2+X_3)^3 - \frac{1}{24}(-X_1+X_2+X_3)^3 \end{aligned}$$

At first sight, it is unclear whether the Waring rank of every degree $\leq d$ polynomial is finite. However, extending the above examples, it is not difficult to show that every monomial of degree $\leq d$ has a finite Waring rank, and this shows that every degree $\leq d$ polynomial has a finite Waring rank (using the fact that monomials of degree $\leq d$ span the space of polynomials of degree $\leq d$). For a proof of this see e.g., [Tei14].

Relation Between Notions. We will first show that for a polynomial f of degree d, the Waring rank, Mixed Waring rank, and Split Waring rank are equivalent up to $\text{poly}(d)$ factors, i.e. if f has a low-rank Waring, Mixed Waring, Split Waring decomposition, then it has a low-rank Waring, Mixed Waring, and Split Waring decomposition (see Lemma 12). To show this, we will use the following lemma.

Lemma 11. *Let $\ell_1, \ell_2 \in \mathbb{Q}[X_1, \ldots, X_N]$ be affine functions. For any $d_1, d_2 \in \mathbb{N}$, the Waring rank of $\ell_1^{d_1} \cdot \ell_2^{d_2}$ is $\leq (d_1+d_2+1)$.*

Proof of Lemma 11. Consider the following univariate polynomial:

$$A(t) := (\ell_1(X_1, \ldots, X_N) + t \cdot \ell_2(X_1, \ldots, X_N))^{d_1+d_2}$$

Note that coefficient of t^{d_2} in the univariate polynomial $A(t)$ is exactly $\binom{d_1+d_2}{d_1}\ell_1^{d_1}\ell_2^{d_2}$. Using interpolation on $A(t)$, we can express each coefficient of $A(t)$ as a linear combination of evaluations of $A(t)$. More precisely, there exists coefficients $c_0, \ldots, c_{d_1+d_2}$ such that the following holds:

$$\text{coefficient of } t^{d_2} \text{ in } A(t) = \sum_{j=0}^{d_1+d_2} c_j A(j) = \sum_{j=0}^{d_1+d_2} c_j(\ell_1 + j\ell_2)^{d_1+d_2}.$$

So we have expressed the coefficient of t^{d_2} as a sum of the $(d_1+d_2)^{th}$ power of (d_1+d_2+1) affine functions. As noted before, the coefficient of t^{d_2} in $A(t)$ is a non-zero scalar multiple of $(\ell_1(X_1, \ldots, X_N))^{d_1} \cdot (\ell_2(X_1, \ldots, X_N))^{d_2}$. This completes the proof of Lemma 11. □

Now we are ready to show that the ranks are equivalent up to $\text{poly}(d)$ factors.

Lemma 12. *Let $f \in \mathbb{Q}[X_1, \ldots, X_N]$ be a polynomial of degree $\leq d$. If any one of the Waring rank/Mixed Waring rank/Split Waring rank of f is $\leq r$, then the remaining two are at most $\mathcal{O}(rd^2)$. In simple words, all three ranks are equivalent up to a multiplicative factor of* $\text{poly}(d)$.

Proof of Lemma 12. We start by observing that if $f(X_1, \ldots, X_N)$ has Waring rank r, then it has Mixed Waring rank $\leq r$ (every Waring decomposition is also a Mixed Waring decomposition). We now show that if f has Waring rank r, then it has Split Waring rank $\leq r(d+1)$. Suppose f has a Waring decomposition $\{(f_1, \alpha_1), \ldots, (f_r, \alpha_r)\}$ of rank r, then we "split" each

affine function f_i as a sum of two affine functions $f_{i,1}$ and $f_{i,2}$ on disjoint sets of variables, i.e.

$$f_i(X_1, \ldots, X_N) \;=\; f_{i,1}(X_1, \ldots, X_{N/2}) \;+\; f_{i,2}(X_{N/2+1}, \ldots, X_N).$$

Using Binomial Theorem on $f_i^d = (f_{i,1} + f_{i,2})^d$ for each $i \in [r]$, we have,

$$(f_i(X_1, \ldots, X_N))^d \;=\; \sum_{j=0}^{d} \binom{d}{j} (f_{i,1}(X_1, \ldots, X_{N/2}))^j \cdot (f_{i,2}(X_{N/2+1}, \ldots, X_N))^{d-j}$$

This gives us a Split Waring decomposition of f of rank $\leq r(d+1)$.

Suppose $f(X_1, \ldots, X_N)$ has Mixed Waring rank r. We will now argue that $f(X_1, \ldots, X_N)$ has Waring rank $\leq r(d+1)$. Suppose f has a Mixed Waring decomposition $\{(f_1, d_1, \alpha_1), \ldots, (f_r, d_r, \alpha_r)\}$ of rank r. Define the polynomial $A(X_1, \ldots, X_N, Z)$ as follows:

$$A(X_1, \ldots, X_N, Z) \;=\; \sum_{i=1}^{r} \alpha_i \, (f_i(X_1, \ldots, X_N))^{d_i} \cdot Z^{d-d_i}, \qquad \text{where each } \; 0 \leq d_i \leq d$$

Applying Lemma 11 on $f_i^{d_i} \cdot Z^{d-d_i}$ for each $i \in [r]$, we get that $A(X_1, \ldots, X_N, Z)$ has a Waring decomposition which has $r(d+1)$ affine functions in $\mathbb{Q}[X_1, \ldots, X_N, Z]$. Observe that $A(X_1, \ldots, X_N, 1) = f(X_1, \ldots, X_N)$. Setting $Z = 1$ shows that $f(X_1, \ldots, X_N)$ has Waring rank $\leq r(d+1)$.

Suppose $f(X_1, \ldots, X_N)$ has Split Waring rank r. We will now argue that $f(X_1, \ldots, X_N)$ has Waring rank $\leq r(d+1)^2$. Suppose f has a Split Waring decomposition $\{(f_1, g_1, d_1, e_1, \alpha_1), \ldots, (f_r, g_r, d_r, e_r, \alpha_r)\}$ of rank r. Applying Lemma 11 on $f_i^{d_i} g_i^{e_i}$ for each $i \in [r]$, we get that the Mixed Waring rank of $f(X_1, \ldots, X_N)$ is $\leq \sum_{i=1}^{r} (d_i + e_i + 1)$. Now using the discussion from the previous paragraph, we conclude that the Waring rank of $f(X_1, \ldots, X_N)$ is $\leq (d+1) \sum_{i=1}^{r} (d_i + e_i + 1)$, which is $\leq r(d+1)^2$.

To finish the proof, we need to argue that the Mixed Waring rank and the Split Waring rank are equivalent. If f has Mixed Waring rank $\leq r$, then it has Waring rank $\leq r(d+1)$, which implies it has Split Waring rank $\leq r(d+1)^2$. Similarly, if f has Split Waring rank $\leq r$, then it has Waring rank $\leq r(d+1)^2$, which implies it has Mixed Waring rank $\leq r(d+1)^2$. This finishes the proof of Lemma 12. □

The next lemma shows that if two polynomials have a small Waring rank, then their product also has a small Waring rank.

Lemma 13. *Let $f, g \in \mathbb{Q}[X_1, \ldots, X_N]$ be polynomials with Waring ranks (resp. mixed-degree Waring ranks) r_f and r_g. Then the polynomial fg has Waring rank (resp. mixed-degree Waring rank) $\leq r_f r_g (\deg(f) + \deg(g) + 1)$.*

Proof. Suppose f and g have Waring decompositions $\{(f_1, \alpha_1), \ldots, (f_{r_f}, \alpha_{r_f})\}$ and $\{(g_1, \alpha_1), \ldots, (g_{r_g}, \alpha_{r_g})\}$ of ranks r_f and r_g respectively. Then,

$$f(X_1, \ldots, X_N) \cdot g(X_1, \ldots, X_N) \;=\; \sum_{\substack{1 \leq i \leq r_f \\ 1 \leq j \leq r_g}} \alpha_i \beta_i \; (f_i(X_1, \ldots, X_N)^{\deg(f)} \;\cdot\; (g_i(X_1, \ldots, X_N))^{\deg(g)}.$$

Now applying Lemma 11 on $f_i^{\deg(f)} g_j^{\deg(g)}$ for each $1 \leq i \leq r_f$ and $1 \leq j \leq r_g$, we get that

the Waring rank of $f(X_1, \ldots, X_N) \cdot g(X_1, \ldots, X_N)$ is $\leq r_f r_g(\deg(f) + \deg(g) + 1)$.

An analogous argument shows that if f and g have Mixed Waring rank r_f and r_g, then $f(X_1, \ldots, X_N) \cdot g(X_1, \ldots, X_N)$ have Mixed Waring rank $\leq r_f r_g(\deg(f) + \deg(g) + 1)$. This finishes the proof of Lemma 13. □

Theorem 14 (Duality). *For every $d, r \in \mathbb{N}$, there exists an encoding $\rho : \mathbb{Q}[X_1, \ldots, X_N] \to \mathbb{Q}^M$ of all polynomials $f \in \mathbb{Q}[X_1, \ldots, X_N]$ with $\deg(f) \leq d$ and Waring rank $\leq r$, where $M = (r+1)N$.*
Additionally, for every $(x_1, \ldots, x_N) \in \mathbb{Q}^N$, there exists a polynomial $g_{x_1,\ldots,x_N} \in \mathbb{Q}[Y_1, \ldots, Y_M]$ of degree $\leq d$ such that $g_{x_1,\ldots,x_N}(\rho(f)) = f(x_1, \ldots, x_N)$, and Waring rank of $g_{x_1,\ldots,x_N}$ is $\leq r$.

Proof. We first describe an encoding ρ. Suppose f is a degree $\leq d$ polynomial with Waring rank $\leq r$, with the following Waring decomposition[12]:

$$f(X_1, \ldots, X_N) \;=\; \sum_{i=1}^{r} \alpha_i \, (f_i(X_1, \ldots, X_N))^d, \tag{2}$$

where for every $i \in [r]$, the affine function $f_i \in \mathbb{Q}[X_1, \ldots, X_N]$ is

$$f_i(X_1, \ldots, X_N) \;=\; c_{i,0} + c_{i,1}X_1 + \ldots + c_{i,N}X_N. \tag{3}$$

The map ρ maps $f(X_1, \ldots, X_N)$ to $\mathbb{Q}^M$ as follows:

$$\rho(f(X_1, \ldots, X_N)) \;=\; (c_{1,0}, \ldots, c_{1,N}, \ldots, c_{r,0}, \ldots, c_{r,N}).$$

In simple words, ρ simply maps f to a M-dimensional vector which has all the coefficients of every affine function f_i in a (fixed) Waring decomposition of f of rank r.

Next we describe the polynomial $g_{x_1,\ldots,x_N}$. Fix an arbitrary $(x_1, \ldots, x_N) \in \mathbb{Q}^N$. For each $i \in [r]$, define a new polynomial $\tilde{f}_i \in \mathbb{Q}[X_1, \ldots, X_N][Y_1, \ldots, Y_M]$ (i.e. $\tilde{f}_i$ is a polynomial in Y-variables with coefficients from the ring $\mathbb{Q}[X_1, \ldots, X_N]$) as follows: We replace $c_{i,j}$ with the variable $Y_{(i-1)(N+1)+(j+1)}$. In other words, for each $i \in [r]$,

$$\tilde{f}_i(X_1, \ldots, X_N, Y_1, \ldots, Y_M) \;=\; Y_{(i-1)N+1} + Y_{(i-1)N+2}X_1 + \ldots + Y_{(i-1)N+(N+1)}X_N.$$

Similarly, using Equation (2), let $\tilde{f} := \sum_{i=1}^{r} \alpha_i \tilde{f}_i$. Observe that $\tilde{f}_i$'s have degree 1 in the Y variables, and $\tilde{f}$ is a polynomial of degree $\leq d$ in the Y variables.

Define the polynomial $g_{x_1,\ldots,x_N} \in \mathbb{Q}[Y_1, \ldots, Y_M]$ as follows:

$$g_{x_1,\ldots,x_N}(Y_1, \ldots, Y_M) \;:=\; \tilde{f}(x_1, \ldots, x_N, Y_1, \ldots, Y_M).$$

By definition, $\tilde{f}(x_1, \ldots, x_N, \rho(f)) = f(x_1, \ldots, x_N)$, which implies that $g_{x_1,\ldots,x_N}(\rho(f)) = f(x_1, \ldots, x_N)$. Since the degree of $\tilde{f}$ in Y-variables is $\leq d$, this implies that the degree of $g_{x_1,\ldots,x_N}$ is $\leq d$. Now it remains to show that the Waring rank of $g_{x_1,\ldots,x_N}$ is $\leq r$. Using Equation (2), it is easy to verify:

$$g_{x_1,\ldots,x_N}(Y_1, \ldots, Y_M) \;=\; \sum_{i=1}^{r} \alpha_i \, \tilde{f}_i(x_1, \ldots, x_N, Y_1, \ldots, Y_M).$$

Note that for each $i \in [r]$, the polynomial $\tilde{f}_i(x_1, \ldots, x_N, Y_1, \ldots, Y_M) \in \mathbb{Q}[Y_1, \ldots, Y_M]$ is an affine function. In other words, we have given a Waring decomposition of $\tilde{f}$ of rank r. This finishes the proof of Theorem 14. □

[12] If there are multiple Waring decompositions of f, we choose one decomposition arbitrarily.

Acknowledgments

We thank Geoffroy Couteau for insightful discussions.
This research was supported by: the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement number 803096 (SPEC); the Danish Independent Research Council under Grant-IDs DFF-3103-00077B (CryptoDigi) and DFF-0165-00107B (C3PO); and the DARPA SIEVE program (contract HR001120C0085 "FROMAGER"). Amik Raj Behera is supported by Srikanth Srinivasan's start-up grant from the University of Copenhagen. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA. Distribution Statement "A" (Approved for Public Release, Distribution Unlimited).

References

ADOS22. Damiano Abram, Ivan Damgård, Claudio Orlandi, and Peter Scholl. An algebraic framework for silent preprocessing with trustless setup and active security. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 421–452. Springer, Cham, August 2022.

AMN+18. Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Constrained PRFs for NC^1 in traditional groups. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 543–574. Springer, Cham, August 2018.

BCM+24. Dung Bui, Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Fast public-key silent OT and more from constrained Naor-Reingold. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 88–118. Springer, Cham, May 2024.

BGI14. Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Berlin, Heidelberg, March 2014.

BGI16. Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 509–539. Springer, Berlin, Heidelberg, August 2016.

BGIK22. Elette Boyle, Niv Gilboa, Yuval Ishai, and Victor I. Kolobov. Programmable distributed point functions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 121–151. Springer, Cham, August 2022.

BLW17. Dan Boneh, Kevin Lewi, and David J. Wu. Constraining pseudorandom functions privately. In Serge Fehr, editor, *PKC 2017, Part II*, volume 10175 of *LNCS*, pages 494–524. Springer, Berlin, Heidelberg, March 2017.

BTVW17. Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained PRFs (and more) from LWE. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 264–302. Springer, Cham, November 2017.

BW13. Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Berlin, Heidelberg, December 2013.

CC17. Ran Canetti and Yilei Chen. Constraint-hiding constrained PRFs for NC^1 from LWE. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 446–476. Springer, Cham, April / May 2017.

CHI+18. Luca Chiantini, Jonathan D. Hauenstein, Christian Ikenmeyer, Joseph M. Landsberg, and Giorgio Ottaviani. Polynomials and the exponent of matrix multiplication. *Bulletin of the London Mathematical Society*, 50(3):369–389, 2018.

Cle90. Richard Cleve. Towards optimal simulations of formulas by bounded-width programs. In *22nd ACM STOC*, pages 271–277. ACM Press, May 1990.

CMPR23. Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Constrained pseudorandom functions from homomorphic secret sharing. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 194–224. Springer, Cham, April 2023.

DGI+24. Pranjal Dutta, Fulvio Gesmundo, Christian Ikenmeyer, Gorav Jindal, and Vladimir Lysikov. Fixed-parameter debordering of waring rank. In Olaf Beyersdorff, Mamadou Moustapha Kanté,

Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, March 12-14, 2024, Clermont-Ferrand, France*, volume 289 of *LIPIcs*, pages 30:1–30:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

DJ01. Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 119–136. Springer, Berlin, Heidelberg, February 2001.

DKN+20. Alex Davidson, Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Adaptively secure constrained pseudorandom functions in the standard model. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 559–589. Springer, Cham, August 2020.

EGdOW18. Klim Efremenko, Ankit Garg, Rafael Mendes de Oliveira, and Avi Wigderson. Barriers for rank methods in arithmetic complexity. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 1:1–1:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

GGM84. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

HKKW19. Dennis Hofheinz, Akshay Kamath, Venkata Koppula, and Brent Waters. Adaptively secure constrained pseudorandom functions. In Ian Goldberg and Tyler Moore, editors, *FC 2019*, volume 11598 of *LNCS*, pages 357–376. Springer, Cham, February 2019.

IK99. Anthony Iarrobino and Vassil Kanev. *Power Sums, Gorenstein Algebras, and Determinantal Loci*. Cambridge Studies in Advanced Mathematics. Springer Berlin, Heidelberg, 1999.

ILL24. Yuval Ishai, Hanjun Li, and Huijia Lin. Succinct partial garbling from groups and applications. Cryptology ePrint Archive, Paper 2024/2073, 2024.

KPTZ13. Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.

Lan17. J. M. Landsberg. *Geometry and Complexity Theory*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2017.

MORS24. Pierre Meyer, Claudio Orlandi, Lawrence Roy, and Peter Scholl. Rate-1 arithmetic garbling from homomorphic secret sharing. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part IV*, volume 15367 of *LNCS*, pages 71–97. Springer, Cham, December 2024.

OSY21. Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 678–708. Springer, Cham, October 2021.

Pai99. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Berlin, Heidelberg, May 1999.

Pra19. Kevin Pratt. Waring rank, parameterized and exact algorithms. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 806–823. IEEE Computer Society, 2019.

PS18. Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 675–701. Springer, Cham, March 2018.

RS21. Lawrence Roy and Jaspal Singh. Large message homomorphic secret sharing from DCR and applications. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part III*, volume 12827 of *LNCS*, pages 687–717, Virtual Event, August 2021. Springer, Cham.

Ser24. Sacha Servan-Schreiber. Constrained pseudorandom functions for inner-product predicates from weaker assumptions. Cryptology ePrint Archive, Paper 2024/058, 2024.

SW14. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

Tei14. Zach Teitler. Ranks of polynomials. `https://www.theoremoftheday.org/Docs/Teitler2014.pdf`, 2014.

Supplementary Material

A More Preliminaries

A.1 Damgård-Jurik-ElGamal encryption scheme

Definition 15 (Decision Composite Residuosity Assumption (DCR), [Pai99]). *Let* $\mathsf{RSA.Gen}$ *be a polynomial-time algorithm which, on input a security parameter* λ*, outputs* (N, p, q) *where* p *and* q *are* λ*-bit primes and* $N = pq$*. Let* λ *be a security parameter. We say that the* Decision Composite Residuosity (DCR) *problem is hard relative to modulus-sampling algorithm* $\mathsf{RSA.Gen}$ *if*

$$\left\{(N,x)\colon \begin{array}{r}(N,p,q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda)\\ x \xleftarrow{\$} (\mathbb{Z}/N^2\mathbb{Z})^\times\end{array}\right\} \overset{c}{\approx} \left\{(N, x^N \bmod N^2)\colon \begin{array}{r}(N,p,q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda)\\ x \xleftarrow{\$} (\mathbb{Z}/N^2\mathbb{Z})^\times\end{array}\right\}.$$

Theorem 16. *Assuming DCR, Damgård–Jurik–ElGamal encryption fig. 8 is a public key encryption scheme satisfying correctness and KDM security for affine functions of the key. Specifically, the following properties hold:*

Correctness: $\mathsf{DJE.Dec}_{\mathsf{sk}}(\mathsf{DJE.Enc}_{\mathsf{pk}}(x)) = x$*, for any* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{DJE.KeyGen}$*, and any* $x \in \mathbb{Z}/N^\zeta\mathbb{Z}$*.*

KDM Security: *For all* p.p.t. *adversaries* Adv*, the oracles* $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}$ *and* $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}$ *are indistinguishable in the following experiment*

$$\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)\\ \text{output } \mathsf{Adv}^{\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R/\$}(\mathsf{pk})}\end{array}$$

where these oracles are defined as

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}(x,y)$:
$(k, N) \leftarrow \mathsf{sk}$
$z \leftarrow x \cdot k + y$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}(x,y)$:
$(k, N) \leftarrow \mathsf{sk}$
$z \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

Proof. This is very similar to many previous encryption schemes used in HSS and proven KDM secure in that context. See, e.g., the Damgård–Jurik instantiation of the NIDLS framework [ADOS22]), though details differ, as we do not require p and q to be safe primes. For completeness, we present a full proof in appendix A.2. □

DJE Damgård-Jurik-ElGamal Cryptosystem

`Requires:`

- $\zeta \geq 1$ is a parameter defining the plaintext size.
- Group isomorphism $\exp\colon (\mathbb{Z}/N^\zeta\mathbb{Z})^+ \to 1 + N(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})$ and its inverse $\log\colon 1 + N(\mathbb{Z}/N^{\zeta+1}\mathbb{Z}) \to (\mathbb{Z}/N^\zeta\mathbb{Z})^+$, as defined as in [RS21]:

$$\exp(x) = \sum_{k=0}^{\zeta} \frac{(Nx)^k}{k!} \qquad \text{and} \qquad \log(1+Nx) = \sum_{k=1}^{\zeta} \frac{(-N)^{k-1}x^k}{k}$$

$\mathsf{DJE.KeyGen}(1^\lambda)$:

1. Sample $N \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda)$
2. Sample $k \xleftarrow{\$} [0, N)$
3. Sample $g \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$
4. Compute $h \leftarrow g^{-k}$
5. Output $(\mathsf{sk} = (k, N), \mathsf{pk} = (g, h, N))$

$\mathsf{DJE.Enc}_{\mathsf{pk}}(x)$:

1. Parse $\mathsf{pk} = (g, h, N)$
2. Sample $r \xleftarrow{\$} [0, N)$
3. Compute $c_0 \leftarrow g^r$
4. Compute $c_1 \leftarrow h^r \cdot \exp(x)$
5. Output $c = (c_0, c_1)$

$\mathsf{DJE.Dec}_{\mathsf{sk}}(c = (c_0, c_1))$:

1. Parse $\mathsf{sk} = (k, N)$
2. Assert $c_0^k \cdot c_1 \equiv 1 \mod N$
3. Output $x \leftarrow \log(c_0^k \cdot c_1)$

Fig. 8: The Damgård-Jurik-ElGamal cryptosystem.

A.2 Damgård-Jurik-ElGamal Cryptosystem Proofs

Correctness is straightfoward from the definitions:

$$\begin{aligned}\mathsf{DJE.Dec}_{\mathsf{sk}}(\mathsf{DJE.Enc}_{\mathsf{pk}}(x)) &= \mathsf{DJE.Dec}_{\mathsf{sk}}(g^r, (g^{-k})^r \cdot \exp(x)) \\ &= \log(g^{kr} \cdot (g^{-k})^r \cdot \exp(x)) \\ &= \log(\exp(x)) = x\end{aligned}$$

A.3 DCR over $(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$

Before proving security, we will need a lemma to extend the DCR assumption to the group $(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$ that we are actually working over.

Lemma 17. *Assuming DCR, for any plaintext size parameter $\zeta \geq 1$ (which in general can be any polynomially-bounded PPT function of N), we have*

$$\left\{(N, x) \colon \begin{matrix}(N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda) \\ x \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times\end{matrix}\right\} \overset{\mathrm{c}}{\approx} \left\{(N, x^{N^\zeta} \bmod N^{\zeta+1}) \colon \begin{matrix}(N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda) \\ x \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times\end{matrix}\right\}.$$

Proof. The following is an adaptation of the proof of security for Damgård–Jurik encryption [DJ01]. Define hybrid distributions $\mathcal{H}_i$ for $i \in \{0, \ldots, \zeta\}$ as follows:

$$\mathcal{H}_i = \left\{(N, x^{N^i}) \colon \begin{matrix}(N, p, q) \xleftarrow{\$} \mathsf{RSA.Gen}(1^\lambda) \\ x \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times\end{matrix}\right\}$$

Clearly, $\mathcal{H}_0$ and $\mathcal{H}_\zeta$ are our original left and right distributions, respectively.

To complete the proof, we must show that $\mathcal{H}_i \overset{\mathrm{c}}{\approx} \mathcal{H}_{i+1}$ for all i. First, write $x = y \cdot \exp(Nr)$ for random $y \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$ and $r \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$. Notice that x only depends on y through $y \bmod N^2$, because $\exp(Nr)$ randomises the coefficients of N^2 and all higher powers in the N-adic expansion of x. (That is, $\exp(Nr)$ outputs a uniform element of $1 + N^2(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})$.) Therefore, by DCR it is indistinguishable to set $y = y'^N$, where $y' \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$. Finally, let $x' = y' \exp(r)$. We now have

$$x^{N^i} = (y \exp(Nr))^{N^i} = (y'^N \exp(Nr))^{N^i} = (y' \exp(r))^{N^{i+1}} = x'^{N^{i+1}},$$

where the distribution of x' is uniform on $(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$. □

A.4 KDM Security

Recall that we want to show that the following oracles

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}(x,y)$:
$(k,N) \leftarrow \mathsf{sk}$
$z \leftarrow x \cdot k + y$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}(x,y)$:
$(k,N) \leftarrow \mathsf{sk}$
$z \xleftarrow{\$} \mathbb{Z}/N^{\zeta}\mathbb{Z}$
return $\mathsf{DJE.Enc}_{\mathsf{pk}}(z)$

are indistinguishable to a PPT adversary who is given pk but not sk. Here, $\mathsf{sk} = (k, N)$ and $\mathsf{pk} = (g, h, N)$ are sampled by generating a random $k \xleftarrow{\$} [0, N)$ and $g \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$, then setting $h \leftarrow g^{-k}$. We present a hybrid proof, starting with the real KDM oracle and ending at the random KDM oracle:

$\mathcal{H}_1$. Sample g instead as $g = g'^{N^{\zeta}}$, where $g' \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$. This is indistinguishable by lemma 17.

$\mathcal{H}_2$. Whenever Enc samples r from $[0, N)$, sample r from $[0, N^{\zeta+1}2^{\lambda})$ instead. Note that r only matters modulo the order of g in $(\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$, and the order of g must divide $\varphi(N) = (p-1)(q-1)$ because g is a perfect N^{ζ}th power. Therefore, this change is statistically indistinguishable because both distributions for r are exponentially close to uniform modulo $\varphi(N)$:

$$\frac{N \bmod \varphi(N)}{N} = \frac{p+q-1}{N} = O(N^{-\frac{1}{2}}) \quad \text{and} \quad \frac{N^{\zeta+1}2^{\lambda} \bmod \varphi(N)}{N^{\zeta+1}2^{\lambda}} < N^{-\zeta}2^{-\lambda}.$$

$\mathcal{H}_3$. Use lemma 17 to change g back to being sampled as $g \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$.

$\mathcal{H}_4$. Sample $s \xleftarrow{\$} [0, N^{\zeta})$ and add $\varphi(N)s$ to r inside Enc. That is, Enc will now calculate $c_0 \leftarrow g^{r+\varphi(N)s}$ and $c_1 \leftarrow h^{r+\varphi(N)s} \cdot \exp(z)$. This is statistically indistinguishable because r is sampled from a much wider range than this shift: $\frac{\varphi(N)N^{\zeta}}{N^{\zeta+1}2^{\lambda}} < 2^{-\lambda}$.

$\mathcal{H}_5$. Rewrite $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}$ as follows, to avoid using sk:

$\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}(x,y)$:
$(g,h,N) \leftarrow \mathsf{pk}$
$r \xleftarrow{\$} [0, N^{\zeta+1}2^{\lambda})$
$s \xleftarrow{\$} [0, N^{\zeta})$
$c_0 \leftarrow g^{r+\varphi(N)s} \cdot \exp(x)$
$c_1 \leftarrow h^{r+\varphi(N)s} \cdot \exp(y)$
return (c_0, c_1)

To see why this is indistinguishable, let $u = \log(g^{\varphi(N)})$ and notice that $-k \cdot u = \log(h^{\varphi(N)})$. Therefore, $c_0 = g^r \exp(u \cdot s)$ and $c_1 = h^r \exp(-k \cdot u \cdot s + x \cdot k + y)$. With probability $1 - N^{-1}$ we have $u \not\equiv 0 \mod N$, so we can substitute $s = (s' + u^{-1}x) \bmod N^{\zeta}$ to get $c_0 = g^r \exp(u \cdot s' + x)$ and $c_1 = h^r \exp(-k \cdot u \cdot s' + y)$. Finally, changing back from $\exp(u)$ and $\exp(-k \cdot u)$ to $g^{\varphi(N)}$ and $h^{\varphi(N)}$, and renaming s' to s, gives the new KDM oracle above.

$\mathcal{H}_6$. Undo the changes from $\mathcal{H}_4$, to remove s from the KDM oracle.

$\mathcal{H}_7$. Once again, let $g = g'^{N^{\zeta}}$, where $g' \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$. This is indistinguishable by lemma 17.

$\mathcal{H}_8$. Sample k from $[0, N^{\zeta+1}2^{\lambda})$ instead $[0, N)$. This is statistically indistinguishable for the same reason as in $\mathcal{H}_2$, as k is now only used to compute $h = g^{-k}$.

$\mathcal{H}_9$. Again, use lemma 17 to change g back to being sampled as $g \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^{\times}$.

$\mathcal{H}_{10}$. Add $\varphi(N)u^{-1} \bmod N^\zeta$ to k when computing h, where $u = \log(g^{\varphi(N)})$. That is, now $h = g^{-k+\varphi(N)u^{-1}} = g^{-k}\exp(1)$. This is statistically indistinguishable for the same reason as $\mathcal{H}_4$.

$\mathcal{H}_{11}$. Again, use lemma 17 to set $g = g'^{N^\zeta}$, where $g' \xleftarrow{\$} (\mathbb{Z}/N^{\zeta+1}\mathbb{Z})^\times$.

$\mathcal{H}_{12}$. Reintroduce s into the KDM oracle.[13] That is, sample $s \xleftarrow{\$} [0, N^\zeta)$ and add $\varphi(N)s$ to r. This is statistically indistinguishable for the same reason as $\mathcal{H}_4$. Because the order of g divides $\varphi(N)$ (as g is a perfect N^ζth power), we have

$$c_0 = g^{r+\varphi(N)s} \cdot \exp(x) = g^r \exp(x)$$
$$c_1 = h^{r+\varphi(N)s} \cdot \exp(y) = g^{-kr} \cdot \exp(\varphi(N)s + y).$$

$\mathcal{H}_{13}$. Replace y with $y' \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$. To show indistinguishability, substitute $s = (s' + \varphi(N)^{-1}(y' - y)) \bmod N^\zeta$, so that now $c_1 = g^{-kr} \cdot \exp(\varphi(N)s' + y')$. And sampling $s' \xleftarrow{\$} [0, N^\zeta)$ is an equivalent distribution to sampling s uniformly.

$\mathcal{H}_{14}$. Now that y has been replaced with random, undo hybrids $\mathcal{H}_{12}$–$\mathcal{H}_1$. We are now have a distribution equivalent to $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},R}$, except that it ignores y and replaces it with a uniformly sampled y'.

$\mathcal{H}_{15}$. Notice that sampling $y' \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$ and computing $z \leftarrow x \cdot k + y'$ is equivalent to just sampling $z \xleftarrow{\$} \mathbb{Z}/N^\zeta\mathbb{Z}$, as the y' works as a one-time pad. Making this change, we are now at $\mathcal{O}^{\mathsf{KDM}}_{\mathsf{sk},\mathsf{pk},\$}$.

A.5 Distributed Discrete Logarithm

DDLOG Damgård-Jurik Distance Function [RS21]

$\mathsf{DDLog}_N(h \in \mathbb{Z}/N^{\zeta+1}\mathbb{Z})$:

Compute and output $z \leftarrow \log\left(\frac{h}{h \bmod N}\right) \in \mathbb{Z}/N^\zeta\mathbb{Z}$

Fig. 9: [RS21]'s distributed discrete logarithm for the Damgård-Jurik cryptosystem [DJ01].

Lemma 18 (Distributed Decryption). *If we have shares over $\mathbb{Z}$ of $\langle x\rangle_1 - \langle x\rangle_0 = x$ and $\langle k \cdot x\rangle_1 - \langle k \cdot x\rangle_0 = k \cdot x$, then*

$$\mathsf{DDLog}_N(c_0^{\langle k\cdot x\rangle_1} c_1^{\langle x\rangle_1}) - \mathsf{DDLog}_N(c_0^{\langle k\cdot x\rangle_0} c_1^{\langle x\rangle_0}) \equiv x \cdot y \mod N^\zeta$$

always holds, for every choice of plaintext size $\zeta \geq 1$, key pair $(\mathsf{sk} = (k, N), \mathsf{pk}) \in \mathsf{Supp}(\mathsf{DJE.KeyGen}(1^\lambda))$, plaintext $y \in \mathbb{Z}/N^\zeta\mathbb{Z}$, ciphertext $(c_0, c_1) \in \mathsf{Supp}(\mathsf{DJE.Enc}_{\mathsf{pk}}(y))$, and scalar $x \in \mathbb{Z}/N^\zeta\mathbb{Z}$.

Proof. Taking the ratio of the inputs to DDLog, we get

$$\frac{c_0^{\langle k\cdot x\rangle_1} c_1^{\langle x\rangle_1}}{c_0^{\langle k\cdot x\rangle_0} c_1^{\langle x\rangle_0}} = c_0^{k\cdot x} c_1^x = \exp(\mathsf{DJE.Dec}_{\mathsf{sk}}(c_0, c_1))^x = \exp(x \cdot y).$$

The second and third equalities are from the definition of DJE.Dec and the correctness of DJE, respectively. The inputs to the DDLogs are therefore multiplicative shares of $\exp(x \cdot y)$, which, by [RS21, Theorem 18], makes the outputs of the DDLogs additive shares over $\mathbb{Z}/N^\zeta\mathbb{Z}$ of $x \cdot y$. □

[13] We couldn't have kept s there the whole time, because the use of $\varphi(N)$ would interfere with using lemma 17 to change how g is sampled.

Lemma 19 (Adapted from [RS21, Lemma 19]). *For all moduli $M > 1$ and all modulo M shares $\langle x\rangle_0, \langle x\rangle_1 \in \mathbb{Z}/M\mathbb{Z}$ of some $x \in \mathbb{Z}$, we have*

$$\Pr_{r \xleftarrow{\$} \mathbb{Z}/M\mathbb{Z}}\Big[(\langle x\rangle_1 + r) \bmod M - (\langle x\rangle_0 + r) \bmod M = x\Big] = \max\left(1 - \frac{|x|}{N^\zeta}, 0\right).$$

B Proofs

B.1 Proof of core lemma 1

The proof of core lemma 1 boils down to proving that all the comments in the pseudocode of $\mathsf{ShareConv}$ are invariants holding with all but negligible probability.
These claims are:

$$\text{For } i \in [r]\text{: } \mathsf{ct}'_i \in \mathsf{Supp}(\mathsf{Enc}(k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}))) \tag{4}$$

$$\text{For } i \in [r]\text{: } \mathsf{ct}''_i \in \mathsf{Supp}(\mathsf{Enc}(f'_i(\mathsf{sh}^{\mathsf{auth}}_{0,1}, \ldots, \mathsf{sh}^{\mathsf{auth}}_{0,1}))) \tag{5}$$

$$\text{For } i \in [r]\text{: } \mathsf{ct}'''_i \in \mathsf{Supp}(\mathsf{Enc}(f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m}))) \tag{6}$$

$$\text{For } i \in [r]\text{: } \mathsf{ct}''''_i \in \mathsf{Supp}(\mathsf{Enc}(k \cdot f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m}))) \tag{7}$$

For $i \in [r]$, $j \in [0, d_i]$, and $\ell \in [d_i - j]$:

$$\mathsf{TAB}_1[i][j][\ell] - \mathsf{TAB}_0[i][j][\ell] = [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^\ell \tag{8}$$

For $i \in [r]$, $j \in [0, d_i]$, and $\ell \in [d_i - j]$:

$$\mathsf{TAB}^{\mathsf{auth}}_1[i][j][\ell] - \mathsf{TAB}^{\mathsf{auth}}_0[i][j][\ell] = k \cdot [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^\ell \tag{9}$$

$$y_1 - y_0 = f(x_1, \ldots, x_n) \tag{10}$$

$$y^{\mathsf{auth}}_1 - y^{\mathsf{auth}}_0 = k \cdot f(x_1, \ldots, x_n) \tag{11}$$

Note that eqs. (10) and (11) imply our desired equalities, because if $y_1 - y_0 \in \mathbb{Z}$ then $\lfloor y_1 \rfloor - \lfloor y_0 \rfloor = y_1 - y_0$.
Let us now prove that these invariants all hold with the desired probability. We do this by first establishing a sequence of problem-reducing implications between these various statements. First observe that for all $i \in [r]$, $|f(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})|, |f'(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})|, |f(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})|, |f'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})| \leq m \cdot B \cdot 2^\lambda \cdot \beta$. In particular, because $k \leq N$, $k \cdot |f(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})|, k \cdot |f'(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})|, k \cdot |f(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})|, k \cdot |f'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})| \leq N \cdot m \cdot B \cdot 2^\lambda \cdot \beta$.

1. ***Equations (4) to (7) hold unconditionally for all $i \in [r]$.*** For each $i \in [r]$, eqs. (4) and (5) holding (with probability 1) follow directly from linear homomorphism of the Damgård-Jurik-ElGamal encryption scheme. Equations (6) and (7) also follow from linear homomorphism, with the observation that $g_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}, 1) = f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) + f_i(0, \ldots, 0) = f_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})$ (for eq. (6)) and $g_i(k \cdot \mathsf{sh}_{0,1}, \ldots, k \cdot \mathsf{sh}_{0,m}, k) = f'_i(k \cdot \mathsf{sh}_{0,1}, \ldots, k \cdot \mathsf{sh}_{0,m}) + f_i(0, \ldots, 0) \cdot k = k \cdot (f_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) - f(0, \ldots, 0)) + f(0, \ldots, 0) \cdot k = k \cdot f(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})$ (for eq. (7)).

2. ***For all $i \in [r]$, if $j = 0$ and $\ell = d_i$, eqs. (8) and (9) holds unconditionally.*** This is true tautologically for all $i \in [r]$ because $\mathsf{TAB}_1[i][0][d_i]$ is defined as 0 and $\mathsf{TAB}_0[i][0][d_i]$ is defined as $[f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i}$, and because $\mathsf{TAB}^{\mathsf{auth}}_1[i][0][d_i]$ is defined as 0 and $\mathsf{TAB}^{\mathsf{auth}}_0[i][0][d_i]$ is defined as $k \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i}$.

3. ***If $\ell = 0$, eq. (8) holds unconditionally for all $i \in [r]$ and $j \in [d_i]$.*** If $\ell = 0$, then eq. (8) is true tautologically for all $i \in [r]$ and $j \in [d_i]$ because $\mathsf{TAB}_1[i][j][0]$ is defined as $[f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j$ and $\mathsf{TAB}_0[i][j][0]$ is defined as 0.

4. ***If $\ell = 0$ and $j = 1$, eq. (9) holds unconditionally for all $i \in [r]$.*** For all $i \in [r]$,

$$\begin{aligned}
\mathsf{TAB}_1^{\mathsf{auth}}[i][1][0] - \mathsf{TAB}_0^{\mathsf{auth}}[i][1][0] =& f'_i(\mathsf{sh}_{1,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}}) - f'_i(\mathsf{sh}_{0,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{0,m}^{\mathsf{auth}}) \\
&+ k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
=& f'_i(\mathsf{sh}_{1,1}^{\mathsf{auth}} - \mathsf{sh}_{0,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}} - \mathsf{sh}_{0,m}^{\mathsf{auth}}) \\
&+ k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
=& f'_i(k \cdot x_1, \ldots, k \cdot x_n) + k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
=& k \cdot f'_i(x_1, \ldots, x_n) + k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
=& k \cdot f'_i(\mathsf{sh}_{1,1} - \mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{1,m} - \mathsf{sh}_{0,m}) \\
&+ k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
=& k \cdot f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})
\end{aligned}$$

by linearity of f'_i and the facts that $\mathsf{sh}_{1,t} - \mathsf{sh}_{0,t} = x_t$ and $\mathsf{sh}_{1,t}^{\mathsf{auth}} - \mathsf{sh}_{0,t}^{\mathsf{auth}} = k \cdot x_t$. Hence eq. (9) holds unconditionally for all $i \in [r]$ is $(j, \ell) = (1, 0)$.

5. ***If $\ell = 0$, then for all $i \in [r]$ and $j \in [2, d_i]$, if eq. (9) holds for $(i, j-1, \ell = 0)$ then it holds with probability at least $1 - 4(nB2^\lambda\beta)^d/N^{\zeta-1}$ for $(i, j, \ell = 0)$.*** For all $i \in [r]$ and $j \in [2, d_i]$, if we assume eq. (9) hold for $(i, j-1, \ell = 0)$ then, because , the following hold with probability at least $1 - 4(nB2^\lambda\beta)^d/N^{\zeta-1}$ by lemma 6 (using eqs. (4) and (5)):

$$\begin{aligned}
\mathsf{TAB}_1^{\mathsf{auth}}[i][j][0] - \mathsf{TAB}_0^{\mathsf{auth}}[i][j][0] =& f'_i(\mathsf{sh}_{1,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}}) \cdot [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\
&+ \mathsf{Mult}(\mathsf{TAB}_1[i][j-1][0], \mathsf{TAB}_1^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}'_i, \mathsf{crs}_{i,j,1}) \\
&- \mathsf{Mult}(\mathsf{TAB}_1[i][j-1][0], \mathsf{TAB}_1^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}''_i, \mathsf{crs}_{i,j,2}) \\
&- \mathsf{Mult}(\mathsf{TAB}_0[i][j-1][0], \mathsf{TAB}_0^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}'_i, \mathsf{crs}_{i,j,1}) \\
&+ \mathsf{Mult}(\mathsf{TAB}_0[i][j-1][0], \mathsf{TAB}_0^{\mathsf{auth}}[i][j-1][0], \mathsf{ct}''_i, \mathsf{crs}_{i,j,2}) \\
=& f'_i(\mathsf{sh}_{1,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}}) \cdot [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\
&+ [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \cdot k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) \\
&- [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \cdot f'_i(\mathsf{sh}_{0,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{0,m}^{\mathsf{auth}}) \\
=& [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\
&\cdot [f'_i(\mathsf{sh}_{1,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{1,m}^{\mathsf{auth}}) + k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m}) - f'_i(\mathsf{sh}_{0,1}^{\mathsf{auth}}, \ldots, \mathsf{sh}_{0,m}^{\mathsf{auth}})] \\
=& [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\
&\cdot [f'_i(k \cdot x_1, \ldots, k \cdot x_n) + k \cdot f'_i(\mathsf{sh}_{0,1}, \ldots, \mathsf{sh}_{0,m})] \\
=& [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^{j-1} \\
&\cdot k \cdot f'_i(x_1 + \mathsf{sh}_{0,1}, \ldots, x_n + \mathsf{sh}_{0,m}) \\
=& k \cdot [f'_i(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j
\end{aligned}$$

6. ***For all $i \in [r]$, $j \in [d_i]$, and $\ell \in [d_i - j]$, if eqs. (8) and (9) hold for $(i, j, \ell - 1)$ then they hold with probability at least $1 - 2(nB2^\lambda\beta)^d/N^{\zeta-1}$ for $(i, j, \ell - 1)$.***

Let $(i, j, \ell) \in [r] \times [d_i] \times [d_i - j]$ and assume eqs. (8) and (9) both hold.

By lemma 6 (and using eq. (6)), with all but probability at most $2(nB2^{\lambda}\beta)^d/N^{\zeta-1}$:

$$\begin{aligned}
\mathsf{TAB}_1[i][j][\ell] - \mathsf{TAB}_0[i][j][\ell] =& \mathsf{Mult}(\mathsf{TAB}_1[i][j][\ell-1], \mathsf{TAB}_1^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i''', \mathsf{crs}_{i,j,3}) \\
& - \mathsf{Mult}(\mathsf{TAB}_1[i][j][\ell-1], \mathsf{TAB}_1^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i''', \mathsf{crs}_{i,j,3}) \\
=& [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{\ell-1} \cdot f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m}) \\
=& [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{\ell}
\end{aligned}$$

which is to say eq. (8) holds for (i, j, ℓ).

Similarly, by lemma 6 (and using eq. (7)), with probability at least $1 - 2B/N^{\varepsilon}$:

$$\begin{aligned}
\mathsf{TAB}_1^{\mathsf{auth}}[i][j][\ell] - \mathsf{TAB}_0^{\mathsf{auth}}[i][j][\ell] =& \mathsf{Mult}(\mathsf{TAB}_1[i][j][\ell-1], \mathsf{TAB}_1^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i'''', \mathsf{crs}_{i,j,4}) \\
& - \mathsf{Mult}(\mathsf{TAB}_1[i][j][\ell-1], \mathsf{TAB}_1^{\mathsf{auth}}[i][j][\ell-1], \mathsf{ct}_i'''', \mathsf{crs}_{i,j,4}) \\
=& [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{\ell-1} \cdot k \cdot f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh} \\
=& k \cdot [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{\ell}
\end{aligned}$$

which is to say eq. (9) holds for (i, j, ℓ).

7. ***If, for all $i \in [r]$, $j \in [d_i]$, eq. (8) holds for $(i, j, \ell = d_i - j)$, then eq. (10) holds.*** Assume eq. (8) holds for $(i, j, \ell = d_i - j)$ for all $(i, j) \in [r] \times [d_i]$, then

$$\begin{aligned}
y_1 - y_0 =& \sum_{i=1}^{r} \sum_{j=1}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot (\mathsf{TAB}_1[i][j][d_i - j] - \mathsf{TAB}_0[i][j][d_i - j]) \\
=& \sum_{i=1}^{r} \sum_{j=1}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i - j} \\
=& \sum_{i=1}^{r} \alpha_i \cdot \left(f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m}) - f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})\right)^{d_i} \\
=& \sum_{i=1}^{r} \alpha_i \cdot (f_i(x_1, \ldots, x_n))^{d_i} \\
=& f(x)
\end{aligned}$$

That is to say, eq. (10) holds.

8. ***If, for all $i \in [r]$, $j \in [d_i]$, eq. (9) holds for $(i, j, \ell = d_i - j)$, then eq. (11) holds.***

Assume eq. (9) holds for $(i, j, \ell = d_i - j)$ for all $(i, j) \in [r] \times [d_i]$, then

$$\begin{aligned}
y_1^{\mathsf{auth}} - y_0^{\mathsf{auth}} =& \sum_{i=1}^{r} \sum_{j=0}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot \left(\mathsf{TAB}_1^{\mathsf{auth}}[i][j][d_i - j] - \mathsf{TAB}_0^{\mathsf{auth}}[i][j][d_i - j] \right) \\
=& \sum_{i=1}^{r} \sum_{j=0}^{d_i} \alpha_i \cdot \binom{d_i}{j} \cdot (-1)^{d_i - j} \cdot k \cdot [f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m})]^j \cdot [f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m})]^{d_i - j} \\
=& k \cdot \sum_{i=1}^{r} \alpha_i \cdot \left(f_i'(\mathsf{sh}_{1,1}, \ldots, \mathsf{sh}_{1,m}) - f_i(-\mathsf{sh}_{0,1}, \ldots, -\mathsf{sh}_{0,m}) \right)^{d_i} \\
=& k \cdot \sum_{i=1}^{r} \alpha_i \cdot (f_i(x_1, \ldots, x_n))^{d_i} \\
=& k \cdot f(x)
\end{aligned}$$

That is to say, eq. (11) holds.

By combining these polynomially many implications, we get that eqs. (10) and (11) hold simultaneously with probability at least $1 - 4rd^2(nB2^\lambda\beta)^d/N^{\zeta-1}$ (by a union bound), which is what we set out to prove.

B.2 Proof of main theorem 1

In this section we prove main theorem 1. Even though definition 5 is simulation-based, we start by establishing a strong notion of correctness in section B.2.1 for convenience, before resuming the proof in section B.2.2.

B.2.1 Correctness lemma.

Lemma 20 (Correctness of the PCPRF of fig. 7). *Let $\lambda \in \mathbb{N}^*$ be a security parameter, let $(\lambda_{\mathsf{DCR}}, \zeta)$ be parameters for the Damgård-Jurik-ElGamal cryptosystem, let B be a bound on inputs. Let r, d, β be polynomial-size bounds. Consider the construction of fig. 7 for any class of constraints C whose dual have Waring rank bounded by r and degree bounded by d, and furthermore such that the coefficients of affine functions used in a corresponding Waring decomposition (*i.e. *"the f_i") are bounded by β.*

$$\begin{aligned}
&\forall C \in \mathcal{C}, \\
&\Pr\left[\begin{array}{l} \forall x \in \{0,1\}^m, \forall k_{\mathsf{PRF}} \in \{0,1\}^\lambda,\ B\text{-admissible w.r.t. } \mathsf{PRF}, \\ \mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - \mathsf{Eval}(1^\lambda, \mathsf{msk}, x) = C(x) \cdot \mathsf{PRF}_{k_{\mathsf{PRF}}}(x) \end{array} : \begin{array}{r} \mathsf{msk} \xleftarrow{\$} \mathsf{PCPRF.KeyGen}(1^\lambda) \\ \mathsf{ck} \xleftarrow{\$} \mathsf{PCPRF.Constrain}(1^\lambda, \mathsf{msk}, C) \end{array} \right] \\
&\qquad \geq 1 - \frac{2|\mathsf{PRF}| \cdot B + 4rd^2(nB2^\lambda\beta)^d}{(2^{\lambda_{\mathsf{DJE}}-1})^{\zeta-1}} \cdot 2^n 2^\lambda \ .
\end{aligned}$$

Proof. Let $C \in \mathcal{C}$, $x \in \{0,1\}^m$, and $k_{\mathsf{PRF}} \in \{0,1\}^\lambda$ which is B-admissible with respect to bound B. By combining core lemma 1 and lemma 8, we get that

$$\begin{aligned}
&\Pr\left[\mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - \mathsf{Eval}(1^\lambda, \mathsf{msk}, x) = C(x) \cdot \mathsf{PRF}_{k_{\mathsf{PRF}}}(x) : \begin{array}{r} \mathsf{msk} \xleftarrow{\$} \mathsf{PCPRF.KeyGen}(1^\lambda) \\ \mathsf{ck} \xleftarrow{\$} \mathsf{PCPRF.Constrain}(1^\lambda, \mathsf{msk}, C) \end{array} \right] \\
&\qquad \geq 1 - \frac{2|\mathsf{PRF}| \cdot B + 4rd^2(nB2^\lambda\beta)^d}{(2^{\lambda_{\mathsf{DJE}}-1})^{\zeta-1}} \ .
\end{aligned}$$

We get the desired result by a union bound over all x and k_{PRF}. □

B.2.2 Full simulation-based proof. We now prove single-key selective PCPRF security, using the strong correctness (lemma 20) of our PCPRF, and the KDM security of the underlying Damgård–Jurik–ElGamal encryption scheme (theorem 16). We must show that the constrained key can be simulated using only the size of the constraint, such that oracle access to master evaluations $\mathsf{Eval}(1^\lambda, \mathsf{msk}, \cdot)$ in the real world is indistinguishable from oracle access to

$$x \mapsto \begin{cases} \mathsf{CEval}(1^\lambda, \mathsf{ck}, x) & \text{if } C(x) = 0 \\ R(x) & \text{if } C(x) = 1 \end{cases},$$

in the simulated world, where $R\colon \{0,1\}^n \to [0, 2^\mu)$ is sampled as a uniformly random function. In fig. 10 we present our simulator for this property.

Simulator Privately Constrained PRF

$\mathcal{S}(1^\lambda, 1^m)$:

1. $(\mathsf{sk} = (k, N), \mathsf{pk} = (g, h, N)) \xleftarrow{\$} \mathsf{DJE.KeyGen}(1^\lambda)$
2. For $i \in [m]$:
   - $\mathsf{sh}_{1,i} \xleftarrow{\$} [2^\lambda]$
   - $\mathsf{sh}^{\mathsf{auth}}_{1,i} \xleftarrow{\$} [N2^\lambda]$
   - $c_i, c^{\mathsf{auth}}_i, c^{\mathsf{ext\text{-}auth}}_i \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\$)$ // All are encryptions of uniformly random plaintexts.
3. For $i \in [\lambda]$:
   - $c_{\mathsf{PRF},i} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\$)$
   - $c^{\mathsf{ext\text{-}auth}}_{\mathsf{PRF},i} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\$)$
4. $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2) \xleftarrow{\$} (\mathbb{Z}/N^\zeta\mathbb{Z})^{4 \cdot rd} \times (\mathbb{Z}/N^\zeta\mathbb{Z})^M$
5. $\mathsf{ck} \leftarrow (\ \mathsf{pk},$
   $(\mathsf{sh}_{1,i}, \mathsf{sh}^{\mathsf{auth}}_{1,i}, c_i, c^{\mathsf{auth}}_i, c^{\mathsf{ext\text{-}auth}}_i)_{i \in [m]},$
   $(c_{\mathsf{PRF},i}, c^{\mathsf{ext\text{-}auth}}_{\mathsf{PRF},i})_{i \in [\lambda]},$
   $\mathsf{crs})$
6. Output ck

Fig. 10: Simulator for single-key selective security.

We now present a hybrid proof starting from the real world and ending at the simulated world.

$\mathcal{H}_1$. First, we use lemma 20 to rewrite the oracle access to $\mathsf{Eval}(1^\lambda, \mathsf{msk}, \cdot)$ in terms of $\mathsf{PEval}(1^\lambda, \mathsf{ck}, \cdot)$. According to the lemma, λ_{DJE} and ζ can be chosen as sufficiently large polynomials in the security parameter so that except with negligible probability we have that, for all evaluation points x,

$$\mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - \mathsf{Eval}(1^\lambda, \mathsf{msk}, x) \equiv C(x) \cdot \mathsf{PRF}_{k_{\mathsf{PRF}}}(x) \mod 2^\mu.$$

Therefore, we rewrite the Eval oracle as

$$x \mapsto (\mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - C(x) \cdot \mathsf{PRF}_{k_{\mathsf{PRF}}}(x)) \bmod 2^\mu.$$

$\mathcal{H}_2$. For all $i \in [m]$, instead of sampling $\mathsf{sh}_{0,i} \xleftarrow{\$} [2^\lambda]$ and setting $\mathsf{sh}_{1,i} \leftarrow \widehat{C}_i + \mathsf{sh}_{0,i}$, sample $\mathsf{sh}_{1,i} \xleftarrow{\$} [2^\lambda]$ and set $\mathsf{sh}_{0,i} \leftarrow -\widehat{C}_i + \mathsf{sh}_{0,1}$. This shifts the distribution of the shares by at most 1, so each change gives an advantage of at most $2^{-\lambda}$. Similarly, we reverse which

share is sampled first for the auth shares: now $\mathsf{sh}^{\mathsf{auth}}_{1,i} \xleftarrow{\$} [N2^\lambda]$ and $\mathsf{sh}^{\mathsf{auth}}_{0,i} \leftarrow -k\cdot\widehat{C}_i + \mathsf{sh}^{\mathsf{auth}}_{1,i}$. This shifts the shares by at most $k \le N$, so again each change has an advantage of at most $2^{-\lambda}$. The total advantage from this hybrid is then at most $m2^{1-\lambda}$.

$\mathcal{H}_3$. Notice that there are now only two places where k is used after $\mathsf{DJE.KeyGen}$: (i) to compute master key auth shares $\mathsf{sh}^{\mathsf{auth}}_{0,\cdots}$, which only used to create ciphertexts $c^{\mathsf{auth}}_{\cdots} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(\mathsf{sh}^{\mathsf{auth}}_{0,\cdots})$, and (ii) to generate the ext-auth ciphertexts $c^{\mathsf{ext\text{-}auth}}_{\cdots} \xleftarrow{\$} \mathsf{DJE.Enc}_{\mathsf{pk}}(k \cdot \mathsf{sh}_{0,\cdots})$. Both of these are only encryptions of affine functions of the key, so by the KDM security of DJE it is indistinguishable to replace all ciphertexts with encryptions of uniformly random values.

$\mathcal{H}_4$. Notice that the master key shares $\mathsf{sh}_{0,\cdots}$ and $\mathsf{sh}^{\mathsf{auth}}_{0,\cdots}$ are no longer used. Remove them.

$\mathcal{H}_5$. Notice that the PRF key k_{PRF} is now only used in the Eval oracle. Therefore, by security of the PRF we can remove k_{PRF}, and instead sample a uniformly random function $R' \colon \{0,1\}^n \to [0, 2^\mu)$ and let the Eval oracle be

$$x \mapsto (\mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - C(x) \cdot R'(x)) \bmod 2^\mu.$$

$\mathcal{H}_6$. Define $R(x) = (\mathsf{CEval}(1^\lambda, \mathsf{ck}, x) - R'(x)) \bmod 2^\mu$. Then we can write the Eval oracle as

$$x \mapsto \begin{cases} \mathsf{CEval}(1^\lambda, \mathsf{ck}, x) & \text{if } C(x) = 0 \\ R(x) & \text{if } C(x) = 1 \end{cases}.$$

$\mathcal{H}_7$. Because R' has outputs uniform on $[0, 2^\mu)$, it is an identical distribution to remove R' and directly sample R uniformly. This hybrid matches the simulation.