Adaptively Secure Constrained Pseudorandom Functions

Abstract. A constrained pseudo random function (PRF) behaves like a standard PRF, but with the added feature that the (master) secret key holder, having secret key K, can produce a constrained key, K_f, that allows for the evaluation of the PRF on a subset of the domain as determined by a predicate function f within some family $\mathcal{F}$. While previous constructions gave constrained PRFs for poly-sized circuits, all reductions for such functionality were based in the selective model of security where an attacker declares which point he is attacking before seeing any constrained keys.
In this paper we give new constrained PRF constructions for arbitrary circuits in the random oracle model based on indistinguishability obfuscation. Our solution is constructed from two recently emerged primitives: an adaptively secure Attribute-Based Encryption (ABE) for circuits and a Universal Sampler Scheme as introduced by Hofheinz et al. Both primitives are constructible from indistinguishability obfuscation ($i\mathcal{O}$) (and injective pseudorandom generators) with only polynomial loss.

1 Introduction

Constrained pseudorandom functions The concept of constrained pseudorandom functions (constrained PRFs) was proposed independently by Boneh and Waters [5], Boyle, Goldwasser and Ivan [7] and Kiayias et al. [23]. A constrained PRF behaves like a standard PRF [16], but with the added feature that the (master) secret key holder, having secret key K, can produce a constrained key, K_f, that allows for the evaluation of the PRF on a subset of the domain as determined by a predicate function f within some family $\mathcal{F}$. The security definition of a constrained PRF system allows for a poly-time attacker to query adaptively on several functions $f_1, \ldots, f_Q$ and receive constrained keys $K_{f_1}, \ldots, K_{f_Q}$. Later the attacker chooses a challenge point x^* such that $f_i(x^*) = 0\ \forall i$. The attacker should not be able to distinguish between the output of the PRF $F(K, x^*)$ and a randomly chosen value with better than negligible probability.

Constrained PRFs can hence be seen as PRFs in which the ability to evaluate the PRF can be delegated, using a constrained key. This feature has proved useful in various applications, e.g., broadcast encryption [5], multiparty key exchange [6] and the development of "punctured programming" techniques using obfuscation [28].

Ideally, we would like to have constrained PRFs that are as universally useful as possible. In particular, they should support as expressive constraints, and thus delegation capabilities, as possible. In their initial work, Boneh and Waters [5] gave a construction for building constrained PRFs for polynomial sized circuits

(with a priori fixed depth) based on multilinear encodings [13, 10]. Furthermore, they demonstrated the power of constrained PRFs with several motivating applications.

For instance, one application (detailed in [5]) is a (secret encryption key) broadcast key encapsulation mechanism with "optimal size ciphertexts", where the ciphertext consists solely of a header describing the recipient list S. The main idea is that the key assigned to a set S is simply the PRF evaluated on S as $F(K, S)$. A user i in the system is assigned a key for a function $f_i(\cdot)$, where $f_i(S) = 1$ if and only if $i \in S$. Other natural applications given include identity-based key exchange and a form of non-interactive policy-based key distribution. Later Sahai and Waters [28] showed the utility of (a limited form of) constrained PRFs in building cryptography from indistinguishability obfuscation and Boneh and Zhandry [6] used them (along with obfuscation) in constructing recipient private broadcast encryption.

Focus: adaptive security While the functionality of the Boneh-Waters construction was expressive, their proof reduction was limited to selective security where the challenge point x^* is declared by the attacker before it makes any queries. For many applications of constrained PRFs, achieving adaptive security requires an underlying adaptively secure constrained PRF. In particular, this applies to the optimal size broadcast, policy-based encryption, non-interactive key exchange and recipient-private broadcast constructions mentioned above.

In this work we are interested in exploring adaptive security in constrained PRFs. Hence, we are interested in the question

> *Is there an adaptively secure constrained PRF for expressive families of constraints? Specifically, is there an adaptively secure constrained PRF for the family of poly-sized circuits?*

Any selectively secure constrained PRF can be proven adaptively secure if one is willing to use a technique called complexity leveraging (as used, e.g., in the context of IBE schemes [4]). This technique, however, leads to a reduction with superpolynomial loss (which leads to a significant quantitative loss in security), and thus it can be desirable to look for alternative ways to achieve adaptive security. Hence, here we are interested in polynomial-time reductions, and thus in avoiding complexity leveraging.

Up until now, constrained PRF constructions that achieve adaptive security have relatively limited functionality. Hohenberger, Koppula, and Waters [21] show how to build adaptive security from indistinguishability obfuscation for a special type of constrained PRFs called puncturable PRF. In a puncturable PRF system the attacker is allowed to make several point queries adaptively, before choosing a challenge point x^* and receiving a key that allows for evaluation at all points $x \neq x^*$. While their work presents progress in this area, there is a large functionality gap between the family of all poly-sized circuits and puncturing-type functions. Fuchsbauer et al. [12] give a subexponential reduction to obfuscation for a larger class of "prefix-type" circuits, however, their reduction is still super polynomial. In addition, they give evidence that the problem of

achieving full security with polynomial reductions might be difficult. They adapt the proof of [24] to show a black box impossibility result for a certain class of "fingerprinting" constructions that include the original Boneh-Waters [5] scheme.

The difficulty of achieving adaptive security (and why we utilize the random oracle model) In order to describe the technical problem that arises with adaptively secure constrained PRFs, say that we want to construct a bit-fixing constrained PRF F, i.e., one that allows for constrained keys $K_{f_{\overline{x}}}$ for "bit-matching" predicates of the form $f_{\overline{x}}(x) = 1 \Leftrightarrow \forall i : x_i = \overline{x}_i \vee \overline{x}_i = \perp$ with $\overline{x} = (\overline{x}_i)_{i=1}^n \in (\{0,1\} \cup \{\perp\})^n$. An adversary A on F may first ask for polynomially many constrained keys $K_{f_{\overline{x}}}$, and then gets challenged on a preimage x^*. The goal of a successful simulation is to be able to prepare all $K_{f_{\overline{x}}}$, but *not* to be able to compute $F(K, x^*)$.

Now if $x^* = (x_i^*)_{i=1}^n$ is known in advance, then the simulation can set up the function $F(K, \cdot)$ in an "all-but-one" way, such that all images except $F(K, x^*)$ can be computed. For instance, the selective-security simulation from [5] sets up

$$F(K, x) = e(g, \ldots, g)^{\prod_{i=1}^n \alpha_{i,x_i}} \qquad (\text{for } K = (\alpha_{i,b})_{i,b}), \tag{1}$$

where e is an $(n-1)$-linear map, and the simulation knows all $\alpha_{i,1-x_i^*}$ (while the α_{i,x_i^*} are only known "in the exponent," as $g^{\alpha_{i,x_i^*}}$). This setup not only allows to compute $F(K, x)$ as soon as there is an i with $x_i \neq x_i^*$ (such that the corresponding $\alpha_{i,x_i} = \alpha_{i,1-x_i^*}$ is known); also, assuming a graded multilinear map, evaluation can be *delegated*. (For instance, a constrained key that allows to evaluate all inputs with $x_1 = 1$ would contain $\alpha_{1,1}$ and $g^{\alpha_{i,b}}$ for all other i, b.)

However, observe now what happens when A chooses the challenge preimage x^* only *after* asking for constrained keys. Then, the simulation may be forced to commit to the full function $F(K, \cdot)$ (information-theoretically) before even knowing where "not to be able to evaluate." For instance, for the constrained PRF from [5] sketched above, already a few suitably chosen constrained keys (for predicates f_i) fully determine $F(K, \cdot)$, while the corresponding predicates f_i leave exponentially many potential challenge preimages x^* uncovered. If we assume that the simulation either can or cannot evaluate $F(K, x)$ on a given preimage x (at least once $F(K, \cdot)$ is fully determined), we have the following dilemma. Let $\mathcal{C}$ be the set of preimages that the simulation *cannot* evaluate. If $\mathcal{C}$ is too small, then $x^* \in \mathcal{C}$ will not happen sufficiently often, so that the simulation cannot learn anything from A. But if $\mathcal{C}$ is too large, then the simulation will not be able to construct "sufficiently general" constrained keys for A (because the corresponding predicates f would evaluate to 1 on some elements of $\mathcal{C}$).[1]

This argument eliminates not only guessing x^* (at least when aiming at a polynomial reduction), but also the popular class of "partitioning arguments". (Namely, while guessing x^* corresponds to $|\mathcal{C}| = 1$ above, partitioning arguments

[1] In fact, for many classes of allowed constraining predicates, A can easily ask for constrained keys that, taken together, allow to evaluate $F(K, \cdot)$ *everywhere* except on x^*. For instance, in our case, A could ask for all keys K_{f_i} with $f_i(x) = 1 \Leftrightarrow x_i = 1 - x_i^*$. Hence, in this case, the simulation *must* fail already whenever $|\mathcal{C}| \geq 2$.

consider larger sets $\mathcal{C}$. However, the argument above excludes sets $\mathcal{C}$ of *any* size for relevant classes of constraining predicates and superpolynomial preimage space.) In particular, since the selectively-secure constrained PRFs from [23, 5, 7] fulfill the assumptions of the argument, it seems hopeless to prove them fully secure, at least for standard preimage sizes.

Hence, to obtain adaptively secure constrained PRFs, we feel that leaving the standard model of computation is unavoidable, and so we utilize the random oracle for our security analysis.

The random oracle model versus the random oracle heuristic When attempting to instantiate a scheme described and proven in the random oracle model the most common method is to apply the heuristic [3] of replacing oracle calls with an evaluation of a hash function such as SHA-256. This heuristic has been (apparently) successful for a number of deployed cryptographic schemes (e.g., [22, 29]), but on the other hand there are well documented [9] issues with this heuristic.

While the random oracle heuristic is tightly associated with the random oracle model, we wish to emphasize that there are potentially other avenues to instantiate the model. In particular, one could try to realize a random oracle like object via specialized and limited trusted hardware or a distributed consensus protocol such as a blockchain. It could even be the case that an existing blockchain could be obliviously leveraged for such a functionality in a similar vein to the work Goyal and Goyal [18] for one time programs.

Our Contributions In this paper, we give an affirmative answer to the question above. That is, we present the first constrained PRF constructions for poly-sized circuits [2] that have polynomial reductions to indistinguishability obfuscation in the random oracle model. While our construction does use heavy tools such as indistinguishability obfuscation, and our proof involves the random oracle heuristic, we wish to emphasize that our solution is currently the only known one for this problem. Moreover, prior to our work, it was not clear if it is possible to construct adaptively secure constrained PRFs even under these strong assumptions. Recent results [1] have shown that for certain problems (such as simulation secure functional encryption), it is impossible to get the desired security guarantees, even assuming the existence of indistinguishability obfuscation and the random oracle heuristic.

Ingredients used in our Construction Our solution is constructed from two recently emerged primitives: an adaptively secure Attribute-Based Encryption (ABE) [27] for circuits and Universal Samplers as introduced by Hofheinz et al. [19]. Both primitives are constructible from indistinguishability obfuscation ($i\mathcal{O}$) (and injective pseudorandom generators) with only polynomial loss. Waters [31] recently gave an adaptively secure construction of ABE[3] based on indis-

[2] More specifically, we present a construction for polynomial-sized circuits of any apriori bounded depth.

[3] The construction is actually for Functional Encryption which implies ABE.

tinguishability obfuscation and Hofheinz et al. [19] showed how to build Universal Samplers from $i\mathcal{O}$ in the random oracle model — emphasizing that the random oracle heuristic is applied outside the obfuscated program.

Before we describe our construction we briefly overview the two underlying primitives. An ABE scheme (for circuits) has four algorithms. A setup algorithm $\mathsf{ABE.setup}(1^\lambda)$ that outputs public parameters $\mathsf{pk}_{\mathsf{ABE}}$, and a master secret key $\mathsf{msk}_{\mathsf{ABE}}$. The encryption algorithm $\mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x)$ takes in the public parameters, message t, and "attribute" string x and outputs a ciphertext c. A key generation algorithm $\mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ outputs a secret key given a boolean circuit C. Finally, the decryption algorithm $\mathsf{ABE.dec}(\mathrm{SK}, c)$ will decrypt an ABE ciphertext encrypted under attribute x iff $C(x) = 1$, where C is the circuit associated with the secret key.

The second primitive is a universal sampler scheme. Intuitively, a universal sampler scheme behaves somewhat like a random oracle except it can sample from arbitrary distributions as opposed to just uniformly random strings. More concretely, a universal sampler scheme consists of two algorithms, $\mathsf{US.setup}$ and $\mathsf{US.sample}$. In a set-up phase, $U \leftarrow \mathsf{US.setup}(1^\lambda)$ will take as input a security parameter and output "sampler parameters" U. We can use these parameters to "obliviously" sample from a distribution specified by a circuit d, in the following sense. If we call $\mathsf{US.sample}(U, d)$ the scheme will output $d(z)$ for hidden random coins z that are pseudorandomly derived from U and d.

Security requires that in the random oracle model, $\mathsf{US.setup}$ outputs images that look like independently and honestly generated d-samples, in the following sense. We require that an efficient simulator can simulate U and the random oracle such that the output of $\mathsf{US.sample}$ on arbitrarily many adversarially chosen inputs d_i coincides with independently and honestly chosen images $d_i(z_i)$ (for truly random z_i that are hidden even from the simulator). Of course, the simulated U and the programmed random oracle must be computationally indistinguishable from the real setting.

Our Solution in a Nutshell We now describe our construction that shows how to build constrained PRFs from adaptively secure ABE and universal samplers. One remarkable feature is the simplicity of our construction once the underlying building blocks are in place.

The constrained PRF key is setup by first running $U \leftarrow \mathsf{US.setup}(1^\lambda)$ and $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$. The master PRF key K is $(U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}))$. To define the PRF evaluation on input x we let $d_{\mathsf{pk}_{\mathsf{ABE}},x}(z = (t, r))$ be a circuit in some canonical form that takes as input a pair $z = (t, r)$ and computes $\mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x; r)$. Here we view $\mathsf{pk}_{\mathsf{ABE}}, x$ as constants hardwired into the circuit d and t, r as the inputs, where we make the random coins of the encryption algorithm explicit. To evaluate the PRF $F(K, x)$ we first compute $c_x = \mathsf{US.sample}(U, d_{\mathsf{pk}_{\mathsf{ABE}},x})$. Then we compute and output $\mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, c_x)$[4].

[4] We use the convention that the master secret key can decrypt all honestly generated ABE ciphertexts. Alternatively, one could just generate a secret key for a circuit that always outputs 1 and use this to decrypt.

Essentially, the evaluation function on input x first uses the universal sampler to encrypt an ABE ciphertext under attribute x for a randomly chosen message t. Then it uses the master secret key to decrypt the ciphertext which gives t as the output.

To generate a constrained key for circuits C, the master key holder simply runs the ABE key generation to compute $\mathsf{sk}_C = \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ and sets the constrained key to be $K\{C\} = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C))$. Evaluation can be done using $K\{C\}$ on input x where $C(x) = 1$. Simply compute c_x from the sampler parameters U as above, but then use sk_C to decrypt. The output will be consistent with the master key evaluation.

The security argument is organized as follows. We first introduce a hybrid game where the calls to the universal sampler scheme are answered by a sampling oracle that generates a fresh sample every time it is called. The security definition of universal samplers schemes argues (in the random oracle model) that the attacker's advantage in this game must be negligibly close to the original advantage. Furthermore, any polynomial time attacker will cause this samples oracle to be called at most some polynomial Q number of times. One of these calls must correspond to the eventual challenge input x^*.

We can now reduce to the security of the underlying ABE scheme. First the reduction guesses with $1/Q$ success probability which samples oracle call will correspond to x^* and embed an ABE challenge ciphertext here. An attacker on the constrained PRF scheme now maps straightforwardly to an ABE attacker.

Future Directions A clear future direction is to attempt to achieve greater functionality in the standard model. There is a significant gap between our random oracle model results of constrained PRFs for all circuits and the standard model results of Hohenberger, Koppula, and Waters for puncturable PRFs [21]. It would be interesting to understand if there are fundamental limitations to achieving such results. Fuchsbauer et al. [12] give some initial steps to negative results, however, it is unclear if they generalize to larger classes of constructions.

Other Related Work Attribute-Based Encryption for circuits was first achieved independently by Garg, Gentry, Halevi, Sahai and Waters [14] from multilinear maps and by Gorbunov, Vaikuntanathan and Wee [17] from the learning with errors [26] assumption. Both works were proven selectively secure; requiring complexity leveraging for adaptive security. In two recent works, Waters [31] and Garg, Gentry, Halevi and Zhandry [15] achieve adaptively secure ABE for circuits under different cryptographic assumptions. We also note that Boneh and Zhandry [6] show how to use indistinguishability obfuscation for circuits and punctured PRFs to create constrained PRFs for circuits. This construction is limited though to either selective security or utilizing complexity leveraging.

Brakerski and Vaikunthanathan [8] showed a constrained PRF construction that is secure against single query attackers based on the LWE assumption. However, our construction and motivating applications are concerned with the case of multiple queries or collusions.

1.1 Discussion of our Assumptions

Our construction uses "heavyweight" tools (i.e., indistinguishability obfuscation and random oracles) for a problem that can also be solved in a much simpler way with complexity leveraging. In this section, we would like to argue the benefits of a "more structured" solution like ours. Specifically, while an ideal or "last word" solution would be given under better assumptions, we feel that our work makes interesting progress that sets a bar for the future work to try to overcome.

One obvious way to relax the required assumptions for our work would be to only build on *one* heavyweight assumption (instead of two, as we do). In other words, this would mean to remove the random oracle or work under standard assumptions with a random oracle. While we definitely agree this would be an improvement, either one of these appears to require completely new techniques. For instance, consider the task of achieving expressive constrained PRFs from standard assumptions (that is, assumptions not based on indistinguishability obfuscation/multilinear maps). Currently, there are no known collusion-resistant constrained PRFs from standard assumptions. Achieving such a constrained PRF would be highly surprising even if it were selectively (not adaptively secure), used subexponential assumptions and the random oracle model.

On the other hand, consider the problem of achieving adaptive security from indistinguishability obfuscation alone, avoiding random oracles. The most similar problem to this is achieving adaptive security in Attribute-Based Encryption schemes. All such solutions (starting with [30]) in this regime have used dual system encryption (or similar) techniques. With those techniques, the simulation maintains and manipulates a special relationship between the private keys and challenge ciphertext. This lets one circumvent impossibilities and lower bounds such as the ones from [11, 20, 24]. In constrained PRFs there are no challenge ciphertexts (only an input point) so the only techniques we know do not apply. Indeed, our contribution, which uses random oracles (in a rather nontrivial way), proposes some approach to this problem. We think our work helps make the challenge clear to the community.

A fine point here is that known approaches to proving indistinguishability obfuscation from non-interactive assumptions seem to already imply some form of complexity leveraging or sub-exponential hardness. So given that we can get adaptive security with sub-exponential hardness anyway, why should our approach help? While we understand this argument, we think it can be misleading. For example, [31] gave an adaptively secure functional encryption scheme from indistinguishability obfuscation where one could have given the exact same rationale. In fact, later [2] built upon these ideas to give a generic selective to adaptive FE conversion where subexponential hardness is not inherent. For our case, we currently do not have such a next step, but it is well possible some future work could find it. As a starting point, [25, Section 1.5] provide an interesting discussion about how in the future one might avoid the subexponential barrier in indistinguishability obfuscation for certain cases.

2 Preliminaries

2.1 Notations

Let $x \leftarrow \mathcal{X}$ denote a uniformly random element drawn from the set $\mathcal{X}$. Given integers $\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}}$, let $\mathcal{C}[\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}}]$ denote the set of circuits that can be represented using ℓ_{ckt} bits, take ℓ_{inp} bits input and output ℓ_{out} bits.

2.2 Constrained Pseudorandom Functions

The notion of constrained pseudorandom functions was introduced in the concurrent works of [5, 7, 23]. Let $\mathcal{K}$ denote the key space, $\mathcal{X}$ the input domain and $\mathcal{Y}$ the range space. A PRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is said to be *constrained* with respect to a boolean circuit family $\mathcal{F}$ if there is an additional key space $\mathcal{K}_c$, and three algorithms $F.\mathsf{setup}$, $F.\mathsf{constrain}$ and $F.\mathsf{eval}$ as follows:

- $F.\mathsf{setup}(1^\lambda)$ is a PPT algorithm that takes the security parameter λ as input and outputs a key $K \in \mathcal{K}$.
- $F.\mathsf{constrain}(K, C)$ is a PPT algorithm that takes as input a PRF key $K \in \mathcal{K}$ and a circuit $C \in \mathcal{F}$ and outputs a constrained key $K\{C\} \in \mathcal{K}_c$.
- $F.\mathsf{eval}(K\{C\}, x)$ is a deterministic polynomial time algorithm that takes as input a constrained key $K\{C\} \in \mathcal{K}_c$ and $x \in \mathcal{X}$ and outputs an element $y \in \mathcal{Y}$. Let $K\{C\}$ be the output of $F.\mathsf{constrain}(K, C)$. For correctness, we require that for all security parameters $\lambda \in \mathbb{N}$, keys $K \leftarrow F.\mathsf{setup}(1^\lambda)$, circuit $C \in \mathcal{F}$, $K\{C\} \leftarrow F.\mathsf{constrain}(K, C)$ and $x \in \mathcal{X}$,

$$F.\mathsf{eval}(K\{C\}, x) = F(K, x) \text{ if } C(x) = 1.$$

Security of Constrained Pseudorandom Functions Intuitively, we require that even after obtaining several constrained keys, no polynomial time adversary can distinguish a truly random string from the PRF evaluation at a point not accepted by the queried circuits. This intuition can be formalized by the following security game between a challenger and an adversary Att.

Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a constrained PRF with respect to a circuit family $\mathcal{F}$. The security game consists of three phases.

Setup Phase The challenger chooses a random key $K \leftarrow \mathcal{K}$ and a random bit $b \leftarrow \{0, 1\}$.

Query Phase In this phase, Att is allowed to ask for the following queries:

- **Evaluation Query** Att sends $x \in \mathcal{X}$, and receives $F(K, x)$.
- **Key Query** Att sends a circuit $C \in \mathcal{F}$, and receives $F.\mathsf{constrain}(K, C)$.
- **Challenge Query** Att sends $x \in \mathcal{X}$ as a challenge query. If $b = 0$, the challenger outputs $F(K, x)$. Else, the challenger outputs a random element $y \leftarrow \mathcal{Y}$.

Guess Att outputs a guess b' of b.

Let $E \subset \mathcal{X}$ be the set of evaluation queries, $L \subset \mathcal{F}$ be the set of constrained key queries and $Z \subset \mathcal{X}$ the set of challenge queries. The attacker Att wins if $b = b'$ and $E \cap Z = \phi$ and for all $C \in L, z \in Z, C(z) = 0$. The advantage of Att is defined to be $\mathsf{Adv}^{F}_{\mathsf{Att}}(\lambda) = \left| \Pr[\mathsf{Att} \text{ wins}] - 1/2 \right|$.

Definition 1. *The PRF F is a secure constrained PRF with respect to $\mathcal{F}$ if for all PPT adversaries* Att, $\mathsf{Adv}^{F}_{\mathsf{Att}}(\lambda)$ *is negligible in* λ.

In the above definition the challenge query oracle may be queried multiple times on different points, and either all the challenge responses are correct PRF evaluations or they are all random points. As argued in [5], such a definition is equivalent (via a hybrid argument) to a definition where the adversary may only submit one challenge query. For our proofs, we will use the single challenge point security definition.

Another simplification that we will use in our proofs is with respect to the evaluation queries. Note that since we are considering constrained PRFs for circuits, without loss of generality, we can assume that the attacker queries for only constrained key queries. This is because any query for evaluation at input x can be replaced by a constrained key query for a circuit C_x that accepts only x.

2.3 Universal Samplers and Attribute Based Encryption

Due to space constraints, the definitions of universal samplers and attribute based encryption are given in the full version of the paper.

3 Adaptively Secure Constrained PRF

In this section, we will describe our constrained pseudorandom function scheme for circuit class $\mathcal{F}$. Let $n = n(\lambda), \ell_{\text{rnd}} = \ell_{\text{rnd}}(\lambda)$ be polynomials in λ, and let ℓ_{ckt} be a polynomial (to be defined in the construction below). We will use an adaptively secure ABE scheme ($\mathsf{ABE.setup}$, $\mathsf{ABE.keygen}$, $\mathsf{ABE.enc}$, $\mathsf{ABE.dec}$) for a circuit family $\mathcal{F}$ with message and attribute space $\{0,1\}^n$. Let us assume the encryption algorithm $\mathsf{ABE.enc}$ uses ℓ_{rnd} bits of randomness to compute the ciphertext. We will also use an $(\ell_{\text{ckt}}, \ell_{\text{inp}} = n + \ell_{\text{rnd}}, \ell_{\text{out}} = n)$ universal sampler scheme $\mathcal{U} = (\mathsf{US.setup}, \mathsf{US.sample})$.

The PRF $F : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, along with algorithms $F.\mathsf{setup}$, $F.\mathsf{constrain}$ and $F.\mathsf{eval}$ are described as follows.

$F.\mathsf{setup}(1^\lambda)$ The setup algorithm computes the sampler parameters $U \leftarrow \mathsf{US.setup}(1^\lambda)$ and $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$. In order to define F, we will first define a program $\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}$ (see Figure 1).

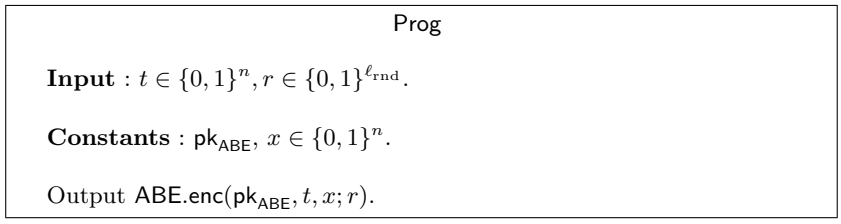

Fig. 1. Program used by setup algorithm: Prog

Let $\mathcal{C}$-$\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}$ be an $\ell_{\text{ckt}} = \ell_{\text{ckt}}(\lambda)$ bit canonical description of $\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}$, [5] where the last n bits of the representation are x, and let $\mathcal{C}$-$\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}$ be $\mathcal{C}$-$\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}$ without the last n bits; that is, $\forall x \in \{0,1\}^n, \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}$.
The PRF key K is set to be $(U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$. To compute $F(K, x)$, the setup algorithm first 'samples' a ciphertext $c = \mathsf{US.sample}(U, \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x)$ and output $\mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, c)$.

$F.\mathsf{constrain}(K = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}), C)$*:* The constrain algorithm first computes an ABE secret key corresponding to circuit C. It computes an ABE secret key $\mathsf{sk}_C = \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ and sets the constrained key to be $K\{C\} = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$.

$F.\mathsf{eval}(K\{C\} = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}), x)$*:* The evaluation algorithm first computes the canonical circuit $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\} = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x$. Next, it computes $c = \mathsf{US.sample}(U, \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\})$. Finally, it outputs $\mathsf{ABE.dec}(\mathsf{sk}_C, c)$.

Correctness Consider any key $K = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$ output by $F.\mathsf{setup}(1^\lambda)$. Let $C \in \mathcal{F}$ be any circuit, and let $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$, $K\{C\} = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$. Let x be any input such that $C(x) = 1$. We require that $F.\mathsf{eval}(K\{C\}, x) = F(K, x)$.

$$
\begin{aligned}
& F.\mathsf{eval}(K\{C\}, x) \\
& = \mathsf{ABE.dec}(\mathsf{sk}_C, \mathsf{US.sample}(U, \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\})) \\
& = \mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, \mathsf{US.sample}(U, \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}, x\}))^{6} \\
& = F(K, x)
\end{aligned}
$$

[5] Note that the value ℓ_{ckt} required by the universal sampler scheme is determined by the ABE scheme. It depends on the size of the encryption circuit $\mathsf{ABE.enc}$ and the length of $\mathsf{pk}_{\mathsf{ABE}}$.

[6] Recall $\mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x))$ outputs m, and so does $\mathsf{ABE.dec}(\mathsf{sk}_C, \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x))$ if $C(x) = 1$.

4 Proof of Security

In this section, we will prove adaptive security for our constrained PRF in the random oracle model. The construction uses adaptively secure universal samplers, which in turn uses an appropriate hash function which is modeled as a random oracle in the ROM. We assume the random oracle outputs ℓ_{RO} bit strings as output. We will first define a sequence of hybrid experiments, and then show that if any PPT adversary Att has non-negligible advantage in one experiment, then it has non-negligible advantage in the next experiment. Game 0 is the constrained PRF adaptive security game in the random oracle model. In Game 1, the challenger simulates the sampler parameters and the random oracle queries. It also implements a Samples Oracle O which is used for this simulation. Let q_{par} denote the number of queries to O during the Setup, Pre-Challenge and Challenge phases. In the next game, the challenger guesses which samples oracle query corresponds to the challenge input. Finally, in the last game, it modifies the output of the samples oracle on challenge input.

4.1 Sequence of Games

Game 0 In this experiment, the challenger chooses PRF key K. It receives random oracle queries and constrained key queries from the adversary Att. On receiving the challenge input x^*, it outputs either $F(K, x^*)$ or a truly random string. The adversary then sends post-challenge random oracle/constrained key queries, and finally outputs a bit b'.

1. **Setup Phase** Choose $U \leftarrow \mathsf{US.setup}(1^\lambda)$, $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$. Let $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}$ be the canonical circuit as defined in the construction.
2. **Pre Challenge Phase**
   - **Constrained Key Queries:** For every constrained key query C, compute $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$.
     Send $(U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$ to Att.
   - **Random Oracle Queries:** For each random oracle query y_i, check if y_i has already been queried.
     If yes, let (y_i, α_i) be the tuple corresponding to y_i. Send α_i to Att.
     If not, choose $\alpha_i \leftarrow \{0,1\}^{\ell_{\mathsf{RO}}}$, send α_i to Att and add (y_i, α_i) to table.
3. **Challenge Phase** On receiving challenge input x^*, set $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$. Compute $c = \mathsf{US.sample}(U, d^*)$, $t_0 = \mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, c)$.
   Choose $b \leftarrow \{0,1\}$. If $b = 0$, send t_0 to Att. Else send $t_1 \leftarrow \{0,1\}^n$.
4. **Post Challenge Phase** Respond to constrained key and random oracle queries as in pre-challenge phase.
5. **Guess** Att outputs a bit b'.

Game 1 This game is similar to the previous one, except that the sampler parameters U and responses to random oracle queries are simulated. The challenger implements a Samples Oracle O, and O is used for simulating U and the random oracle. Also, instead of using $\mathsf{US.sample}$ to compute $F(K, x^*)$, the challenger

uses the samples oracle O. Please note that even though O is defined during the Setup Phase, it is used in all the remaining phases.

1. **Setup Phase** Choose $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$. Let $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}$ be the canonical circuit as defined in the construction.
   Implement the Samples Oracle O as follows:
   - Implement a table T. Initially T is empty.
   - For each query $d \in \mathcal{C}[\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}}]$(recall $\mathcal{C}[\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}}]$ is the family of circuits whose bit representation is of length ℓ_{ckt}, takes input of length ℓ_{inp} and provides output of length ℓ_{out}) ,
     - If $\exists$ an entry of the form (d, α, β), output α.
     - Else if d is of the form $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x$,
       choose $t \leftarrow \{0,1\}^n$, $r \leftarrow \{0,1\}^{\ell_{\text{rnd}}}$.
       Output $c = \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x; r)$.
       Add (d, c, t) to T.
     - Else, choose $t \leftarrow \{0,1\}^{\ell_{\text{inp}}}$, compute $\alpha = d(t)$.
       Add $(d, \alpha, \perp)$ to T and output α.

   Choose $(U, \tau) \leftarrow \mathsf{SimUGen}(1^\lambda)$.
2. **Pre Challenge Phase**
   - **Constrained Key Queries:** For every constrained key query C, compute $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$.
     Send $(U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$ to Att.
   - **Random Oracle Queries:** For each random oracle query y_i, output $\mathsf{SimRO}(\tau, y_i)$ (recall SimRO can make polynomially many calls to Samples Oracle O).
3. **Challenge Phase** On receiving challenge input x^*, set $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$.
   If T does not contain an entry of the form (d^*, α, β),
   Query the Samples Oracle O with input d^*.
   Let (d^*, α, β) be the entry in T corresponding to d^*.
   Set $t_0 = \mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, O(d^*)) = \beta$ [7].
   Choose $b \leftarrow \{0,1\}$. If $b = 0$, send t_0 to Att. Else send $t_1 \leftarrow \{0,1\}^n$.
4. **Post Challenge Phase** Respond to constrained key and random oracle queries as in pre-challenge phase.
5. **Guess** Att outputs a bit b'.

Game 2 In this game, the challenger 'guesses' the samples oracle query which will correspond to the challenge input. The attacker wins if this guess is correct, or if the challenge input has not been queried before. Recall q_{par} denotes the number of calls to the Samples Oracle O during the Setup, Pre-Challenge and Challenge phases.

1. **Setup Phase** Choose $i^* \leftarrow [q_{\text{par}}]$. Remaining experiment is same as in Game 1.

[7] Recall $O(d^*) = \alpha$, and $\mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, \alpha) = \beta$.

Game 3 The only difference between this game and the previous one is in the behavior of the Sample Oracle on the $(i^*)^{\text{th}}$ query. Suppose the $(i^*)^{\text{th}}$ input is of the form $d^* = \mathcal{C}\text{-Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$. In the previous game, the entry in table T corresponding to d^* is of the form (d^*, α^*, β^*) where α^* is an encryption of β^* for attribute x^* using public key $\mathsf{pk}_{\mathsf{ABE}}$. In this game, the entry corresponding to d^* is (d^*, α^*, β^*), where α^* is the encryption of a random message for attribute x^* using $\mathsf{pk}_{\mathsf{ABE}}$.

1. **Setup Phase** Choose $i^* \leftarrow [q_{\text{par}}]$.
   Choose $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$. Let $\mathcal{C}\text{-Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}$ be the canonical circuit as defined in the construction. Implement the Samples Oracle O as follows:
   - Implement a table T. Initially T is empty.
   - For each query $d \in \mathcal{C}[\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}}]$,
     - If there exists an entry of the form (d, α, β), output α.
     - Else if d is of the form $\mathcal{C}\text{-Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x$ for some x, choose $t, \tilde{t} \leftarrow \{0,1\}^n$, $r \leftarrow \{0,1\}^{\ell_{\text{rnd}}}$.
       If d is not the $(i^*)^{\text{th}}$ unique query,
       output $c \leftarrow \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x; r)$, add (d, c, t) to T.
       Else set $c \leftarrow \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, \tilde{t}, x; r)$, add (d, c, t).
     - Else, choose $t \leftarrow \{0,1\}^{\ell_{\text{inp}}}$, compute $\alpha = d(t)$. Add $(d, \alpha, \perp)$ to T and output α.

   Choose $(U, \tau) \leftarrow \mathsf{SimUGen}(1^\lambda)$.
2. Remaining experiment is same as in Game 2.

4.2 Analysis

For any PPT adversary Att, let $\mathsf{Adv}^i_{\mathsf{Att}}$ denote the advantage of Att in Game i.

Claim 1 *Assuming $\mathcal{U} = (\mathsf{US.setup}, \mathsf{US.sample})$ is a secure $(\ell_{\text{ckt}}, \ell_{\text{inp}}, \ell_{\text{out}})$ universal sampler scheme, for any PPT adversary* Att*,*

$$\left|\mathsf{Adv}^0_{\mathsf{Att}} - \mathsf{Adv}^1_{\mathsf{Att}}\right| \leq \mathit{negl}(\lambda).$$

Proof. Suppose there exists a PPT adversary Att such that $\left|\mathsf{Adv}^0_{\mathsf{Att}} - \mathsf{Adv}^1_{\mathsf{Att}}\right| = \epsilon$. For any SimUGen, SimRO, we will construct a PPT algorithm $\mathcal{B}$ such that

$$\left|\Pr[\mathsf{Real}^{\mathcal{B}}(1^\lambda) = 1] - \Pr[\mathsf{Ideal}^{\mathcal{B}}_{\mathsf{SimUGen},\mathsf{SimRO}}(1^\lambda) = 1]\right| = \epsilon.$$

$\mathcal{B}$ interacts with Att and participates in either the Real or Ideal game. It receives the sampler parameters U. It chooses $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$.

During the pre-challenge phase, $\mathcal{B}$ receives either secret key queries or random oracle queries. On receiving secret key query for circuit $\mathcal{C}$, it computes $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ and sends $K\{C\} = (U, (\mathsf{pk}_{\mathsf{ABE}}, \mathsf{sk}_C), \mathcal{C}\text{-Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$ to Att. On receiving random oracle query y, it forwards it to the universal sampler challenger. It receives response α, which it forwards to Att.

On receiving the challenge message x^*, it sets d^* to be the circuit $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$, computes $c = \mathsf{US.sample}(U, d^*)$, $t_0 = \mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, c)$. It chooses $b \leftarrow \{0,1\}$. If $b = 0$, it sends t_0, else it sends $t_1 \leftarrow \{0,1\}^n$.

The post challenge queries are handled similar to the pre challenge queries. Finally, Att outputs b'. If $b = b'$, $\mathcal{B}$ send 0 to the universal sampler challenger, indicating Real experiment. Else it sends 1.

Note that due to the honest sample violation probability being 0, Att participates in either Game 0 or Game 1. This concludes our proof.

Observation 1 *For any adversary* Att, $\mathsf{Adv}^2_{\mathsf{Att}} \geq \frac{\mathsf{Adv}^1_{\mathsf{Att}}}{q_{\mathrm{par}}}$.

Proof. Since the challenger's choice i^* is independent of Att, if $d = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$ was queried before the challenge phase, then the challenger's guess is correct with probability $1/q_{\mathrm{par}}$.

Claim 2 *Assuming* $\mathsf{ABE} = (\mathsf{ABE.setup}, \mathsf{ABE.keygen}, \mathsf{ABE.enc}, \mathsf{ABE.dec})$ *is an adaptively secure attribute based encryption scheme, for any PPT adversary* Att,

$$\left|\mathsf{Adv}^2_{\mathsf{Att}} - \mathsf{Adv}^3_{\mathsf{Att}}\right| \leq \mathit{negl}(\lambda).$$

Proof. Note that the only difference between Game 2 and Game 3 is in the implementation of Samples Oracle O. Suppose there exists a PPT adversary Att such that $\left|\mathsf{Adv}^2_{\mathsf{Att}} - \mathsf{Adv}^3_{\mathsf{Att}}\right| = \epsilon$. We will construct a PPT algorithm $\mathcal{B}$ that interacts with Att and breaks the adaptive security of ABE scheme with advantage ϵ.

$\mathcal{B}$ receives $\mathsf{pk}_{\mathsf{ABE}}$ from the ABE challenger. It chooses $i^* \leftarrow [q_{\mathrm{par}}]$ and computes $(U, \tau) \leftarrow \mathsf{SimUGen}(1^\lambda)$.

Implementing the Samples Oracle $O : \mathcal{B}$ must implement the Samples Oracle. It maintains a table T which is initially empty. On receiving a query d for O, if there exists an entry of the form (d, α, β) in T, it outputs α. Else, if d is a new query, and is not of the form $\mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x$ for some x, it chooses $t \leftarrow \{0,1\}^{\ell_{\mathrm{inp}}}$, outputs $d(t)$ and stores $(d, d(t), \perp)$. Else, if $d = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x$, and d is not the $(i^*)^{\mathrm{th}}$ query, it chooses $t \in \{0,1\}^n$, computes $c = \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x)$ and stores (d, c, t) in T. Else, if $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$ is the $(i^*)^{\mathrm{th}}$ query, $\mathcal{B}$ chooses $t, \tilde{t} \leftarrow \{0,1\}^n$, sends $t, \tilde{t}$ as the challenge messages and x^* as the challenge attribute to the ABE challenger. It receives c in response. $\mathcal{B}$ stores (d^*, c, t) in T and outputs c.

The remaining parts are identical in both Game 2 and Game 3. During the pre-challenge query phase, $\mathcal{B}$ receives either constrained key queries or random oracle queries. On receiving constrained key query for circuit C, it sends C to the ABE challenger as a secret key query, and receives sk_C. It sends $(U, (\mathsf{pk}, \mathsf{sk}_C), \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\})$ to Att. On receiving a random oracle query y, it computes $\mathsf{SimRO}(\tau, y)$, where SimRO is allowed to query the Samples Oracle O. If $\mathcal{B}$ receives any constrained key query C such that $C(x^*) = 1$ (where $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$ was the $(i^*)^{\mathrm{th}}$ unique query to O), then B aborts.

In the challenge phase, $\mathcal{B}$ receives input x^*. If $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$ was not the $(i^*)^{\text{th}}$ query to O, $\mathcal{B}$ aborts. Else, let (d^*, α^*, β^*) be the corresponding entry in T. It chooses $b \leftarrow \{0,1\}$. If $b = 0$, it outputs $t_0 = \beta^*$, else it outputs $t_1 \leftarrow \{0,1\}^n$.

The post challenge phase is handled similar to the pre-challenge phase. Finally, Att outputs b'. If $b = b'$, $\mathcal{B}$ outputs 0, indicating c is an encryption of t. Else it outputs 1.

We will now analyse $\mathcal{B}$'s winning probability. Let x^* be the challenge input sent by Att. Note that if $\mathcal{B}$ aborts, then the $(i^*)^{\text{th}}$ unique query to O was not $d^* = \mathcal{C}\text{-}\mathsf{Prog}\{\mathsf{pk}_{\mathsf{ABE}}\}||x^*$, in which case, Att wins with probability exactly $1/2$.

If d^* was the $(i^*)^{\text{th}}$ query and c is an encryption of t, then this corresponds to Game 2. Else, it corresponds to Game 3. Note that $\Pr[\mathcal{B}$ outputs 0 — $c \leftarrow \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, t, x^*)] = \Pr[\mathsf{Att}$ wins in Game 2] and similarly, $\Pr[\mathcal{B}$ outputs 0 — $c \leftarrow \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, \tilde{t}, x^*)] = \Pr[\mathsf{Att}$ wins in Game 3]. Therefore, $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{ABE}} = \epsilon$.

Observation 2 *For any adversary* Att, $\mathsf{Adv}_{\mathsf{Att}}^{3} = 0$.

Proof. Note that Att receives no information about t_0 in the pre-challenge and post challenge phases. As a result, t_0 and t_1 look identical to Att.

References

1. Agrawal, S., Koppula, V., Waters, B.: Impossibility of simulation secure functional encryption even with random oracles. Cryptology ePrint Archive, Report 2016/959 (2016)
2. Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From selective to adaptive security in functional encryption. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II. pp. 657–677 (2015), https://doi.org/10.1007/978-3-662-48000-7_32
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security. pp. 62–73 (1993)
4. Boneh, D., Boyen, X.: Efficient selective-id secure identity-based encryption without random oracles. In: Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings. pp. 223–238 (2004), https://doi.org/10.1007/978-3-540-24676-3_14
5. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: ASIACRYPT. pp. 280–300 (2013)
6. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Proceedings of CRYPTO 2014 (2014)
7. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings. pp. 501–519 (2014)

8. Brakerski, Z., Vaikuntanathan, V.: Constrained key-homomorphic prfs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In: Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II. pp. 1–30 (2015)
9. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: STOC. pp. 209–218 (1998)
10. Coron, J., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 476–493 (2013)
11. Coron, J.S.: Optimal security proofs for pss and other signature schemes. In: Proc. EUROCRYPT 2002. Lecture Notes in Computer Science, vol. 2332, pp. 272–287. Springer (2002)
12. Fuchsbauer, G., Konstantinov, M., Pietrzak, K., Rao, V.: Adaptive security of constrained prfs. In: Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II. pp. 82–101 (2014)
13. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT (2013)
14. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II. pp. 479–499 (2013)
15. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Fully secure attribute based encryption from multilinear maps. Cryptology ePrint Archive, Report 2014/622 (2014), `http://eprint.iacr.org/`
16. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: FOCS. pp. 464–479 (1984)
17. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC (2013)
18. Goyal, R., Goyal, V.: Overcoming cryptographic impossibility results using blockchains. In: Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I. pp. 529–561 (2017)
19. Hofheinz, D., Jager, T., Khurana, D., Sahai, A., Waters, B., Zhandry, M.: How to generate and use universal parameters. In: ASIACRYPT (2016)
20. Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Proc. Public Key Cryptography 2012. Lecture Notes in Computer Science, vol. 7293, pp. 66–83. Springer (2012)
21. Hohenberger, S., Koppula, V., Waters, B.: Adaptively secure puncturable pseudorandom functions in the standard model. In: Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I. pp. 79–102 (2015)
22. Kaliski, B., Staddon, J.: Pkcs #1: Rsa cryptography specifications version 2.0 (1998)
23. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: ACM Conference on Computer and Communications Security. pp. 669–684 (2013)

24. Lewko, A.B., Waters, B.: Why proving HIBE systems secure is difficult. In: Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings. pp. 58–76 (2014)
25. Liu, Q., Zhandry, M.: Decomposable obfuscation: A framework for building applications of obfuscation from polynomial hardness. In: Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I. pp. 138–169 (2017)
26. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005. pp. 84–93 (2005)
27. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: EUROCRYPT. pp. 457–473 (2005)
28. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: STOC. pp. 475–484 (2014)
29. U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology: Digital Signature Standards (DSS) (2013), federal Information Processing Standards Publication 186-4
30. Waters, B.: Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In: CRYPTO. pp. 619–636 (2009)
31. Waters, B.: A punctured programming approach to adaptively secure functional encryption. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II. pp. 678–697 (2015)

A Preliminaries Continued

A.1 Universal Samplers

In a recent work, Hofheinz et al. [19] introduced the notion of universal samplers. Intuitively, a universal sampler scheme provides a concise way to sample pseudorandomly from arbitrary distributions. More formally, a universal sampler scheme $\mathcal{U}$, parameterized by polynomials $\ell_{\text{ckt}}, \ell_{\text{inp}}$ and ℓ_{out}, consists of algorithms $\mathsf{US.setup}$ and $\mathsf{US.sample}$ defined below.

- $\mathsf{US.setup}(1^\lambda)$ takes as input the security parameter λ and outputs the sampler parameters U.
- $\mathsf{US.sample}(U, d)$ is a deterministic algorithm that takes as input the sampler parameters U and a circuit d of size at most ℓ_{ckt} bits. The circuit d takes as input ℓ_{inp} bits and outputs ℓ_{out} bits. The output of $\mathsf{US.sample}$ also consists of ℓ_{out} bits.

Intuitively, $\mathsf{US.sample}$ is supposed to sample from d, in the sense that it outputs a value $d(z)$ for pseudorandom and hidden random coins z. However, it is nontrivial to define what it means that the random coins z are hidden, and that even multiple outputs (for adversarially and possibly even adaptively chosen circuits d) look pseudorandom.

Hofheinz et al. [19] formalize security by mandating that $\mathsf{US.sample}$ is programmable in the random oracle model. In particular, there should be an efficient

way to simulate U and the random oracle, such that $\mathsf{US.sample}$ outputs an externally given value that is honestly sampled from d. This programming should work even for arbitrarily many $\mathsf{US.sample}$ outputs for adversarially chosen inputs d simultaneously, and it should be indistinguishable from a real execution of $\mathsf{US.setup}$ and $\mathsf{US.sample}$.

In this work, we will be using a universal sampler scheme that is even adaptively secure. In order to formally define adaptive security for universal samplers, let us first define the notion of an admissible adversary $\mathcal{A}$.

An admissible adversary $\mathcal{A}$ is defined to be an efficient interactive Turing Machine that outputs one bit, with the following input/output behavior:

- $\mathcal{A}$ takes as input security parameter λ and sampler parameters U.
- $\mathcal{A}$ can send a random oracle query (RO, x), and receives the output of the random oracle on input x.
- $\mathcal{A}$ can send a message of the form (params, d) where $d \in \mathcal{C}[\ell_{\mathrm{ckt}}, \ell_{\mathrm{inp}}, \ell_{\mathrm{out}}]$. Upon sending this message, $\mathcal{A}$ is required to honestly compute $p_d = \mathsf{US.sample}(U, d)$, making use of any additional random oracle queries, and $\mathcal{A}$ appends (d, p_d) to an auxiliary tape (this is required to check for *Honest Sample Violation* in the Ideal experiment).

Let $\mathsf{SimUGen}$ and SimRO be PPT algorithms. Consider the following two experiments:

$\mathsf{Real}^{\mathcal{A}}(1^\lambda)$:

1. The random oracle RO is implemented by assigning random outputs to each unique query made to RO.
2. $U \leftarrow \mathsf{US.setup}^{\mathsf{RO}}(1^\lambda)$.
3. $\mathcal{A}(1^\lambda, U)$ is executed, where every random oracle query, represented by a message of the form (RO, x), receives the response $\mathsf{RO}(x)$.
4. Upon termination of $\mathcal{A}$, the output of the experiment is the final output of the execution of $\mathcal{A}$.

$\mathsf{Ideal}^{\mathcal{A}}_{\mathsf{SimUGen},\mathsf{SimRO}}(1^\lambda)$:

1. A truly random function F that maps ℓ_{ckt} bits to ℓ_{inp} bits is implemented by assigning random ℓ_{inp}-bit outputs to each unique query made to F. Throughout this experiment, a Samples Oracle O is implemented as follows: On input d, where $d \in \mathcal{C}[\ell_{\mathrm{ckt}}, \ell_{\mathrm{inp}}, \ell_{\mathrm{out}}]$, O outputs $d(F(d))$.
2. $(U, \tau) \leftarrow \mathsf{SimUGen}(1^\lambda)$. Here, $\mathsf{SimUGen}$ can make arbitrary queries to the Samples Oracle O.
3. $\mathcal{A}(1^\lambda, U)$ and $\mathsf{SimRO}(\tau)$ begin simultaneous execution.
   - Whenever $\mathcal{A}$ sends a message of the form (RO, x), this is forwarded to SimRO, which produces a response to be sent back to $\mathcal{A}$.
   - SimRO can make any number of queries to the Samples Oracle O.
   - Finally, after $\mathcal{A}$ sends any message of the form (params, d), the auxiliary tape of $\mathcal{A}$ is examined until an entry of the form (d, p_d) is added to it. At this point, if p_d is not equal to $d(F(d))$, then experiment aborts, resulting in an *Honest Sample Violation*.

4. Upon termination of $\mathcal{A}$, the output of the experiment is the final output of the execution of $\mathcal{A}$.

Definition 2. *A universal sampler scheme $\mathcal{U} = (\mathsf{US.setup}, \mathsf{US.sample})$, parameterized by polynomials $\ell_{\mathrm{ckt}}, \ell_{\mathrm{inp}}$ and ℓ_{out}, is said to be adaptively secure in the random oracle model if there exist PPT algorithms $\mathsf{SimUGen}$ and SimRO such that for all admissible PPT adversaries $\mathcal{A}$, the following hold:*

$$\Pr[\mathsf{Ideal}^{\mathcal{A}}_{\mathsf{SimUGen},\mathsf{SimRO}}(1^\lambda) \text{ aborts }] = 0,$$ [8]

and

$$\left|\Pr[\mathsf{Real}^{\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{Ideal}^{\mathcal{A}}_{\mathsf{SimUGen},\mathsf{SimRO}}(1^\lambda) = 1]\right| \leq \mathit{negl}(\lambda)$$

Hofheinz et al. [19] construct a universal sampler scheme that is adaptively secure in the random oracle model, assuming a secure indistinguishability obfuscator, a selectively secure puncturable PRF and an injective pseudorandom generator.

A.2 Attribute Based Encryption

An attribute based encryption scheme ABE for a circuit family $\mathcal{F}$ with message space $\mathcal{M}$ and attribute space $\mathcal{X}$ consists of algorithms $\mathsf{ABE.setup}$, $\mathsf{ABE.keygen}$, $\mathsf{ABE.enc}$ and $\mathsf{ABE.dec}$ defined below.

- $\mathsf{ABE.setup}(1^\lambda)$ is a PPT algorithm that takes as input the security parameter and outputs the public key $\mathsf{pk}_{\mathsf{ABE}}$ and the master secret key $\mathsf{msk}_{\mathsf{ABE}}$.
- $\mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ is a PPT algorithm that takes as input the master secret key $\mathsf{msk}_{\mathsf{ABE}}$, a circuit $C \in \mathcal{F}$ and outputs a secret key sk_C for circuit C.
- $\mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x)$ takes as input a public key $\mathsf{pk}_{\mathsf{ABE}}$, message $m \in \mathcal{M}$, an attribute $x \in \mathcal{X}$ and outputs a ciphertext c. We will assume the encryption algorithm takes ℓ_{rnd} bits of randomness [9]. The notation $\mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x; r)$ is used to represent the randomness r used by $\mathsf{ABE.enc}$.
- $\mathsf{ABE.dec}(\mathsf{sk}_C, c)$ takes as input secret key sk_C, ciphertext c and outputs $y \in \mathcal{M} \cup \{\perp\}$.

Correctness For any circuit $C \in \mathcal{F}$, $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$, message $m \in \mathcal{M}$, attribute $x \in \mathcal{X}$ such that $C(x) = 1$, we require the following:

$$\mathsf{ABE.dec}(\mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C), \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x)) = m.$$

For simplicity of notation, we will assume $\mathsf{ABE.dec}(\mathsf{msk}_{\mathsf{ABE}}, \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x)) = m$ for all messages m, attributes x [10].

[8] The definition in [19] only requires this probability to be negligible in λ. However, the construction actually achieves zero probability of Honest Sample Violation. Hence, for the simplicity of our proof, we will use this definition

[9] This assumption can be justified by the use of an appropriate pseudorandom generator that maps ℓ_{rnd} bits to the required length.

[10] We can assume this holds true, since given $\mathsf{msk}_{\mathsf{ABE}}$, one can compute a secret key sk for circuit C_{all} that accepts all inputs, and then use sk to decrypt $\mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m, x)$.

Security Security for an ABE scheme is defined via the following adaptive security game between a challenger and adversary Att.

1. **Setup Phase** The challenger chooses $(\mathsf{pk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.setup}(1^\lambda)$ and sends $\mathsf{pk}_{\mathsf{ABE}}$ to Att.
2. **Pre-Challenge Phase** The challenger receives multiple secret key queries. For each $C \in \mathcal{F}$ queried, it computes $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$ and sends sk_C to Att.
3. **Challenge** Att sends messages $m_0, m_1 \in \mathcal{M}$ and attribute $x \in \mathcal{X}$ such that $C(x) = 0$ for all circuits queried during the Pre-Challenge phase. The challenger chooses $b \leftarrow \{0,1\}$, computes $c \leftarrow \mathsf{ABE.enc}(\mathsf{pk}_{\mathsf{ABE}}, m_b, x)$ and sends c to Att.
4. **Post-Challenge Phase** Att sends multiple secret key queries $C \in \mathcal{F}$ as in the Pre-Challenge phase, but with the added restriction that $C(x) = 0$. It receives $\mathsf{sk}_C \leftarrow \mathsf{ABE.keygen}(\mathsf{msk}_{\mathsf{ABE}}, C)$.
5. **Guess** Finally, Att outputs its guess b'.

Att wins the ABE security game for scheme ABE if $b = b'$. Let $\mathsf{Adv}^{\mathsf{ABE}}_{\mathsf{Att}} = \left| \Pr[\mathsf{Att} \text{ wins}] - 1/2 \right|$.

Definition 3. *An ABE scheme* $\mathsf{ABE} = (\mathsf{ABE.setup}, \mathsf{ABE.keygen}, \mathsf{ABE.enc}, \mathsf{ABE.dec})$ *is said to be adaptively secure if for all PPT adversaries* Att, $\mathsf{Adv}^{\mathsf{ABE}}_{\mathsf{Att}} \leq negl(\lambda)$.

In a recent work, Waters [31] showed a construction for an adaptively secure functional encryption scheme, using indistinguishability obfuscation. An adaptively secure functional encryption scheme implies an adaptively secure attribute based encryption scheme. Garg, Gentry, Halevi and Zhandry [15] showed a direct construction based on multilinear encodings. Ananth, Brakerski, Segev and Vaikuntanathan [2] showed how to transform any selectively secure FE scheme to achieve adaptive security.