Private Constrained Pseudorandom Functions with Succinct Keys

Pedro Capitão

Instituto Superior Técnico

July 2021

Abstract

A constrained pseudorandom function is a pseudorandom function (PRF) in which constrained keys can be derived from the master secret key. Each constrained key is associated with a constraint f and allows its user to evaluate the PRF at points x satisfying $f(x) = 0$, but gives no information about the PRF values at points x such that $f(x) = 1$. In a private constrained PRF, constrained keys do not reveal their corresponding constraints.

We consider the problem of building a private constrained PRF in which the size of the constrained keys is independent of the constraints. We show that this is possible to achieve under a generalized definition of constrained PRF, in which the public parameters may be updated whenever a key is generated. We provide two distinct constructions that fulfill these requirements, starting from a private constrained PRF and using as tools the cryptographic primitives of attribute-based encryption and functional encodings, respectively. Their security is based on the hardness of the learning with errors problem, which is related to the intractability of well-studied problems on lattices.

1 Introduction

Lattice cryptography is the study of cryptographic protocols provably secure under the assumption of intractability of certain computational problems on lattices. Such problems are conjectured to not be efficiently solvable by classical or quantum algorithms, making these systems quantum-resistant. But security against quantum attacks is not the only noteworthy quality of lattice cryptography. The problem of building fully-homomorphic encryption, a powerful cryptographic primitive which allows computation over encrypted data, remained largely unanswered for thirty years until the work of Gentry [Gen09], which used lattices to give the first candidate construction based on cryptographic assumptions. Lattice cryptography has provided the first (and, in some cases, all) constructions of this and other useful cryptographic primitives such as attribute-based encryption for arbitrary constraints [GVW13] or indistiguishability obfuscation [GGH$^+$13]. Moreover, lattice-based protocols are generally simple, efficient and parallelizable. The cryptographic protocols studied in this work rely on the hardness of the learning with errors (LWE) problem for their security. LWE was introduced by Regev [Reg05] and is a computational problem related to lattices with wide application in cryptography.

In this work we study the cryptographic protocol known as private constrained pseudorandom function. A pseudorandom function (PRF) [GGM86] is a keyed function such that, for a random key, its outputs are indistinguishable from those of a truly random function. In a constrained pseudorandom function (CPRF) [BW13,

KPTZ13, BGI14], the owner can delegate constrained keys which allow other parties to evaluate the PRF. However, each key is associated with a constraint (a predicate over the domain of the PRF) and only evaluates the PRF correctly at points which satisfy that constraint.

A CPRF is said to be private (or constraint-hiding) [BLW17] if the constrained keys reveal no information about the corresponding constraints. Constructions of private CPRFs for different classes of constraints have been proposed, including for point-functions [BKM17], for constraints in NC^1 [CC17], and for all polynomial-size circuits [BLW17, BTVW17, PS18]. All of these are based on LWE, with the exception of [BLW17], which requires the powerful cryptographic primitive of indistinguishability obfuscation [BGI$^+$12] and is also the only one that achieves collusion resistance.

A desirable property for CPRFs is that the constrained keys are short. Ideally, the size of such a key should be (asymptotically) independent from the constraint that is associated to it – when this is the case we say that the keys are succinct. An example of this is the LWE-based CPRF proposed by Brakerski and Vaikuntanathan [BV15], which has succinct constrained keys. However, the problem appears to be harder when we move from standard CPRFs to private CPRFs.

Our main contribution is a private CPRF with succinct keys for the constraint class of all polynomial-size circuits (with bounded depth). We provide two different constructions matching this description, both of which build upon a private CPRF [BTVW17] which supports the same constraint class, but in which the keys are not succinct. The first relies on the internal structure of this private CPRF and on the use of attribute-based encryption with short keys [BGG$^+$14]. The second uses functional encodings [WW21] and a private CPRF as a black box. We do not know of any previous private CPRF construction with succinct constrained keys.

There is a caveat to our results – in order to achieve the property of succinct keys, we resorted to a generalized definition of constrained PRF. In this new notion, which we call CPRF with updatable parameters, the public parameters of the scheme can be updated whenever a constrained key is generated (this is equivalent to generating both a public and a private constrained key). We argue that little is lost by considering this definition, as it is a simple and intuitive generalization and it can still be used in all applications of private CPRFs that we are aware of.

In terms of security, both of our constructions satisfy essentially the same definition as the underlying private CPRF, which is single-key selective security. This means that the scheme is resistant against an adversary that has access to only one constrained key, chosen at the start of the security game. While this is a relatively weak notion, there are indicators that achieving stronger security is substantially harder. For instance, it has been shown that a 2-key secure private CPRF implies the existence of indistinguishability obfuscation [CC17] and that a certain simulation-based definition of full (or adaptive) security is impossible to obtain [BKM17]. We define two generalizations of single-key selective security for CPRFs with updatable parameters. Our first construction satisfies only the weaker of the two definitions, while the second satisfies both.

2 Preliminaries

We begin by establishing some notation that will be used throughout the thesis. If χ is a probability distribution over a set X, the expression $x \leftarrow \chi$ denotes that x is a random variable with distribution χ, and, when X is finite, $x \leftarrow X$

denotes that the distribution of x is the uniform distribution on X. If Alg is a probabilistic algorithm, $y \leftarrow \mathsf{Alg}(x)$ indicates that y is a random variable distributed according to the output of Alg on input x. For a variable x considered as the input or output of some algorithm, $|x|$ denotes its size in bits. The expression $\mathsf{poly}(\cdot)$ denotes any polynomial function; for instance, $f(n) \leqslant \mathsf{poly}(n)$ means that f is bounded by some polynomial. Likewise, $\mathsf{negl}(\cdot)$ denotes a negligible function: a function f such that, for any positive polynomial p, $f(n) \leqslant \frac{1}{p(n)}$ for sufficiently large n. We use the standard asymptotic notation $\mathcal{O}(\cdot)$ and its variant $\tilde{\mathcal{O}}(\cdot)$ which ignores logarithmic factors in the indicated variable.

The expression $\log$ always denotes the logarithm in base 2. For a real number x, we denote by $\lceil x \rceil$ the smallest integer that is greater than or equal to x, and by $\lfloor x \rceil = \lceil x - \frac{1}{2} \rceil$ the integer closest to x. For a natural number q, the symbol $\mathbb{Z}_q$ represents the ring of integers modulo q. We also define $[n] = \{1, 2, \ldots, n\}$. If $\mathbf{A}_1, \mathbf{A}_2$ are matrices, $[\mathbf{A}_1 \mid \mathbf{A}_2]$ denotes their horizontal concatenation and $\left(\begin{smallmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{smallmatrix}\right)$ their vertical concatenation.

2.1 Learning with errors

The learning with errors problem (LWE) was introduced by Regev [Reg05] and is the basis of many cryptographic constructions, including the ones used in this work. Below we formulate the decision version of LWE.

Definition 2.1 (LWE)**.** *Let n, m, q be positive integers and χ a probability distribution over $\mathbb{Z}$. The* decisional learning with errors problem $\mathrm{LWE}_{n,m,q,\chi}$ *is to distinguish the distributions*

$$(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e}) \text{ and } (\mathbf{A}, \mathbf{u}),$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$ and $\mathbf{u} \leftarrow \mathbb{Z}_q^m$.

2.2 Constrained pseudorandom functions

In a constrained pseudorandom function (CPRF), the owner of the master secret key can generate constrained keys corresponding to constraints $f : \{0,1\}^z \to \{0,1\}$. Such a key allows its holder to compute the value of the PRF on points $x \in \{0,1\}^z$ such that $f(x) = 0$, i.e. x satisfies f.[1] Constrained PRFs were proposed independently by Boneh and Waters [BW13], Kiayias, Papadopoulos, Triandopoulos, and Zacharias [KPTZ13], and Boyle, Goldwasser, and Ivan [BGI14].

Following [BTVW17] and without loss of generality, we consider CPRFs with domain $\{0,1\}^z$, range $\mathbb{Z}_p$ and constraints represented by strings in $\{0,1\}^\ell$. All the CPRF constructions featured in this work are for the class of all constraints computable by Boolean circuits of polynomial size and a priori bounded depth t.

Definition 2.2 (Constrained PRF)**.** *A* constrained pseudorandom function *is a tuple* $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Constrain}, \mathsf{ConstrainEval})$ *of polynomial-time algorithms with the following syntax:*

- $\mathsf{KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$ *is a probabilistic algorithm that receives as input the security parameter λ, the maximum description length ℓ of constraint functions, their input length z and their maximum depth t. It outputs a master secret key* msk *and public parameters* pp.

[1] Since we are working with the CPRF of [BTVW17], we follow their convention of writing $f(x) = 0$ when x satisfies f. Note that this is contrary to the more common convention in which f is said to be satisfied if $f(x) = 1$, but the two are equivalent.

- $\mathsf{Eval}_{\mathsf{pp}}(\mathsf{msk}, x)$ *is a deterministic algorithm that receives as input a key* msk *and a string* $x \in \{0,1\}^z$*, and outputs a value* $y \in \mathbb{Z}_p$*.*
- $\mathsf{Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$ *is a probabilistic algorithm that receives as input a key* msk *and a circuit* $f : \{0,1\}^z \to \{0,1\}$*. It outputs a constrained key* ck*.*
- $\mathsf{ConstrainEval}_{\mathsf{pp}}(\mathsf{ck}, x)$ *is a deterministic algorithm that receives as input a string* $x \in \{0,1\}^z$ *and a constrained key* ck*. It outputs a value* $y \in \mathbb{Z}_p$*.*

Correctness. For any $x \in \{0,1\}^z$ and $f \in \{0,1\}^\ell$ such that $f(x) = 0$,

$$\Pr\left[\mathsf{Eval}_{\mathsf{pp}}(\mathsf{msk}, x) = \mathsf{ConstrainEval}_{\mathsf{pp}}(\mathsf{ck}, x)\right] \geqslant 1 - \mathrm{negl}(\lambda),$$

where $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\mathsf{ck} \leftarrow \mathsf{Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$ and the probability is taken over the randomness of KeyGen and $\mathsf{Constrain}$.

A constrained PRF is secure if it is pseudorandom at constrained points. This means that a key corresponding to a constraint f gives no information about the PRF output at input points that do not satisfy f. A stronger form of this protocol, named private constrained pseudorandom function, was introduced by Boneh, Lewi and Wu [BLW17]. These have the additional security requirement of constraint hiding (or privacy), which states that constrained keys should not reveal information about their corresponding constraints.

Succinct keys. We say that a constrained PRF has *succinct keys* if the size of each constrained key is asymptotically independent of the size of the description of the constraint, depending only on the security parameter λ. More specifically, the scheme has succinct keys if there exists a polynomial $p(\cdot)$ such that, for any polynomial $\ell(\cdot)$, if ck is a constrained key corresponding to $f \in \{0,1\}^{\ell(\lambda)}$, then $|\mathrm{ck}| \leqslant p(\lambda)$ for sufficiently large λ.

2.3 Updatable parameters

We define a generalized notion of constrained PRFs, in which the public parameters are updated whenever a constrained key is generated.

Syntax. A *constrained pseudorandom function with updatable parameters* is a tuple of algorithms $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Constrain}, \mathsf{ConstrainEval})$ satisfying the conditions of Definition 2.2, with the following difference: the $\mathsf{Constrain}$ algorithm outputs a pair $(\mathsf{pp}', \mathsf{ck})$ consisting of new public parameters pp' and a constrained key ck.

Correctness. Let $f \in \{0,1\}^\ell$ be a constraint, $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and $(\mathsf{pp}', \mathsf{ck}) \leftarrow \mathsf{Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$. Then $\mathsf{Eval}_{\mathsf{pp}'}(\mathsf{msk}, x) = \mathsf{Eval}_{\mathsf{pp}}(\mathsf{msk}, x)$ for any $x \in \{0,1\}^z$. Moreover, if $f(x) = 0$ then

$$\Pr\left[\mathsf{Eval}_{\mathsf{pp}'}(\mathsf{msk}, x) = \mathsf{ConstrainEval}_{\mathsf{pp}'}(\mathsf{ck}, x)\right] \geqslant 1 - \mathrm{negl}(\lambda).$$

In the usual definition of a constrained PRF, the $\mathsf{Constrain}$ algorithm outputs only a constrained key. This definition includes that notion as the particular case in which the public parameters are not updated, i.e. $\mathsf{pp}' = \mathsf{pp}$. Note that in our constructions the updates are always incremental – if the parameters pp are replaced by pp', then pp' contains all the information present in pp. Therefore this new notion could equivalently be defined as a CPRF

in which the constrained key is split into two parts: a private part (which corresponds to the actual constrained key) and a public part (the additional information added to the parameters).

When clear from the context, we will often omit the expression "with updatable parameters". Note that both of our proposed constructions, labelled SCPRF in Chapters 3 and 4, are private CPRFs with updatable parameters.

Regarding the size of constrained keys, the main advantage of this new definition is that it allows the Constrain algorithm to generate additional information that does not need to be succinct (because it becomes part of the parameters) while still having succinct keys. However, we need a new security definition that reflects the fact that the updated parameters are assumed to be public – otherwise any CPRF could trivially be made into a succinct-key CPRF with updatable parameters by placing the old constrained key in the updated parameters and letting the new constrained key be empty.

We define two different notions of security for private constrained PRFs with updatable parameters, both of which generalize the definition of single-key selective security for private constrained PRFs. In the first, which we call weak security, the adversary has access to updated public parameters that are generated for uniformly chosen constraints, unknown to the adversary. This corresponds to a real-world adversary waiting for the parameters to be updated as keys are generated for other users.

Definition 2.3 (Weak security)**.** *A constrained pseudorandom function with updatable parameters* $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Constrain}, \mathsf{ConstrainEval})$ *is a* weak single-key selective private constrained pseudorandom function *if the following conditions hold:*

- **Pseudorandomness at constrained points.** *Consider the following game between a challenger and a stateful PPT adversary $\mathcal{A}$:*

  1. *$\mathcal{A}$ sends 1^ℓ, 1^t and $f \in \{0,1\}^\ell$ to the challenger.*
  2. *The challenger generates $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$, $(\mathsf{pp}', \mathsf{ck}) \leftarrow \mathsf{Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$. It flips a coin $b \leftarrow \{0,1\}$ and sends $(\mathsf{pp}', \mathsf{ck})$ to $\mathcal{A}$.*
  3. *In this phase $\mathcal{A}$ can send queries $x \in \{0,1\}^z$ such that $f(x) = 1$ (with no value x queried more than once), to which the challenger replies with $y = \mathsf{Eval}_{\mathsf{pp}'}(\mathsf{msk}, x)$, if $b = 0$, or $y \leftarrow \mathbb{Z}_p$, if $b = 1$. $\mathcal{A}$ may also request updated parameters, in which case the challenger samples $g \leftarrow \{0,1\}^\ell$, computes $(\mathsf{pp}', \mathsf{ck}_g) \leftarrow \mathsf{Constrain}_{\mathsf{pp}'}(\mathsf{msk}, g)$ and sends the new parameters pp' to $\mathcal{A}$.*
  4. *$\mathcal{A}$ outputs a guess $b' \in \{0,1\}$.*

  Then $\left|\Pr[b' = b] - \frac{1}{2}\right| = \mathrm{negl}(\lambda)$ for any adversary $\mathcal{A}$.

- **Constraint hiding.** *Consider the following game between a challenger and a stateful PPT adversary $\mathcal{A}$:*

  1. *$\mathcal{A}$ sends 1^ℓ, 1^t and $f^0, f^1 \in \{0,1\}^\ell$ to the challenger.*
  2. *The challenger generates $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$. It flips a coin $b \leftarrow \{0,1\}$ and sends $(\mathsf{pp}', \mathsf{ck}) \leftarrow \mathsf{Constrain}_{\mathsf{pp}}(\mathsf{msk}, f^b)$ to $\mathcal{A}$.*
  3. *In this phase $\mathcal{A}$ can send queries $x \in \{0,1\}^z$ such that $f^0(x) = f^1(x)$, to which the challenger replies with $y = \mathsf{Eval}_{\mathsf{pp}'}(\mathsf{msk}, x)$. $\mathcal{A}$ may also request updated parameters, in which case the challenger*

samples $g \leftarrow \{0,1\}^\ell$*, computes* $(\mathsf{pp}', \mathsf{ck}_g) \leftarrow \mathsf{Constrain}_{\mathsf{pp}'}(\mathsf{msk}, g)$ *and sends the new parameters* pp' *to* $\mathcal{A}$*.*

4. $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$*.*

Then $\left|\Pr[b' = b] - \frac{1}{2}\right| = \mathrm{negl}(\lambda)$ *for any adversary* $\mathcal{A}$*.*

Our proposed private constrained PRF in Chapter 4 satisfies a stronger form of security, which we define next. In this definition, the adversary can choose the constraints for which the challenger must generate updated parameters. This is a natural strengthening of the previous definition, as in many applications it may not be justified to assume that the distribution of constraints for which the parameters are updated is uniform – it is likely that some constraints are much more probable than others, or it may even be the case that a large portion of the possible constraints is meaningless and has probability zero.

Definition 2.4 (Strong security)**.** *A constrained pseudorandom function with updatable parameters is a* strong single-key selective private constrained pseudorandom function *if any adversary* $\mathcal{A}$ *has negligible advantage in the games of Definition 2.3 when the constraints* g *for which the parameters are updated are chosen by* $\mathcal{A}$*.*

As the names indicate, strong single-key selective security implies weak single-key selective security. The converse is not true – our first construction of a private constrained PRF, in Section 3, satisfies Definition 2.3 but not Definition 2.4.

3 Succinct-key private constrained PRF from attribute-based encryption

The starting point for this construction is the private CPRF of Brakerski, Tsabary, Vaikuntanathan, and Wee [BTVW17][2], in which the first component of a constrained key is of the form $(\Psi_1, \ldots, \Psi_\ell)$ and each Ψ_i only depends on the i-th bit f_i of the description of the constraint f. Relying on this fact, we encrypt with ABE each Ψ_i for both the cases $f_i = 0$ and $f_i = 1$ and place those encryptions in the public parameters. Then we give as the constrained key for our scheme an ABE constrained key that allows the user to decrypt only the relevant ciphertexts in the parameters. A similar procedure applies to the second component of the key. This technique has been used before by Brakerski and Vaikuntanathan [BV15] to turn their construction of a standard CPRF into a CPRF with succinct keys. Additionally, we permutate the aforementioned ciphertexts using bits $d_1, \ldots, d_\ell$ in order to keep the constraint hidden. The additional information contained in the updated parameters allows the user to decrypt the correct ciphertexts, despite this permutation.

Consider the following two subroutines of the algorithm BTVW.Constrain, which we highlight in order to make the presentation clearer.

- $\mathsf{BTVW.GenP}_{\mathsf{pp}}(\mathsf{msk}, \mathbf{e}_0, \beta \in \{0,1\})$: Sample $\mathbf{R} \leftarrow \{0,1\}^{(n+1)\log q \times (n+1)\log q}$. Output

$$\Psi = \begin{pmatrix} \mathbf{B} \\ \mathbf{s}^\mathsf{T}\mathbf{B} + \mathbf{e}_0^\mathsf{T} \end{pmatrix} \mathbf{R} + \beta \mathbf{G}.$$

[2] In [BTVW17] two constructions of a private CPRF are proposed. Here we always refer to the first, which has the title "The dual-use technique".

- $\mathsf{BTVW.GenC_{pp}}(\mathsf{msk}, \mathbf{A}, \beta \in \{0,1\})$: Sample $\mathbf{e} \leftarrow \chi^{(n+1)\log q}$. Output

$$\mathbf{c}^\mathsf{T} = \mathbf{s}^\mathsf{T}(\mathbf{A} - \beta\overline{\mathbf{G}}) + \mathbf{e}^\mathsf{T},$$

  where $\overline{\mathbf{G}}$ denotes the gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{(n+1)\times(n+1)\log q}$ with its last row removed.

In BTVW, a constrained key for a constraint $f \in \{0,1\}^\ell$ is of the form $\mathsf{ck} = (\Psi, \{\mathbf{c}_j\}_{j\in[L]})$, where $\Psi_i \leftarrow \mathsf{BTVW.GenP_{pp}}(\mathsf{msk}, \mathbf{e}_0, f_i)$, the string $(\psi_1, \ldots, \psi_L)$ is the binary representation of $\Psi = [\Psi_1 \mid \ldots \mid \Psi_\ell]$, and $\mathbf{c}_j \leftarrow \mathsf{BTVW.GenC_{pp}}(\mathsf{msk}, \mathbf{B}_j, \psi_j)$.

Let $\mathsf{ABE} = (\mathsf{ABE.Setup}, \mathsf{ABE.Enc}, \mathsf{ABE.Constrain}, \mathsf{ABE.Dec})$ be an attribute-based encryption scheme and let $\mathsf{BTVW} = (\mathsf{BTVW.KeyGen}, \mathsf{BTVW.Eval}, \mathsf{BTVW.Constrain}, \mathsf{BTVW.ConstrainEval})$ be the private constrained pseudorandom function scheme of [BTVW17]. Our succinct-key private constrained PRF with updatable parameters SCPRF consists of the following algorithms.

- $\mathsf{SCPRF.KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$: The input parameters are the security parameter λ, the maximum description length ℓ of constraint functions, their input length z and their maximum depth t.

  Let $(\mathsf{BTVW.msk}, \mathsf{BTVW.pp}) \leftarrow \mathsf{BTVW.KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$. Sample $\mathbf{e}_0$ and $\mathbf{B}_1, \ldots, \mathbf{B}_L$ as in the BTVW CPRF and then generate

$$\Psi_{i,\beta} \leftarrow \mathsf{BTVW.GenP_{BTVW.pp}}(\mathsf{BTVW.msk}, \mathbf{e}_0, \beta),$$
$$\mathbf{c}_{j,\beta} \leftarrow \mathsf{BTVW.GenC_{BTVW.pp}}(\mathsf{BTVW.msk}, \mathbf{B}_j, \beta)$$

  for all $i \in [\ell], j \in [L]$ and $\beta \in \{0,1\}$, where the parameter L is set as in $\mathsf{BTVW.KeyGen}$.

  Set up independently two ABE schemes as follows: $(\mathsf{ABE.msk_1}, \mathsf{ABE.pk_1}) \leftarrow \mathsf{ABE.Setup}(1^\lambda)$ and $(\mathsf{ABE.msk_2}, \mathsf{ABE.pk_2}) \leftarrow \mathsf{ABE.Setup}(1^\lambda)$. Then sample $d_1, \ldots, d_\ell \leftarrow \{0,1\}$ and compute

$$\mathbf{a}_{i,\beta} \leftarrow \mathsf{ABE.Enc}(\mathsf{ABE.pk_1}, (i,\beta), \Psi_{i,\beta\oplus d_i}),$$
$$\mathbf{b}_{j,\beta} \leftarrow \mathsf{ABE.Enc}(\mathsf{ABE.pk_2}, (j,\beta), \mathbf{c}_{j,\beta})$$

  for all $i \in [\ell], j \in [L], \beta \in \{0,1\}$. Output

$$\mathsf{msk} = \left(\mathsf{BTVW.msk}, \{d_i\}_{i\in[\ell]}, \mathsf{ABE.msk_1}, \mathsf{ABE.msk_2}\right),$$
$$\mathsf{pp} = \left(\mathsf{BTVW.pp}, \{\mathbf{a}_{i,\beta}\}_{i\in[\ell],\beta\in\{0,1\}}, \{\mathbf{b}_{j,\beta}\}_{j\in[L],\beta\in\{0,1\}}, \mathsf{ABE.pk_1}, \mathsf{ABE.pk_2}\right)$$

  as the master secret key and the public parameters, respectively.

- $\mathsf{SCPRF.Eval_{pp}}(\mathsf{msk}, x)$: Output $y = \mathsf{BTVW.Eval_{BTVW.pp}}(\mathsf{BTVW.msk}, x)$.

- $\mathsf{SCPRF.Constrain_{pp}}(\mathsf{msk}, f)$: Let $(f_1, \ldots, f_\ell)$ be the description of f and $u = (u_1, \ldots, u_\ell) = (f_1 \oplus d_1, \ldots, f_\ell \oplus d_\ell)$. Compute $\Psi_i \leftarrow \mathsf{ABE.Dec}(\mathsf{ABE.msk_1}, \mathbf{a}_{i,u_i})$ for all $i \in [\ell]$. Let $\psi_1, \ldots, \psi_L$ be the digits

of the binary decomposition of $\Psi = [\Psi_1 \mid \ldots \mid \Psi_\ell]$ and define the predicates

$$\Phi_u(i,\beta) = \begin{cases} 0, & \text{if } \beta = u_i \\ 1, & \text{otherwise,} \end{cases}$$

$$\Lambda_\Psi(j,\beta) = \begin{cases} 0, & \text{if } \beta = \psi_j \\ 1, & \text{otherwise.} \end{cases}$$

Let $\mathsf{ABE.ck}_1 \leftarrow \mathsf{ABE.Constrain}(\mathsf{ABE.msk}_1, \Phi_u)$ and $\mathsf{ABE.ck}_2 \leftarrow \mathsf{ABE.Constrain}(\mathsf{ABE.msk}_2, \Lambda_\Psi)$. Output $\mathsf{ck} = (\mathsf{ABE.ck}_1, \mathsf{ABE.ck}_2)$ as the constrained key and $\mathsf{pp}' = (\mathsf{pp}, u)$ as the new public parameters.

- $\mathsf{SCPRF.ConstrainEval}_{\mathsf{pp}'}(\mathsf{ck}, x)$: Retrieve u from the public parameters. For each $i \in [\ell]$, compute $\Psi_i = \mathsf{ABE.Dec}(\mathsf{ABE.ck}_1, \Phi_u, \mathbf{a}_{i,u_i})$. Observe that the ABE constrained key for Φ_u allows the decryption of $\mathbf{a}_{i,u_i}$ but not $\mathbf{a}_{i,1-u_i}$. Let $(\psi_1, \ldots, \psi_L)$ be the binary decomposition of $\Psi = [\Psi_1 | \ldots | \Psi_\ell]$. Compute $\mathbf{c}_j = \mathsf{ABE.Dec}(\mathsf{ABE.ck}_2, \Lambda_\Psi, \mathbf{b}_{j,\psi_j})$ for all $j \in [L]$.

  Output $y = \mathsf{BTVW.ConstrainEval}_{\mathsf{BTVW.pp}}((\Psi, \{\mathbf{c}_j\}_{j\in[L]}), x)$.

Below we state the main result of this section, which shows that this construction satisfies our definition of weak security. Moreover, its constrained keys are succinct. We consider SCPRF to be instantiated with the attribute-based encryption scheme of Boneh et al. [BGG$^+$14], which is selectively secure under the hardness of LWE, in addition to the private constrained PRF BTVW, which has single-key selective security under the LWE assumption.

Theorem 3.1. *Consider the* SCPRF *construction from Section 3 with the attribute-based encryption scheme of [BGG$^+$14] in place of* ABE*. Under the LWE assumption,* SCPRF *is a weak single-key selective private CPRF with updatable parameters in which the constrained keys are succinct.*

4 Succinct-key private constrained PRF from functional encodings

Let $\mathsf{CPRF} = (\mathsf{CPRF.KeyGen}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.ConstrainEval})$ be a private constrained PRF. We assume that the probabilistic algorithm $\mathsf{CPRF.Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$ can be split into two procedures: a probabilistic algorithm $\mathsf{CPRF.Sample}_{\mathsf{pp}}(\mathsf{msk})$, which generates all the randomness R necessary to generate a constrained key, independently of the constraint f, and a deterministic algorithm $\mathsf{CPRF.Comp}_{\mathsf{pp}}(\mathsf{msk}, f, R)$, which computes a constrained key corresponding to f using the randomness R. We require that generating a key with these algorithms is equivalent to generating it with the usual constraining algorithm, that is, the outputs $\mathsf{ck} \leftarrow \mathsf{CPRF.Comp}_{\mathsf{pp}}(\mathsf{msk}, f, \mathsf{CPRF.Sample}_{\mathsf{pp}}(\mathsf{msk}))$ and $\mathsf{ck} \leftarrow \mathsf{CPRF.Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$ are equally distributed.

Let $\mathsf{FE} = (\mathsf{FE.Enc}, \mathsf{FE.Open}, \mathsf{FE.Dec})$ be a functional encoding scheme and let $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be a symmetric-key encryption system. The following algorithms constitute our succinct-key private constrained PRF with updatable parameters SCPRF, which supports the same class of constraints as the underlying scheme CPRF.

- $\mathsf{SCPRF.KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$: The input parameters are the security parameter λ, the maximum description length ℓ of constraint functions, their input length z and their maximum depth t. To set up the scheme, generate $(\mathsf{CPRF.msk}, \mathsf{CPRF.pp}) \leftarrow \mathsf{CPRF.KeyGen}(1^\lambda, 1^\ell, 1^z, 1^t)$, $k \leftarrow \mathsf{SKE.Setup}(1^\lambda)$ and $R \leftarrow \mathsf{CPRF.Sample}_{\mathsf{CPRF.pp}}(\mathsf{CPRF.msk})$. Then compute $(\mathbf{C}, r) \leftarrow \mathsf{FE.Enc}(1^\lambda, (\mathsf{CPRF.msk}, k, R))$. Output

$$\mathsf{msk} = (\mathsf{CPRF.msk}, k, R, r)\,, \quad \mathsf{pp} = (\mathsf{CPRF.pp}, \mathbf{C})$$

  as the master secret key and the public parameters, respectively.

- $\mathsf{SCPRF.Eval}_{\mathsf{pp}}(\mathsf{msk}, x)$: Output $y = \mathsf{CPRF.Eval}_{\mathsf{CPRF.pp}}(\mathsf{CPRF.msk}, x)$.

- $\mathsf{SCPRF.Constrain}_{\mathsf{pp}}(\mathsf{msk}, f)$: Let $h \leftarrow \mathsf{SKE.Enc}(k, f)$. Consider the following circuit $\mathcal{C}_h$: on input a tuple (s, k, R), it computes $f = \mathsf{SKE.Dec}(k, h)$ and then outputs $\mathsf{CPRF.Comp}_{\mathsf{CPRF.pp}}(s, f, R)$. Compute $\mathbf{d} \leftarrow \mathsf{FE.Open}(\mathcal{C}_h, (\mathsf{CPRF.msk}, k, R), r)$ and output it as the constrained key for f. Output also $\mathsf{pp}' = (\mathsf{pp}, h)$ as the new public parameters.

  Below we consider our private constrained PRF with updatable parameters SCPRF from this section to be instantiated with the following schemes as building blocks:

  - The private constrained PRF of Brakerski, Tsabary, Vaikuntanathan, and Wee [BTVW17] in place of CPRF;
  - The functional encoding scheme of Wee and Wichs [WW21] in place of FE;
  - The Regev cryptosystem [Reg05] in place of SKE, with encryption and decryption performed bit by bit.

  Observe that the algorithm $\mathrm{BTVW.Constrain}$ (see [BTVW17]) can easily be split into a probabilistic part $\mathrm{BTVW.Sample}$ and a deterministic part $\mathrm{BTVW.Comp}$, as described next.

  - $\mathrm{BTVW.Sample}_{\mathrm{pp}}(\mathrm{msk})$: Sample vectors $\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_\ell \leftarrow \chi^{(n+1)\log q}$ and matrices $\mathbf{R}_1, \ldots, \mathbf{R}_\ell \leftarrow \{0,1\}^{(n+1)\log q \times (n+1)\log q}$. Output $R = (\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_\ell, \mathbf{R}_1, \ldots, \mathbf{R}_\ell)$.
  - $\mathrm{BTVW.Comp}_{\mathrm{pp}}(\mathrm{msk}, f, R)$: Compute

$$\Psi_i = \begin{pmatrix} \mathbf{B} \\ \mathbf{s}^\mathsf{T}\mathbf{B} + \mathbf{e}_0^\mathsf{T} \end{pmatrix} \mathbf{R}_i + f_i \mathbf{G}$$

    for all $i \in [\ell]$ and let $(\psi_1, \ldots, \psi_L)$ be the binary representation of $\Psi = [\Psi_1 \mid \ldots \mid \Psi_\ell]$. Compute also $\mathbf{c}_j^\mathsf{T} = \mathbf{s}^\mathsf{T}(\mathbf{B}_j - \psi_j \overline{\mathbf{G}}) + \mathbf{e}_j^\mathsf{T}$ for all $j \in [L]$ and output $\mathsf{ck} = (\Psi, \{\mathbf{c}_j\}_{j \in [L]})$.

- $\mathsf{SCPRF.ConstrainEval}_{\mathsf{pp}'}(\mathbf{d}, x)$: Retrieve h from the parameters and compute $\mathsf{ck} \leftarrow \mathsf{FE.Dec}(\mathcal{C}_h, \mathbf{C}, \mathbf{d})$. Output $y = \mathsf{CPRF.ConstrainEval}_{\mathsf{CPRF.pp}}(\mathsf{ck}, x)$.

Under the assumption of hardness of the LWE problem, the private constrained PRF BTVW has single-key selective security, the functional encoding scheme FE is 1-SIM secure, and SKE is IND-CPA secure. With these properties we can prove that our construction satisfies the definition of strong security and the property of succinct keys, as stated below.

Theorem 4.1. *Consider the* SCPRF *construction from this section instantiated with the private CPRF of [BTVW17], the FE scheme of [WW21] and the SKE scheme of [Reg05]. Under the LWE assumption,* SCPRF *is a strong single-key selective private CPRF with updatable parameters in which the constrained keys are succinct.*

References

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, pages 533–556, 2014.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2), 2012.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *PKC*, pages 501–519, 2014.

[BKM17] Dan Boneh, Sam Kim, and Hart Montgomery. Private puncturable PRFs from standard lattice assumptions. In *EUROCRYPT*, pages 415–445, 2017.

[BLW17] Dan Boneh, Kevin Lewi, and David J. Wu. Constraining pseudorandom functions privately. In *PKC*, 2017.

[BTVW17] Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained PRFs (and more) from LWE. In *TCC*, pages 264–302, 2017.

[BV15] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *TCC*, pages 1–30, 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT*, pages 280–300, 2013.

[CC17] Ran Canetti and Yilei Chen. Constraint-hiding constrained PRFs for NC^1 from LWE. In *EUROCRYPT*, pages 446–476, 2017.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49, 2013.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554, 2013.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *CCS*, pages 669–684, 2013.

[PS18] Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In *PKC*, pages 675–701, 2018.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 84–93, 2005.

[WW21] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. In *EUROCRYPT*, pages 127–156, 2021.