Private Functional Encryption: Indistinguishability-Based Definitions and Constructions from Obfuscation

Afonso Arriaga[1], Manuel Barbosa[2], and Pooya Farshim[3]

[1] SnT, University of Luxembourg, Luxembourg
[2] HASLab - INESC TEC, DCC FC University of Porto, Portugal
[3] Queen's University Belfast, United Kingdom
afonso.delerue@uni.lu mbb@dcc.fc.up.pt pooya.farshim@gmail.com

Abstract. Private functional encryption guarantees that not only the information in ciphertexts is hidden but also the circuits in decryption tokens are protected. A notable use case of this notion is query privacy in searchable encryption. Prior privacy models in the literature were fine-tuned for specific functionalities (namely, identity-based encryption and inner-product encryption), did not model correlations between ciphertexts and decryption tokens, or fell under strong uninstantiability results. We develop a new indistinguishability-based privacy notion that overcomes these limitations and give constructions supporting different circuit classes and meeting varying degrees of security.

Obfuscation is a common building block that these constructions share, albeit the obfuscators necessary for each construction are based on different assumptions. Our feasibility results go beyond previous constructions in a number of ways. In particular, a keyword search scheme that we base on point obfuscators tolerates *arbitrary and possibly low-entropy* correlations between encrypted data and queried keywords, even under (mildly restricted) adaptive token-extraction queries. Our more elaborate keyword search scheme achieves the strongest notion of privacy that we put forth (with no restrictions), but relies on stronger forms of obfuscation. We also develop a composable and distributionally secure inner-product obfuscator and use it to build an inner-product encryption scheme that achieves an unprecedented level of privacy. This, in particular, positively answers a question left open by Boneh, Raghunathan and Segev (ASIACRYPT 2013) concerning the extension and realization of *enhanced security* for hyperplane membership.

Keywords: Function privacy, functional encryption, obfuscation, keyword search, inner-product encryption.

Table of Contents

1 Introduction

Standard notions of security for public-key functional encryption [21,36] do not cover important use cases where, not only encrypted data, but also the circuits (functions) associated with decryption tokens contain sensitive information. The typical example is that of a cloud provider that stores an encrypted data set created by Alice, over which Bob wishes to make advanced data mining queries. Functional encryption provides a solution to this use case: Bob can send a decryption token to the cloud provider that allows it to recover the result of computing a query over the encrypted data set. Standard security notions guarantee that nothing about the plaintexts beyond query results are revealed to the server. However, they do *not* guarantee that the performed query, which may for example contain a keyword sensitive to Bob, is hidden from the server.

1.1 Function privacy

Function privacy is an emerging new notion that aims to address this problem. The formal study of this notion begins in the work of Boneh, Raghunathan and Segev [19], where the authors focused on identity-based encryption (IBE) and presented the first constructions offering various degrees of privacy. From the onset, it became clear that formalizing such a notion is challenging, even for simple functionalities such as IBE, as the holder of a token for circuit C may encrypt arbitrary messages using the public key, and obtain a large number of evaluations of C via the decryption algorithm. Boneh et al. therefore considered privacy for identities with high min-entropy. In general, however, the previous observation implies that (non-trivial) function privacy can only be achieved as long as the token holder is unable to learn C through such an attack, immediately suggesting a strong connection between private functional encryption and obfuscation.

Boneh, Raghunathan and Segev [19,20] give indistinguishability-based definitions of function privacy for IBE and subspace membership (a generalization of inner-product encryption). Roughly speaking, the IBE model imposes that whenever the token queries of the adversary have high min-entropy (or form a block source) decryption tokens will be indistinguishable from those corresponding to identities sampled from the uniform distribution. For subspace membership, the definition requires the random variables associated with vector components to be a block source.

Tokens for high-entropy identities, however, rarely exist in isolation and are often available in conjunction with ciphertexts encrypted for the very same identities. To address this requirement, the same authors [19] proposed an *enhanced* model for IBE in which the adversary also gets access to ciphertexts encrypted for identities associated with the challenge tokens. This model was subsequently shown in [6] to be infeasible under the formalism of Boneh et al., as correlations with encrypted identities can lead to distinguishing attacks, e.g. via *repetition patterns.* (We will discuss this later in the paper.) Although the model can be salvaged by further restricting the class of admissible distributions, it becomes primitive-specific and formulating a definition for other functionalities is not obvious (and indeed a similar extension was not formalized for subspace membership in [20]). Additionally, this model also falls short of capturing *arbitrary* correlations between encrypted messages and tokens, as it does not allow an adversary to see ciphertexts for identities which, although correlated with those extracted in the challenge tokens, *do not match any of them.*

Very recently, Agrawal et al. [2] have put forth a model for functional encryption that aims to address this problem with a very general UC-style definition (called "wishful security"). The core of the definition is an ideal security notion for functional encryption, which makes it explicit that both data privacy and function privacy should be simultaneously enforced. However, not only is this general simulation-based definition difficult to work with, but also aiming for it would amount to constructing virtual black-box obfuscation, for which strong impossibility results are known [9,28]. Indeed, the positive results of [2] are obtained in idealized models of computation.

1.2 Contributions

The above discussion highlights the need for a general and convenient definition of privacy that incorporates arbitrary correlations between decryption tokens and encrypted messages, and yet can be shown to be feasible without relying on idealized models of computation. The first contribution of our work is an *indistinguishability-based* definition that precisely models arbitrary correlations for general circuits. Our definition builds on a framework for *unpredictable* samplers and unifies within a single definition all previous indistinguishability-based notions.

The second contribution of the paper is four constructions of private functional encryption supporting different classes of circuits and meeting varying degrees of security: (1) a simple and functionality-agnostic construction shown to be secure in the absence of correlated messages, (2) a more evolved and still functionality-agnostic construction (taking advantage of recent techniques from [5,25]) that achieves function privacy with respect to a general class of samplers that we call *concentrated*; (3) a conceptually simpler construction specific for point functions achieving privacy in the presence of correlated messages beyond all previously proposed indistinguishability-based security definitions; (4) a construction specific for point functions that achieves our strongest notion of privacy (but relies on a more expressive form of obfuscation than the previous construction). We also develop an obfuscator for hyperplane membership that, when plugged into the second construction above gives rise to a private inner-product encryption scheme, answering a question left open by Boneh, Raghunathan and Segev [20] on how to define and realize *enhanced* security (i.e., privacy in the presence of correlated messages) for schemes supporting this functionality.

The unpredictability framework. At the core of our definitional work lies a precise definition characterizing which distributions over circuits and what correlated side information can be tolerated by a private FE scheme. We build on ideas from obfuscation [12,14,7,12], functional encryption [21,36] and prior work in function privacy [19,20,6,2] to define a game-based notion of *unpredictability* for general functions. Our definition allows a *sampler* $\mathcal{S}$ to output a pair of circuit vectors $(\mathbf{C}_0, \mathbf{C}_1)$ and a pair of message vectors $(\mathbf{m}_0, \mathbf{m}_1)$ with arbitrary correlations between them, along with some side information z. Unpredictability then imposes that no predictor $\mathcal{P}$ interacting with oracles computing evaluations on these circuits and messages can find a point x such that $\mathbf{C}_0(x) \neq \mathbf{C}_1(x)$. (We do not impose indistinguishability, which is stronger, results in a smaller class of unpredictable samplers, and hence leads to weaker security.) The predictor $\mathcal{P}$ sees z and the outputs of the sampled circuits on the sampled messages. It can run in bounded or unbounded time, but it can only make polynomially many oracle queries to obtain additional information about the sampled circuits and messages. To avoid attacks that arise in the presence of computationally unpredictable auxiliary information [22,13] we adopt *unbounded* prediction in our analyses.

This formalism fixes the unpredictability notion throughout the paper. We can then capture specific types of samplers by imposing extra structural requirements on them. For instance, we may require the sampler to always return empty auxiliary information, or to include the descriptions of the circuits (leading to functionally equivalent [30,22,26] and differing inputs [4,12] notions) or even its coins [35]. A bounded number of circuits and even the non-existence of correlated messages can be modeled similarly. We emphasize that this definition intentionally does *not* require the messages to be unpredictable. Further discussion on this choice can be found in Section 3.

The PRIV model. Building on unpredictability, we put forth a new indistinguishability-based notion of function privacy. Our notion, which we call PRIV, bears close resemblance to the standard IND-CPA model for functional encryption: it comes with a left-or-right LR oracle, a token-extraction TGen oracle and the goal of the adversary is to guess a bit. The power of the model lies in that we endow LR with the ability to generate arbitrary messages and circuits via an unpredictable sampler. Trivial attacks are excluded by the joint action of unpredictability and the usual FE legitimacy condition, imposing equality of images on left and right. The enhanced model of Boneh, Raghunathan and Segev [19] falls in as a special case where the sampler is structurally restricted to be a block source. But our definition goes well beyond this and considers arbitrary and possibly low-entropy correlations. Furthermore, since unpredictability is *not* imposed on messages, PRIV implies IND-CPA security, and consequently it also guarantees *anonymity* for primitives such as IBEs and ABEs [21]. Correlated circuits may be "low entropy" as long as they are identical on left and right, and since previous definitions adopted a real-or-random definition, they had to exclude this possibility. By giving the sampler the option to omit, manipulate and repeat the messages, our security notion implies previous indistinguishability-based notions in the literature, including those in [19,20,6,2].

The implications of our new definition become clearer when we focus on (public-key encryption with) keyword search (KS) [18]. Consider a scenario where a client searches for a keyword, slightly modifies it by editing it, and then uploads an encryption of the keyword to the server. In this setting, the server sees ciphertexts encrypting unknown keywords that are closely related to keywords which the server holds tokens for. Our model ensures that if searched keywords are unpredictable from the perspective of the server, this uncertainty is preserved by the KS scheme after the searches are carried out. This does *not* imply that the server will be unable to distinguish a sequence of successful queries over the *same* high-entropy keyword, from a sequence of successful queries over *different* high-entropy keywords (this is impossible to achieve [6]). However, when keyword searches do *not* match any of the correlated ciphertexts, then search patterns are guaranteed to remain hidden, even in the presence of low-entropy correlated encrypted keywords. We note that this captures a strong notion of unlinkability and untraceability between *unmatched queries.*

Constructions. We start by formalizing the intuition that obfuscating circuits before extraction should provide some level of privacy in FE. Using unpredictable samplers, we first generalize distributionally-indistinguishable (DI) obfuscators [14] from point functions to general circuits. Our obfuscate-then-extract OX transform shows that PRIV security in the absence of correlated messages can be achieved using DI obfuscators. (A partial reduction in the reverse direction also holds, and as a result some form of DI obfuscation is also necessary.) Similarly to point functions, we prove that single-circuit DI is implied by virtual grey-box (VGB) obfuscation, for which positive results have been emerging [15,14]. Analogously, *(self-)composable* VGB obfuscation implies full-fledged DI obfuscation. (For these results we crucially rely on the fact that the predictor is *semi-bounded.*)

To move beyond the above *token-only* model, we need to "decouple" the correlations between encrypted messages and challenge circuits so we can take advantage of FE security (that protects ciphertexts) and obfuscation (that protects the circuits) in a cumulative way. Building on ideas from [5] and [15] we identify a class of *concentrated* samplers that can be used in conjunction with the so-called "trojan" method—a technique to boost selective security to adaptive security in FE—to achieve function privacy. This construction improves on the security guarantees of OX considerably, but comes with the caveat that a mild restriction on second-stage token queries must be imposed: they must reveal (via circuit outputs) no more information about encrypted correlated messages than those revealed by first-stage queries. We give non-trivial examples of concentrated samplers and derive constructions for classes of circuits that encompass, among other functionalities, IBE, KS and *inner-product encryption*. By giving a construction of composable obfuscator for hyperplane membership we resolve a question left open by Boneh, Raghunathan and Segev [20] on the extension of enhanced security to inner-product encryption. Indeed, the restriction we impose on second-stage queries is weak enough to yield a model that implies enhanced security.

The previous construction excludes search patterns that can reveal sensitive information via special types of low-entropy encrypted messages. Our third construction is specific to point functions, and besides being simpler and more efficient, can tolerate arbitrary correlations between challenge keywords and encrypted messages. Put differently this construction removes the concentration restriction on samplers. For this construction we require a functional encryption scheme that supports the OR composition of two DI-secure point obfuscations. The composable VGB point obfuscator of Bitansky and Canetti [14] implies that the required DI point obfuscator exists. Furthermore, we also rely on a standard functional encryption scheme that supports the evaluations of four group operations in a DDH group (corresponding to the disjunction of two point function obfuscations), which is a relatively modest computation. We are, however, unable to lift the mild second-stage restriction.

Our last construction lifts the second-stage restriction at the cost of relying on more expressive forms of obfuscators. The novelty in this construction resides in the observation that, in order to offer the keyword search functionality, it suffices to encrypt information that enables equality checks between words and messages to be carried out. In our fourth construction we encode a message m as an *obfuscation* of the point function $\mathsf{C}[\mathsf{m}]$. Concretely, we obfuscate words before extraction *and* messages before encryption. Equality with w can be checked using a circuit $\mathsf{D}[\mathsf{w}]$ that on input an obfuscated point function $\mathsf{Obf}(\mathsf{C}[\mathsf{m}])$ returns $\mathsf{Obf}(\mathsf{C}[\mathsf{m}])(\mathsf{w})$. We emphasize that $\mathsf{D}[\mathsf{w}]$ is *not* a point function. We also need to ensure that an attacker cannot exploit the $\mathsf{D}[\mathsf{w}]$ circuits by, say, encrypting obfuscations of malicious circuits of its choice. We do this using NIZK proofs to ensure the outputs of the point obfuscator are *verifiable*: one can publicly verify that an obfuscation indeed corresponds to some point function. To summarize, our construction relies on a composable VGB obfuscator supporting point functions $\mathsf{C}[\mathsf{m}](\mathsf{w}) := (\mathsf{m} = \mathsf{w})$ and circuits $\mathsf{D}[\mathsf{w}](\overline{\mathsf{C}}) := \overline{\mathsf{C}}(\mathsf{w})$ and a general-purpose FE. The circuits $\mathsf{C}[\mathsf{m}]$ and $\mathsf{D}[\mathsf{w}]$ were used negatively by Barak et al. [9] to launch generic attacks against VBB and VGB obfuscators. Here, the restrictions imposed on legitimate PRIV samplers ensure that these attacks cannot be carried out in our setting, and obfuscators supporting them can be used positively to build private FE schemes.

2 Preliminaries

NOTATION. We denote the security parameter by $\lambda \in \mathbb{N}$ and assume it is implicitly given to all algorithms in unary representation 1^λ. We denote the set of all bit strings of length ℓ by $\{0,1\}^\ell$ and the length of a string x by $|x|$. The bit complement of a string x is denoted by $\overline{x}$. A vector of strings $\mathbf{x}$ is written in boldface, and $\mathbf{x}[i]$ denotes its ith entry. The number of entries of $\mathbf{x}$ is denoted by $|\mathbf{x}|$. For a finite set X, we denote its cardinality by $|X|$ and the action of sampling a uniformly random element x from X by $x \leftarrow_\$ X$. For a random variable X we denote its support by $[X]$. For a circuit

C we denote the size C by $|\mathsf{C}|$. We call a real-valued function $\mu(\lambda)$ negligible if $\mu(\lambda) \in \mathcal{O}(\lambda^{-\omega(1)})$ and denote the set of all negligible functions by Negl. Throughput the paper $\perp$ denotes a special failure symbol outside the spaces underlying a cryptographic primitive. We adopt the code-based game-playing framework. As usual "ppt" stands for probabilistic polynomial time.

Circuit families. Let $\mathsf{MSp} := \{\mathsf{MSp}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathsf{OSp} := \{\mathsf{OSp}_\lambda\}_{\lambda\in\mathbb{N}}$ be two families of finite sets parametrized by a security parameter $\lambda \in \mathbb{N}$. A circuit family $\mathsf{CSp} := \{\mathsf{CSp}_\lambda\}_{\lambda\in\mathbb{N}}$ is a sequence of circuit sets indexed by the security parameter. We assume that for all $\lambda \in \mathbb{N}$, all circuits in CSp_λ share a common input domain MSp_λ and output space OSp_λ. We also assume that membership in sets can be efficiently decided. For a vector of circuits $\mathbf{C} = [\mathsf{C}_1, \ldots, \mathsf{C}_n]$ and a vector of messages $\mathbf{m} = [\mathsf{m}_1, \ldots, \mathsf{m}_m]$ we define $\mathbf{C}(\mathbf{m})$ to be an $n \times m$ matrix whose ijth entry is $\mathsf{C}_i(\mathsf{m}_j)$. When $\mathsf{OSp}_\lambda = \{0, 1\}$ for all values of λ we call the circuit family Boolean.

2.1 Pseudorandom permutations

Pseudorandom permutations. Let $\mathsf{KSp} := \{\mathsf{KSp}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathsf{MSp} := \{\mathsf{MSp}_\lambda\}_{\lambda\in\mathbb{N}}$ be two families of finite sets parametrized by a security parameter $\lambda \in \mathbb{N}$. A pseudorandom permutation (PRP) family $\mathsf{PRP} := (\mathsf{K}, \mathsf{E}, \mathsf{D})$ is a triple of ppt algorithms as follows. (1) K on input the security parameter outputs a uniform element in KSp_λ; (2) E is deterministic and on input a key $\mathsf{k} \in \mathsf{KSp}_\lambda$ and a point $\mathsf{x} \in \mathsf{MSp}_\lambda$ outputs a point in MSp_λ; (3) D is deterministic and on input a $\mathsf{k} \in \mathsf{KSp}_\lambda$ and a point $\mathsf{x} \in \mathsf{MSp}_\lambda$ outputs a point in MSp_λ. The PRP family PRP is correct if for all $\lambda \in \mathbb{N}$, all $\mathsf{k} \in \mathsf{KSp}_\lambda$ and all $\mathsf{x} \in \mathsf{MSp}_\lambda$ we have that $\mathsf{D}(\mathsf{k}, \mathsf{E}(\mathsf{k}, \mathsf{x})) = \mathsf{x}$. A pseudorandom permutation $\mathsf{PRP} := (\mathsf{K}, \mathsf{E}, \mathsf{D})$ is called PRP secure if for every ppt adversary $\mathcal{A}$ we have that

$$\mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathrm{PRP}^{\mathcal{A}}_{\mathsf{PRP}}(1^\lambda)\right] - 1 \in \textsc{Negl}$$

where game $\mathrm{PRP}^{\mathcal{A}}_{\mathsf{PRP}}(1^\lambda)$ is defined in Fig. 1. For our purposes, we rely on the non-strong security notion where inverse queries are not allowed. Furthermore, we do not necessarily require the inverse map D to be efficiently computable.

```
$\mathrm{PRP}^{\mathcal{A}}_{\mathsf{PRP}}(1^\lambda)$:
$b \leftarrow\$ \{0,1\}$
$\mathsf{k} \leftarrow\$ \mathsf{K}(1^\lambda)$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Fn}}(1^\lambda)$
return $(b = b')$

$\textsc{Fn}(\mathsf{x})$:
if $T[\mathsf{x}] = \perp$ then
    $T[\mathsf{x}] \leftarrow\$ \mathsf{MSp}_\lambda \setminus T$
if $b = 1$ return $T[\mathsf{x}]$
else return $\mathsf{E}(\mathsf{k}, \mathsf{x})$
```

Fig. 1. Game defining the PRP security of a pseudorandom permutation $\mathsf{PRP} := (\mathsf{K}, \mathsf{E}, \mathsf{D})$.

2.2 Functional encryption

Syntax. A functional encryption scheme FE associated with a circuit family CSp is specified by four ppt algorithms as follows. (1) $\mathsf{FE.Gen}(1^\lambda)$ is the setup algorithm and on input a security parameter 1^λ it outputs a master secret key msk and a master public key mpk; (2) $\mathsf{FE.TGen}(\mathsf{msk}, \mathsf{C})$ is the token-generation algorithm and on input a master secret key msk and a circuit $\mathsf{C} \in \mathsf{CSp}_\lambda$ outputs a token tk for C; (3) $\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{m})$ is the encryption algorithm and on input a master public

```
$\mathrm{CC}^{\mathcal{A}}_{\mathsf{FE}}(1^\lambda)$:
  $(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
  $(\mathsf{m}, \mathsf{C}) \leftarrow\$ \mathcal{A}^{\mathrm{TGen}}(\mathsf{mpk})$
  $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{m})$
  $\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, \mathsf{C})$
  $y \leftarrow\$ \mathsf{FE.Eval}(\mathsf{c}, \mathsf{tk})$
  return $(y \neq \mathsf{C}(\mathsf{m}))$

$\mathrm{TGen}(\mathsf{C})$:
  $\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, \mathsf{C})$
  return $\mathsf{tk}$
```

```
$\mathrm{IND\text{-}CPA}^{\mathcal{A}}_{\mathsf{FE}}(1^\lambda)$:
  $(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
  $b \leftarrow\$ \{0, 1\}$
  $b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
  return $(b = b')$
```

```
$\mathrm{LR}(\mathsf{m}_0, \mathsf{m}_1)$:
  $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{m}_b)$
  $\mathsf{MList} \leftarrow (\mathsf{m}_0, \mathsf{m}_1) : \mathsf{MList}$
  return $\mathsf{c}$

$\mathrm{TGen}(\mathsf{C})$:
  $\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, \mathsf{C})$
  $\mathsf{TList} \leftarrow \mathsf{C} : \mathsf{TList}$
  return $\mathsf{tk}$
```

Fig. 2. Left: Computational correctness of FE. **Right:** IND-CPA security of FE.

key mpk and a message $\mathsf{m} \in \mathsf{MSp}_\lambda$ outputs a ciphertext c; (4) $\mathsf{FE.Eval}(\mathsf{c}, \mathsf{tk})$ is the deterministic evaluation (or decryption) algorithm and on input a ciphertext c and a token tk outputs a value $\mathsf{y} \in \mathsf{OSp}_\lambda$ or failure symbol $\perp$.

We adopt a *computational* notion of correctness for FE schemes and require that no ppt adversary is able to produce a message m and a circuit C that violates the standard correctness property of the FE scheme (that is, $\mathsf{FE.Eval}(\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{m}), \mathsf{tk}) \neq \mathsf{C}(\mathsf{m})$), even with the help of an (unrestricted) token-generation oracle. We also adopt the standard notion of IND-CPA security [21,36] where an adversary with access to a token-generation oracle cannot distinguish encryptions of messages m_0, m_1 under the standard restriction that it cannot obtain a decryption token for a circuit C for which $\mathsf{C}(\mathsf{m}_0) \neq \mathsf{C}(\mathsf{m}_1)$.

Correctness. We will adopt a game-based definition of *computational* correctness for FE schemes which has been widely adopted in the literature [1,27] and suffices for the overwhelming majority of use cases. Roughly speaking, this property requires that no efficient adversary is able to come up with a message and a circuit which violates the correctness property of the FE scheme, even with the help of an (unrestricted) token-generation oracle. Formally, scheme FE is computationally correct if for all ppt adversaries $\mathcal{A}$

$$\mathbf{Adv}^{\mathrm{cc}}_{\mathsf{FE},\mathcal{A}}(\lambda) := \Pr\left[\mathrm{CC}^{\mathcal{A}}_{\mathsf{FE}}(1^\lambda)\right] \in \mathrm{Negl}\ ,$$

where game $\mathrm{CC}^{\mathcal{A}}_{\mathsf{FE}}(1^\lambda)$ is shown in Fig. 2 on the left. Perfect correctness corresponds to the setting where the above advantage is required to be zero.

Security. A functional encryption scheme FE is IND-CPA secure [21,36] if for any legitimate ppt adversary $\mathcal{A}$

$$\mathbf{Adv}^{\mathrm{ind\text{-}cpa}}_{\mathsf{FE},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathrm{IND\text{-}CPA}^{\mathcal{A}}_{\mathsf{FE}}(1^\lambda)\right] - 1 \in \mathrm{Negl}\ ,$$

where game $\mathrm{IND\text{-}CPA}_{\mathsf{FE},\mathcal{A}}(1^\lambda)$ is defined in Fig. 2 on the right. We say $\mathcal{A}$ is legitimate if for all messages pairs queried to the let-or-right oracle, i.e., for all $(\mathsf{m}_0, \mathsf{m}_1) \in \mathsf{MList}$, and all extracted circuits $\mathsf{C} \in \mathsf{TList}$ we have that $\mathsf{C}(\mathsf{m}_0) = \mathsf{C}(\mathsf{m}_1)$.

The IND-CPA notion self-composes in the sense that security against adversaries that place one LR query is equivalent to the setting where an arbitrary number of queries is allowed. It is also well known that IND-CPA security is weaker than generalizations of semantic security for functional encryption [21,36,10], and strong impossibility results for the latter have been established [21,31,3]. On the other hand, IND-CPA-secure FE schemes for all polynomial-size circuit families have been recently constructed [31,26,29]. Other recent feasibility results have been established in weaker forms of the IND-CPA model such as the selective model [31,26,29] where the adversary commits to its challenge messages at the onset; or the weak model for Boolean circuits, where the adversary is restricted to extract tokens that evaluate to 0 on the challenge messages [32].

2.3 Keyword search

Set circuits. In this work we are interested in circuits that assume a non-zero value on only a polynomially large subset of their domains. We call these *set circuits*. Formally, a family of Boolean circuits CSp is a set circuit family if there is a polynomial poly such that for all $\lambda \in \mathbb{N}$ and all $\mathsf{C} \in \mathsf{CSp}_\lambda$ we have that $|S(\mathsf{C})| \leq \mathsf{poly}(\lambda)$ where $S(\mathsf{C}) := \{\mathsf{m} \in \mathsf{MSp}_\lambda : \mathsf{C}(\mathsf{m}) = 1\}$. For any set circuit C' with corresponding set $S = S(\mathsf{C}')$ we define the *canonical* representation of C' as the circuit $\mathsf{C}[S]$ that has the set S explicitly hardwired in it. Point circuits/functions correspond to the case where $\mathsf{poly}(\lambda) = 1$. We use $\mathsf{C}[\mathsf{m}]$ to denote the point circuit that on input m returns 1 and 0 otherwise. Throughout the paper, we assume that non-obfuscated set circuits are canonically represented.

Syntax. A public-key encryption with keyword search scheme (or simply a keyword search scheme) KS is a functional encryption scheme for a point circuit family over the message space: $\mathsf{CSp}_\lambda := \{\mathsf{C}[\mathsf{m}] : \mathsf{m} \in \mathsf{MSp}_\lambda\}$. We often identify circuit $\mathsf{C}[\mathsf{m}]$ with its message m, but in order to distinguish circuits from messages we use the term *keyword* to refer to the former. We write the algorithms associated to a KS scheme as $\mathsf{KS.Gen}(1^\lambda)$, $\mathsf{KS.Enc}(\mathsf{pk}, \mathsf{m})$, $\mathsf{KS.TGen}(\mathsf{sk}, \mathsf{w})$ and $\mathsf{KS.Test}(\mathsf{c}, \mathsf{tk})$, where the latter outputs either 0 or 1. The computational correctness of a KS scheme is defined identically to that of an FE scheme. IND-CPA security is also defined identically to FE schemes for point families. Note that weak and standard IND-CPA notions are equivalent for KS schemes.

2.4 NIZK proof systems

Syntax. A non-interactive zero-knowledge proof system for an **NP** language $\mathcal{L}$ with an efficiently computable binary relation $\mathcal{R}$ consists of three ppt algorithms as follows. (1) $\mathsf{NIZK.Setup}(1^\lambda)$ is the setup algorithm and on input a security parameter 1^λ it outputs a common reference string crs; (2) $\mathsf{NIZK.Prove}(\mathsf{crs}, \mathsf{x}, \omega)$ is the proving algorithm and on input a common reference string crs, a statement x and a witness ω it outputs a proof π or a failure symbol $\perp$; (3) $\mathsf{NIZK.Verify}(\mathsf{crs}, \mathsf{x}, \pi)$ is the verification algorithm and on input a common reference string crs, a statement x and a proof π it outputs either true or false.

Perfect completeness. Completeness imposes that an honest prover can always convince an honest verifier that a statement belongs to $\mathcal{L}$, provided that it holds a witness testifying to this fact. We say a NIZK proof is *perfectly complete* if for every (possibly unbounded) adversary $\mathcal{A}$

$$\mathbf{Adv}^{\mathrm{Complete}}_{\mathsf{NIZK},\mathcal{A}}(\lambda) := \Pr\left[\mathrm{Complete}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)\right] = 0\ ,$$

where game $\mathrm{Complete}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)$ is shown in Fig. 3 on the left.

Perfect soundness. Soundness imposes that a malicious prover cannot convince an honest verifier of a false statement. We say a NIZK proof is *perfectly sound* if for every (possibly unbounded) adversary $\mathcal{A}$ we have that

$$\mathbf{Adv}^{\mathrm{Sound}}_{\mathsf{NIZK},\mathcal{A}}(\lambda) := \Pr\left[\mathrm{Sound}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)\right] = 0\ ,$$

```
$\underline{\mathrm{Complete}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)}$:
$\mathsf{crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$
$(\mathsf{x}, \omega) \leftarrow\!\$\ \mathcal{A}(1^\lambda, \mathsf{crs})$
if $(\mathsf{x}, \omega) \notin \mathcal{R}$ return 0
$\pi \leftarrow\!\$\ \mathsf{NIZK.Prove}(\mathsf{crs}, \mathsf{x}, \omega)$
return $\neg(\mathsf{NIZK.Verify}(\mathsf{crs}, \mathsf{x}, \pi))$
```

```
$\underline{\mathrm{Sound}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)}$:
$\mathsf{crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$
$(\mathsf{x}, \pi) \leftarrow\!\$\ \mathcal{A}(1^\lambda, \mathsf{crs})$
return $(\mathsf{x} \notin \mathcal{L} \wedge$
$\mathsf{NIZK.Verify}(\mathsf{crs}, \mathsf{x}, \pi))$
```

```
$\underline{\mathrm{ZK\text{-}Real}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)}$:
$\mathsf{crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$
$b \leftarrow\!\$\ \mathcal{A}^{\mathrm{Prove}}(1^\lambda, \mathsf{crs})$

$\underline{\mathrm{Prove}(\mathsf{x}, \omega)}$:
if $(\mathsf{x}, \omega) \notin \mathcal{R}$ return $\bot$
$\pi \leftarrow\!\$\ \mathsf{NIZK.Prove}(\mathsf{crs}, \mathsf{x}, \omega)$
return $\pi$
```

```
$\underline{\mathrm{ZK\text{-}Ideal}^{\mathcal{A},\mathsf{Sim}}_{\mathsf{NIZK}}(1^\lambda)}$:
$(\mathsf{crs}, \mathsf{tp}) \leftarrow\!\$\ \mathsf{Sim}_1(1^\lambda)$
$b \leftarrow\!\$\ \mathcal{A}^{\mathrm{Prove}}(1^\lambda, \mathsf{crs})$

$\underline{\mathrm{Prove}(\mathsf{x}, \omega)}$:
if $(\mathsf{x}, \omega) \notin \mathcal{R}$ return $\bot$
$\pi \leftarrow\!\$\ \mathsf{Sim}_2(\mathsf{crs}, \mathsf{tp}, \mathsf{x})$
return $\pi$
```

Fig. 3. Games defining the completeness, soundness and zero-knowledge properties of a NIZK proof system.

where game $\mathrm{Sound}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)$ is shown in Fig. 3.

Computational zero knowledge. The zero-knowledge property guarantees that proofs do not leak information about the witnesses that originated them. Technically, this is formalized by requiring the existence of a ppt simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ where Sim_1 takes the security parameter 1^λ as input and outputs a simulated common reference string crs together with a trapdoor tp, and Sim_2 takes the trapdoor as input tp together with a statement $\mathsf{x} \in \mathcal{L}$ for which it must forge a proof π. We say a proof system is *computationally zero knowledge* if, for every ppt adversary $\mathcal{A}$, there exists a simulator Sim such that

$$\mathbf{Adv}^{\mathsf{zk}}_{\mathsf{NIZK},\mathcal{A},\mathsf{Sim}}(\lambda) := \left|\Pr\left[\mathrm{ZK\text{-}Real}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)\right] - \left[\mathrm{ZK\text{-}Ideal}^{\mathcal{A},\mathsf{Sim}}_{\mathsf{NIZK}}(1^\lambda)\right]\right| \in \mathrm{N{\scriptstyle EGL}} ,$$

where games $\mathrm{ZK\text{-}Real}^{\mathcal{A}}_{\mathsf{NIZK}}(1^\lambda)$ and $\mathrm{ZK\text{-}Ideal}^{\mathcal{A},\mathsf{Sim}}_{\mathsf{NIZK}}(1^\lambda)$ are shown in Fig. 3 on the right.

3 Unpredictable Samplers

The privacy notions that we will be developing in the coming sections rely on multistage adversaries that must adhere to certain high-entropy requirements on the sampled circuits. Rather than speaking about specific distributions for specific circuit classes, we introduce a uniform treatment for *any* circuit class via an unpredictability game. Our framework allows one to introduce restricted classes of samplers by imposing structural restrictions on their internal operation *without changes* to the reference unpredictability game. Our framework extends that of Bellare, Stepanov and Tessaro [12] for obfuscators and also models the challenge-generation phase in private functional encryption in prior works [19,20,6,2].

3.1 Definitions

Syntax. A sampler for a circuit family CSp is an algorithm $\mathcal{S}$ that on input the security parameter 1^λ and possibly some state information st outputs a pair of vectors of CSp_λ circuits $(\mathbf{C}_0, \mathbf{C}_1)$ of equal dimension, a pair of vectors of MSp_λ messages $(\mathbf{m}_0, \mathbf{m}_1)$ of equal dimension, and some auxiliary information z. We require the components of the two circuit (resp., message) vectors to be encoded as bit strings of equal length. Input st may encode information about the environment where the sampler is run (e.g., the public parameters of a higher-level protocol) and z models side information available on the sampled circuits or messages.

In the security games we will be considering later on, the goal of adversary will be to distinguish which of two circuit distributions produced by an unpredictable sampler was used to form some cryptographic data (e.g., an obfuscated circuit or an FE token). Our unpredictability definition

```
$\text{mPred}_{\mathcal{S}}^{\mathcal{P}}(1^\lambda)$:
    $(i, \mathsf{m}) \leftarrow\$ \mathcal{P}^{\text{Sam},\text{Func},\text{Sp}}(1^\lambda)$
    $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathsf{list}[i]$
    return $(\mathbf{C}_0(\mathsf{m}) \neq \mathbf{C}_1(\mathsf{m}))$

$\text{Sam}(st)$:
    $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
    $\mathsf{list} \leftarrow \mathsf{list} : (\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1)$
    return $z$
```

```
$\text{Func}(i, \mathsf{m}, \mathsf{C})$:
    $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathsf{list}[i]$
    return $(\mathbf{C}_0(\mathsf{m}), \mathsf{C}(\mathbf{m}_0))$

$\text{Sp}(i, j)$:
    $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathsf{list}[i]$
    $(\mathbf{C}'_0, \mathbf{C}'_1, \mathbf{m}'_0, \mathbf{m}'_1) \leftarrow \mathsf{list}[j]$
    return $\mathbf{C}_0(\mathbf{m}'_0)$
```

```
$\text{Pred}_{\mathcal{S}}^{\mathcal{P}}(1^\lambda)$:
    $(st, st') \leftarrow\$ \mathcal{P}_1(1^\lambda)$
    $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
    $\mathsf{m} \leftarrow\$ \mathcal{P}_2^{\text{Func}}(1^\lambda, \mathbf{C}_0(\mathbf{m}_0), z, st')$
    return $(\mathbf{C}_0(\mathsf{m}) \neq \mathbf{C}_1(\mathsf{m}))$

$\text{Func}(\mathsf{m}, \mathsf{C})$:
    return $(\mathbf{C}_0(\mathsf{m}), \mathsf{C}(\mathbf{m}_0))$
```

Fig. 4. Left: Game defining the multi-instance unpredictability of a sampler $\mathcal{S}$. **Right**: Game defining single-instance unpredictability of a sampler $\mathcal{S}$ against $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2)$.

formalizes the intuition that by examining the input/output behavior of the sampled circuits on messages of choice, the evaluation of legitimate circuits of choice on sampled messages, and the evaluation of sampled circuits on sampled messages, a point leading to differing outputs on some pair of sampled circuits cannot be found. Drawing a parallel to the functional encryption setting, once decryption tokens or encrypted messages become available, the tokens can be used by a legitimate adversary to compute the circuits underneath on arbitrary values, including some special messages that are possibly correlated with the circuits.

Unpredictability. A *legitimate* sampler $\mathcal{S}$ is *statistically multi-instance unpredictable* if for any unbounded *legitimate* predictor $\mathcal{P}$ that places polynomially many queries

$$\mathbf{Adv}_{\mathcal{S},\mathcal{P}}^{\text{mpred}}(\lambda) := \Pr\left[\text{mPred}_{\mathcal{S}}^{\mathcal{P}}(1^\lambda)\right] \in \text{Negl} ,$$

where game $\text{mPred}_{\mathcal{S}}^{\mathcal{P}}(1^\lambda)$ is shown in Fig. 4. Sampler $\mathcal{S}$ is called legitimate if $\mathbf{C}_0(\mathbf{m}'_0) = \mathbf{C}_1(\mathbf{m}'_1)$ for all queries made to the Sp oracle. Predictor $\mathcal{P}$ is legitimate if $\mathsf{C}(\mathbf{m}_0) = \mathsf{C}(\mathbf{m}_1)$ for all queries made to the Func oracle.[4]

The mPred game is multi-instance and the predictor can place polynomially many queries to Sam and set st arbitrarily. The latter essentially ensures that $\mathcal{S}$ generates fresh entropy on any input st. We emphasize that the winning condition demands *component-wise* inequality of circuit outputs. In particular the predictor is *not* considered successful if it outputs a message which leads to different outputs across different Sam queries or within the same Sam query but on different circuit indices.

A number of technical choices have been made in devising this definition. By the legitimacy of the sampler $\mathbf{C}_0(\mathbf{m}_0) = \mathbf{C}_1(\mathbf{m}_1)$ and hence only one of these values is provided to the predictor. Furthermore, since the goal of the predictor is to find a differing input, modifying the experiment so that Func returns $\mathbf{C}_1(\mathsf{m})$ (or both values) would result in an equivalent definition. Our definition intentionally does *not* consider unpredictability of messages. One can ask the predictor to output either a message that results in differing evaluations on challenge circuits or a circuit that evaluates differently on challenge messages. This definition, however, leads to an excessively restrictive unpredictability notion and excludes many circuit samplers of practical relevance.

Composition. A standard guessing argument shows that any (stateless) sampler is multi-instance unpredictable (where $\mathcal{P}$ can place q queries to Sam) if and only if it is single-instance unpredictable (where $\mathcal{P}$ can only place a single Sam query). The reduction in one direction is trivial. In the

[4] We *do not* impose that $\mathbf{C}_0(\mathsf{m}) = \mathbf{C}_1(\mathsf{m})$ within the Func oracle as this is exactly the event that $\mathcal{P}$ is aiming to invoke to win the game. The restriction we do impose allows for a sampler to be unpredictable while possibility outputting low-entropy messages that might even differ on left and right.

other direction we guess the index i^* that the multi-instance predictor $\mathcal{P}$ will output and simulate Sam queries $1, \ldots, (i^* - 1)$ and $(i^* + 1), \ldots, q$ by running the sampler $\mathcal{S}$ in the reduction—this is where we need the stateless property—and answer the i^*th one using the Sam oracle in the single-instance game. Queries to Func with index i^* are answered analogously using the single-instance counterpart whereas those with index different from i^* will use the explicit knowledge of the circuits and messages generated by the reduction. Queries Sp with index (i, j) are answered as follows. If both i and j are different from i^*, use the explicit knowledge of circuits and messages. If $i = i^*$ but $j \neq i^*$, use the explicit knowledge of the messages and the single-instance Func oracle on i^*. If $i \neq i^*$ but $j = i^*$, use the knowledge of the circuit and single-instance Func. For (i^*, i^*) queries, use the Sp oracle in the single-instance game. Note that the legitimacy of the constructed single-instance predictor follows from the legitimacy of the multi-instance predictor *and* the sampler.

Proposition 1 (Unpredictability composition). *A sampler is multi-instance unpredictable if and only if it is single-instance unpredictable.*

Although the samplers that we study in this work are stateless, our framework can be used to analyze *stateful* samplers as well. We leave the study of such samplers as an important (and practically relevant) direction for future work. In Appendix A we define a number of special classes of samplers by imposing structural restrictions on their internal operation. This serves to illustrate how various samplers that previously appeared in the literature can be modeled within our framework. In particular, definitions of high-entropy and block source samplers for keywords [19], block sources for inner products [20], and circuit sampler distributions used in various obfuscation definitions can be seen as particular cases within this framework.

3.2 The case of point functions

We take a moment to discuss the relation between our notion of unpredictability and previous approaches in the literature for point circuits. Previous notions of unpredictability consider distributions on vectors of points whose components have high min-entropy [19,20,6,2]. Such distributions, when viewed as (possibly inefficient) samplers can be shown to be unpredictable according to the above definition.[5] The converse implication, however, does not hold in general. Consider a sampler that returns vectors which contain low-entropy circuits that are identical on left and right. The outputs of such circuits cannot be used to directly distinguish the challenge bit, but the circuits themselves might leak information about other correlated high-entropy challenge components, which might lead to a breach in privacy. It is unclear how previous approaches [19,20,6,2], which adopt a real-or-random formalization, can be modified to model this class of attackers. A similar argument can be made for low-entropy messages. Consider a scenario where a sampler outputs two low-entropy messages that are equal on the left and right.[6] Then, such messages could provide side information that permits distinguishing the high-entropy queries, which is a different type of

[5] Given a predictor $\mathcal{P}$ for such a sampler, run it and return its output while simulating its Func oracle by always answering with 0. This simulation is consistent unless $\mathcal{P}$ queries Func on a message matching the point underlying one of the circuits, an event that immediately contradicts the high min-entropy of the points in the original distribution. Furthermore, if $\mathcal{P}$ outputs a point where two circuits differ, this must be because it guessed one of the points.

[6] In practice, this could correspond, for example, to encryptions of low-entropy data that may or may not be matched by correlated search queries.

privacy breach not previously addressed by existing security models. Therefore, achieving security under our definition of unpredictability is *stronger* than those achieved under previous notions.

Agrawal et al. [2] propose an admissibility definition where only a single challenge circuit is present, there are no correlated messages, no auxiliary information is present, and no state is passed to the sampler. The predictor there is also restricted to run in polynomial time. Although our framework permits modeling such samplers, this definition becomes unsatisfiable for large classes of circuits. Indeed, a function-private FE for a circuit class under this definition immediately leads to a virtual black-box (VBB) obfuscator for the same class, and the latter is known to be impossible to achieve for wide classes of circuits [9,28].[7] Our semi-bounded formulation of a predictor, on the other hand, leads to obfuscation notions that are implied by (and potentially weaker than) *virtual grey-box* (VGB) obfuscation (see Section 4). Negative results for VGB are more restricted and general feasibility results have been recently emerging [14,15]. As mentioned above, Agrawal et al.'s definition does not address correlations either, which is a central point of our work.

4 Obfuscators

Syntax. An obfuscator for a circuit family CSp is a uniform ppt algorithm Obf that on input the security parameter 1^λ and the description of a circuit $\mathsf{C} \in \mathsf{CSp}_\lambda$ outputs the description of another circuit $\overline{\mathsf{C}}$. We require any obfuscator to satisfy the following two requirements.

Functionality preservation : For any $\lambda \in \mathbb{N}$, any $\mathsf{C} \in \mathsf{CSp}_\lambda$ and any $\mathsf{m} \in \mathsf{MSp}_\lambda$, with overwhelming probability over the choice of $\overline{\mathsf{C}} \leftarrow_\$ \mathsf{Obf}(1^\lambda, \mathsf{C})$ we have that $\mathsf{C}(\mathsf{m}) = \overline{\mathsf{C}}(\mathsf{m})$.

Polynomial slowdown : There is a polynomial poly such that for any $\lambda \in \mathbb{N}$, any $\mathsf{C} \in \mathsf{CSp}_\lambda$ and any $\overline{\mathsf{C}} \leftarrow_\$ \mathsf{Obf}(1^\lambda, \mathsf{C})$ we have that $|\overline{\mathsf{C}}| \leq \mathsf{poly}(|\mathsf{C}|)$.

Roughly speaking, security definitions for obfuscators can be divided into the indistinguishability-based and simulation-based notions. Perhaps the most natural notion is the virtual black-box (VBB) property [9], which requires that whatever that can be computed from an obfuscated circuit can be also simulated using oracle access to the circuit. Here we consider a weakening of this notion, known as virtual *grey-box* (VGB) security [14,15] that follows the VBB approach, but allows simulators to run in *unbounded* time, as long as they make polynomially many queries to their oracles; we call such simulators semi-bounded. Below we present a self-composable strengthening of this notion where the VGB property is required to hold in the presence of multiple obfuscated circuits.

In the context of security definitions for obfuscators, we consider samplers that do not output any messages. Furthermore, we call a sampler *one-sided* if its sampled circuits are identical on left and right with probability 1.

Composable VGB. An obfuscator Obf is composable VGB (CVGB) secure if for every ppt adversary $\mathcal{A}$ there exists a semi-unbounded simulator Sim such that for every ppt one-sided circuit sampler $\mathcal{S}$ the advantage

$$\mathbf{Adv}^{\mathrm{cvgb}}_{\mathsf{Obf},\mathcal{S},\mathcal{A},\mathsf{Sim}}(\lambda) := \left|\Pr\left[\text{CVGB-Real}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)\right] - \Pr\left[\text{CVGB-Ideal}^{\mathcal{S},\mathsf{Sim}}_{\mathsf{Obf}}(1^\lambda)\right]\right| ,$$

[7] Such impossibility results hold even in the case where no composition is considered, i.e., where only a single circuit is obfuscated. Note also that if auxiliary information is made available, the definition becomes unachievable for even simpler circuits: VBB multi-bit point obfuscation with auxiliary information is impossible if indistinguishability obfuscation exists [22].

```
$\text{CVGB-Real}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)$:
    $(\mathbf{C}, z) \leftarrow\$ \mathcal{S}(1^\lambda, \varepsilon)$
    $\overline{\mathbf{C}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{C})$
    $b \leftarrow\$ \mathcal{A}(1^\lambda, \overline{\mathbf{C}}, z)$
    return $b$
```

```
$\text{CVGB-Ideal}^{\mathcal{S},\mathsf{Sim}}_{\mathsf{Obf}}(1^\lambda)$:
    $(\mathbf{C}, z) \leftarrow\$ \mathcal{S}(1^\lambda, \varepsilon)$
    $b \leftarrow\$ \mathsf{Sim}^{\textsc{Func}}(1^\lambda, 1^{|\mathbf{C}|}, z)$
    return $b$

$\textsc{Func}(\mathsf{m})$:
    return $\mathbf{C}(\mathsf{m})$
```

```
$\text{DI}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)$:
    $b \leftarrow\$ \{0, 1\}$
    $b' \leftarrow\$ \mathcal{A}^{\textsc{Sam}}(1^\lambda)$
    return $(b = b')$

$\textsc{Sam}(st)$:
    $(\mathbf{C}_0, \mathbf{C}_1, z) \leftarrow\$ \mathcal{S}(1^\lambda, st)$
    $\overline{\mathbf{C}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{C}_b)$
    return $(\overline{\mathbf{C}}, z)$
```

```
$\text{OW}^{\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)$:
    $1^t \leftarrow\$ \mathcal{A}_1(1^\lambda)$
    $\mathsf{w} \leftarrow\$ \mathsf{MSp}_\lambda$
    $\mathbf{C} \leftarrow [\mathsf{C}[\mathsf{w}], \ldots, \mathsf{C}[\mathsf{w}]]$ // $t$ copies
    $\overline{\mathbf{C}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathbf{C})$
    $\mathsf{w}' \leftarrow\$ \mathcal{A}_2(1^\lambda, \overline{\mathbf{C}})$
    return $(\mathsf{w} = \mathsf{w}')$
```

Fig. 5. Games defining the CVGB, DI and OW security of an obfuscator. One-way security is defined for point-functions only.

is negligible, where games $\text{CVGB-Real}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(\lambda)$ and $\text{CVGB-Ideal}^{\mathcal{S},\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)$ are shown in Figure 5.

By considering samplers that only output a single circuit we recover the standard (worst-case) VGB property. The VBB property corresponds to the case where the simulator is required to run in polynomial time. Average-case notions of obfuscation correspond to definitions where the circuit samplers are fixed. A results of Bitansky and Canetti [14, Proposition A.3] on the equivalence of VGB with and without auxiliary information can be easily shown to also hold in the presence of multiple circuits, from which one can conclude that CVGB with auxiliary information is the same as CVGB without auxiliary information.

We also introduce the following adaptation of an indistinguishability-based notion of obfuscation introduced in [14] for point functions.

Distributional indistinguishability. An obfuscator Obf is DI secure if, for every unpredictable ppt sampler $\mathcal{S}$ and every ppt adversary $\mathcal{A}$,

$$\mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\text{DI}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)\right] - 1 \in \textsc{Negl} \ ,$$

where game $\text{DI}^{\mathcal{S},\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)$ is defined in Fig. 5.

The above definition strengthens that in [14] and gives the sampler the possibility to leak auxiliary information to the adversary. In particular, we can consider the case where images of an (internally generated) vector of messages that are correlated with the circuits are provided to $\mathcal{A}$. (Our constructions will rely on this property for point obfuscators.) Throughout the paper we consider DI adversaries that place a single query to the Sam oracle. It can easily be shown that the DI self-composes for stateless samplers, meaning that security against adversaries that place one Sam query is equivalent to the setting where an arbitrary number of queries are allowed. Note also that we allow the adversary to pass some state information st to the sampler. Security with respect to *all* ppt and statistically unpredictable samplers can be shown to be equivalent to a weaker variant where the adversary is run *after* the sampler and $st := \varepsilon$.

Following a similar argument to the first part of the proof in [14, Theorem 5.1], we show in Appendix B that CVGB with auxiliary information and for a any circuit family implies distributional indistinguishability with auxiliary information for the same circuit family. Hence, our notion is potentially *weaker* than CVGB. This proof crucially relies on the restriction that samplers are required to be unpredictable even in the presence of unbounded predictors. The proof of the converse direction in [14, Theorem 5.1] for point function (and without auxiliary information) uses techniques specific to point functions and we leave a generalization to wider classes of circuits for future work.

Proposition 2 (CVGB $\implies$ DI)**.** *Any* CVGB *obfuscator for a class of circuits* CSp *and with respect to all ppt samplers is also* DI *secure for the same class* CSp.

We conclude this discussion by introducing a new notion of *one-way* point obfuscation that requires it to be infeasible to recover the point given many obfuscations of it.

One-way point obfuscation. Let Obf be an obfuscator for a point circuit family CSp. We say Obf is OW secure if for every ppt adversary $\mathcal{A}$

$$\mathbf{Adv}^{\mathrm{ow}}_{\mathsf{Obf},\mathcal{A}}(\lambda) := \Pr\left[\mathrm{OW}^{\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)\right] \in \textsc{Negl} \ ,$$

where game $\mathrm{OW}^{\mathcal{A}}_{\mathsf{Obf}}(1^\lambda)$ is shown in Fig. 5 on the right.

The next proposition, proved in Appendix C, shows that OW is a weakening of DI for point circuits.

Proposition 3 (DI $\implies$ OW **for point circuits).** *Let* Obf *be an obfuscator for a point circuit family* CSp*. If* Obf *is* DI *secure with respect to all ppt samplers, then it is also* OW *secure.*

Instantiations and obfuscation-based attacks. A concrete instantiation of a CVGB obfuscator for point functions *with* auxiliary information (AI) is given by Bitansky and Canetti [14]. This construction is based on the hardness of a variant of the DDH assumption called strong vector-DDH (SVDDH) assumption. The SVDDH assumption is an assumption that is formulated *without* reference to any auxiliary information. Recently, Bellare, Stepanovs and Tessaro [13] have shown that the SVDHH assumption (and *verifiable* point obfuscation) in presence of arbitrary AI is in contention with the existence of VGB obfuscation for general circuits (that is, one of the two cannot exist). We take a moment to clarify how these two results relate to each other. In this discussion we assume that all obfuscation notions are considered for a single circuit only (i.e., we do not consider composability). First, note that the notion of AIPO (auxiliary-information point obfuscation) used in [13] follows a notion equivalent to distributional indistinguishability where the right distribution is fixed to be uniform. As shown in [14, Theorem 5.1] any point obfuscation (without AI) is equivalent to VGB point obfuscation. It is also shown in [14, Proposition A.3] that VGB obfuscation without AI is equivalent to VGB obfuscation *with* AI for any circuit class. (Intuitively, to construct a simulator that works for all possible AI, one uses the fact that the simulator is unbounded to find the best simulator that works for a non-uniform adversary that takes a value of the AI as advice.) Together with Proposition 2 above we get that all these notions (in their non-composable variants) are *equivalent*. This then raises the question whether the results of [13] are also in contention with PO without AI. To see that this does not follow from the equivalence of notions, note that in Proposition 2 we crucially rely on a predictor that runs a possibly unbounded simulator. Put differently, the AI must be *statistically* unpredictable. Indeed, the results of [13] rely on special forms of AI which only computationally hide the sampled point. (Roughly speaking, the AI contains a VGB obfuscation of a (non-point) circuit that depends on the sampled point.) To avoid such feasibility problems, and in line with the above equivalence results, we constrain auxiliary information to be statistically unpredictable throughout this work.

Inner products. In Appendix G we build on the results of [14,23] to construct a DI-secure obfuscator for the hyperplane membership (aka. inner product) functionality under a generalization of the strong vector DDH assumption used in [14]. In our analysis, we assume that the assumption holds in the presence of random auxiliary information, which immediately translates to an obfuscator that also tolerates this type of leakage. Once again, statistical unpredictability rules out iO and VGB-based attacks.

5 Function Privacy: A Unified Approach

We now define what function privacy for general functional encryption schemes means and derive the model specific to keyword search schemes by restriction to point circuit families. Our definition follows the indistinguishability-based approach to defining FE security and comes with an analogous legitimacy condition that prevents the adversary from learning the challenge bit simply by extracting a token for a circuit that has differing outputs for the left and right challenge messages. The model extends the IND-CPA game via a left-or-right (LR) oracle that returns ciphertexts *and* tokens for possibly correlated keywords and messages. Since the adversary in this game has access to tokens that depend on the challenge bit, we use the unpredictability framework of Section 3 to rule out trivial attacks.

The game follows a left-or-right rather than a real-or-random formulation of the challenge oracle [19,20,6,2] as this choice frees the definition from restrictions that must be imposed to render samplers compatible with uniform distribution over circuits. (In particular, it allows analyzing security under repetitions of keywords.)

The sampler allows us to model within a single game token-only adversarial strategies via samplers that output no messages, those which obtain encrypted messages that match the extracted keywords (the enhanced privacy notion of [19]), as well as *messages that are arbitrarily correlated with the keywords* and which might not match any of them. Our model unify previous models in this area [19,6] and when restricted to point circuit families it gives rise to a new privacy notion that offers significant improvements over those in prior work [19,6].

The definition also provides the adversary with the ability to adaptively obtain multiple challenges and tokens. However, similarly to unpredictability, a hybrid argument shows that for (stateless) samplers the definition self-composes and we consider the simpler single-shot game in the remainder of the paper.

PRIV SECURITY. A functional encryption scheme FE is PRIV secure if, for every unpredictable ppt sampler[8] $\mathcal{S}$ and every ppt adversary $\mathcal{A}$

$$\mathbf{Adv}^{\text{priv}}_{\mathsf{FE},\mathcal{A},\mathcal{S}}(\lambda) := 2 \cdot \Pr\left[\text{PRIV}^{\mathcal{A},\mathcal{S}}_{\mathsf{FE}}(1^\lambda)\right] - 1 \in \textsc{Negl} \ ,$$

where game $\text{PRIV}^{\mathcal{A},\mathcal{S}}_{\mathsf{FE}}(1^\lambda)$ is defined in Fig. 6. We exclude adversaries $(\mathcal{A},\mathcal{S})$ that attempt to trivially win the PRIV game via decryption tokens, by either extracting them explicitly via the token-generation oracle, or implicitly via the left-or-right oracle. Formally, the pair $(\mathcal{A},\mathcal{S})$ is legitimate if, with overwhelming probability

$$\forall(\mathbf{C}_0,\mathbf{C}_1) \in \mathsf{TList}\,, \forall(\mathbf{m}_0,\mathbf{m}_1) \in \mathsf{MList} : \mathbf{C}_0(\mathbf{m}_0) = \mathbf{C}_1(\mathbf{m}_1) \ .$$

Note also that for two sampler classes $\mathbb{S}_1$ and $\mathbb{S}_2$ with $\mathbb{S}_1 \subseteq \mathbb{S}_2$ security with respect to samplers in $\mathbb{S}_2$ is a stronger security guarantee that one for those only in $\mathbb{S}_1$. In particular a stronger restriction on sampler classes results in a weaker definition.

[8] We limit samplers to ppt because in proving the security of our constructions, samplers are used to construct computational adversaries against functional encryption schemes. In general, one could consider unbounded samplers.

```
$\mathrm{PRIV}_{\mathsf{FE}}^{\mathcal{A},\mathcal{S}}(1^\lambda)$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
return $(b = b')$
```

```
$\mathrm{LR}(st)$:
$(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
$\mathsf{TList} \leftarrow \mathsf{TList} : (\mathbf{C}_0, \mathbf{C}_1)$
$\mathsf{MList} \leftarrow \mathsf{MList} : (\mathbf{m}_0, \mathbf{m}_1)$
$\mathbf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, \mathbf{C}_b)$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathbf{m}_b)$
return $(\mathbf{tk}, \mathbf{c}, z)$
```

```
$\mathrm{TGen}(\mathsf{C})$:
$\mathsf{TList} \leftarrow \mathsf{TList} : (\mathsf{C}, \mathsf{C})$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, \mathsf{C})$
return $\mathsf{tk}$
```

Fig. 6. Game defining enhanced privacy of a functional encryption scheme FE.

Restricted PRIV and PRIV-TO. We call an adversary *token-only* if $\mathcal{S}$ does not output any messages, and call the resulting security notion PRIV-TO. Note that, for token-only adversaries, the additional legitimacy constraint above is redundant. We call an adversary *restricted* if for every second-phase TGen query C_2 there is a first-phase TGen query C_1 such that $\mathsf{C}_2(\mathbf{m}_b) = \mathsf{C}_1(\mathbf{m}_b)$ for $b \in \{0,1\}$. Intuitively, this amounts to imposing that images exposed via second-stage queries (i.e. queries placed *after* receiving the challenge) can reveal no more than the images obtained in the first stage (i.e. from queries placed *before* receiving the challenge). We call the resulting security notion Res-PRIV. We emphasize that the Res-PRIV model inherits many of the strengths of the full PRIV model such as arbitrary correlations and a wide range of adaptive token queries.[9]

Keyword search. Two important aspects of our definition are that it considers (1) correlated *unpaired* encrypted messages in the system that do not match any of the searched keywords; and (2) *low-entropy* messages/keywords that are correlated with the high-entropy searches whose privacy must be protected. The former aspect entails that the full equality pattern between challenge messages and keywords may remain hidden from the adversary (and hence a wider class of non-trivial attacks can be launched). Indeed, the adversary always obtains the image matrix resulting from evaluating tokens on ciphertexts, and hence sees the equality pattern between *paired* challenge keywords and messages. But this information will *not* include the repetition patterns among unpaired messages or unpaired keywords, which may reveal sensitive information as well. Low entropy messages and keywords model the presence of ciphertexts and tokens in the system, over which the uncertainty of the adversary may be small, but which are correlated with sensitive data that must still be protected. Indeed, our unpredictability notion allows the sampler to output such low-entropy keywords and messages as long as low-entropy keywords are equal on left and right. A real-or-random modeling of this setting cannot capture this scenario. When low-entropy messages differ on the left and right, the adversary cannot learn them via the TGen oracle due to the legitimacy condition: imposing that they are not leaked maps to IND-CPA security. When they are equal on the left and right, they can be learned by successive queries to the token extraction oracle, which permits capturing attack scenarios where *adaptive* searches over low entropy correlated messages may be carried out. In particular, this permits an adversary to recover a correlated repetition search pattern *after* the PRIV challenge has been revealed. As a result, low-entropy messages and keywords are tolerated, even when correlated with other messages or keywords. Furthermore, the values and equality patterns of high-entropy keywords are protected, as well as those of all encrypted messages for which a token was not explicitly extracted. Our main results in Sections 6.3 and 6.4 show the existence of keyword search schemes which are secure in the aforementioned scenarios.

On revealing images. The outputs of challenge circuits on challenge messages can be always computed by the adversary, and by imposing equality of images we ensure that they do not lead to

[9] When the restriction here is imposed on the IND-CPA model for point function, the resulting model remains as strong as the full IND-CPA model.

$$\begin{tikzcd}
 & & \mathrm{ResPRIV} \arrow[dl, dashed] \arrow[dr] & \\
\mathrm{PRIV} \arrow[r] & \mathrm{ResPRIV} \wedge \mathrm{CPA} \arrow[ur] \arrow[dr] & & \mathrm{PRIV\text{-}TO} \\
 & & \mathrm{PRIV\text{-}TO} \wedge \mathrm{CPA} \arrow[ur] \arrow[d] & \\
 & & \mathrm{CPA} &
\end{tikzcd}$$

Fig. 7. Relations among security notions for private functional encryption. The dotted implication only holds for keyword search schemes as weak (aka. restricted) and standard IND-CPA security models are equivalent for point circuits.

trivial distinguishing attacks. (This is similar to the legitimacy condition in FE security models.) It is however less clear why these image values should be explicitly provide to the predictor in the unpredictability game, even when they are equal for left and right circuits-messages pairs. To see this, consider the sampler that for a random word w outputs

$$\mathsf{w}_0 = \mathsf{w}, \quad \mathsf{w}_1 = \overline{\mathsf{w}}, \quad \mathsf{m}_{0,i} := \begin{cases} \mathsf{w} & \text{if } \mathsf{w}[i] = 0 \ ; \\ \overline{\mathsf{w}} & \text{otherwise,} \end{cases} \quad \text{and} \quad \mathsf{m}_{1,i} := \begin{cases} \overline{\mathsf{w}} & \text{if } \mathsf{w}[i] = 0 \ ; \\ \mathsf{w} & \text{otherwise.} \end{cases}$$

Note that $\mathsf{C}[\mathsf{w}_0](\mathsf{m}_{0,i}) = \mathsf{C}[\mathsf{w}_1](\mathsf{m}_{1,i}) = \mathsf{w}[i]$ and hence the images are equal on left and right. Word w_0 can be recovered bit by bit from the image values $\mathsf{C}[\mathsf{w}_b](\mathsf{m}_{0,b})$ and computing $1 - \mathsf{C}[\mathsf{w}_b](\mathsf{w}_0)$ would then reveal the challenge bit b. Finally, without access to the images $\mathsf{C}[\mathsf{w}_0](\mathsf{m}_{0,i})$ the sampler can be shown to be unpredictable as w is chosen randomly. On the other hand, in the presence of images, the sampler is trivially predicable. This counterexample is similar to that briefly discussed in [6] and can be modified to show that the *enhanced* model of Boneh, Raghunathan and Segev [19] for the so-called $(k_1, \ldots, k_T)$-distributions is not achievable.

Relations among notions. Clearly PRIV implies its weaker variant Res-PRIV, which in turn implies PRIV-TO. It is not too difficult to see that PRIV also implies IND-CPA.[10] A noteworthy consequence of this is that for *all-or-nothing* functionalities (such as PEKS, IBE or ABE) any PRIV-secure construction is also *index hiding* (aka. anonymous), whereby ciphertexts do not leak any information about their intended recipients (i.e., about tokens that may permit recovering the payload). Res-PRIV would imply a restricted analogue of IND-CPA (where images in the second phase should match one in the first phase), which for point functions is equivalent to the standard IND-CPA model. IND-CPA security does not imply PRIV-TO: consider an IND-CPA-secure scheme that is modified to append circuits in the clear to their tokens. PRIV-TO does not imply IND-CPA either: consider a PRIV-TO-secure scheme that is modified to return messages in the clear with ciphertexts. (Note that these separations hold even for point functions.) Figure 7 summarizes relations among notions of security.

[10] Consider a sampler which does not output any circuits and simply returns (possibly low-entropy) messages included in the state *st* passed to it. This sampler is trivially unpredictable. Furthermore, the legitimacy conditions in the two games exactly match.

6 Constructions

6.1 The Obfuscate-Extract (OX) transform

Our first construction formalizes the intuition that obfuscating circuits before computing a token for them will provide some form of token privacy.

The OX transform. Let Obf be an obfuscator supporting a circuit family CSp and let FE be a functional encryption scheme supporting all polynomial-size circuits. We construct a functional encryption scheme $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$ via the OX transform as follows. Setup, encryption and evaluation algorithms are identical to those of the base functional encryption scheme. The token-generation algorithm creates a token for the circuit that results from obfuscating the extracted circuit, i.e., $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}].\mathsf{TGen}(\mathsf{msk}, \mathsf{C}) := \mathsf{FE}.\mathsf{TGen}(\mathsf{msk}, \mathsf{Obf}(1^\lambda, \mathsf{C}))$. Correctness of this construction follows from those of its underlying components. We now show that this construction yields function privacy against PRIV-TO adversaries. Since PRIV-TO does not imply IND-CPA security—see the discussion in Section 5—we establish IND-CPA security independently. The proof of the following theorem is straightforward and results from direct reductions to the base FE and Obf schemes used in the construction. A formal proof can be found in Appendix E.

Theorem 1 (OX **is** PRIV-TO $\wedge$ IND-CPA)**.** *If obfuscator* Obf *is* DI *secure, then scheme* $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$ *is* PRIV-TO *secure. Furthermore, if* FE *is* IND-CPA *secure* $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$ *is* IND-CPA *secure.*

We note that this proof holds for arbitrary classes of circuits and arbitrary (circuits-only) samplers. Using the composable VGB point-function obfuscator of Bitansky and Canetti [14] and any secure functional encryption scheme that is powerful enough to support one exponentiation and one equality test (e.g., supports $\mathbf{NC}^1$ circuits) we obtain a private keyword search scheme in the presence of tokens for arbitrarily correlated keywords. If the underlying functional encryption scheme supports the more powerful functionality that permits attaching a payload to the point, one obtains a PRIV-TO anonymous identity-based encryption scheme where arbitrary correlations are tolerated. In this case, on input $(\mathsf{ID}, \mathsf{m})$, the functionality supported by the underlying FE scheme would return m if $\overline{\mathsf{C}}(\mathsf{ID}) = 1$, where $\overline{\mathsf{C}}$ was sampled from $\mathsf{Obf}(\mathsf{C}[\mathsf{ID}^\star])$ during token generation; it would return $\perp$ otherwise.

The above theorem shows that DI is sufficient to build a PRIV-TO scheme. It is however easy to see that the existence of a single-circuit DI obfuscator is also necessary. Indeed, given any PRIV-TO scheme FE we can DI-obfuscate a single circuit C by generating a fresh FE key pair, and outputting $\mathsf{FE}.\mathsf{Eval}(\cdot, \mathsf{tk})$ where tk is a token for C. A formal proof of this argument appears in Appendix D.

Proposition 4 (PRIV-TO **vs.** DI)**.** *A* PRIV-TO*-secure functional encryption for a circuits family* CSp *exists if a* DI *obfuscator for* CSp *exists. Conversely, a single-circuit* DI *obfuscator for* CSp *exists if a* PRIV-TO*-secure functional encryption for* CSp *exists.*

A similar line of reasoning shows that the extractor-based constructions of private FE by Boneh, Raghunathan and Segev [19] and Arriaga, Tang and Ryan [6] give rise to single-circuit DI obfuscators for point functions for the specific classes of samplers considered in those works.

Agrawal et al. [2] have proposed a simulation-based definition of privacy that strikes a different balance between practical relevance and feasibility. However, the definition in [2] implies VBB obfuscation, which is known to be feasible only for restricted classes of circuits [16,7], in idealized models of computation [24,17,8] or with restricted forms of auxiliary information. The above proposition shows that our model is closer to the weaker form of DI obfuscation, which as shown in Proposition 2 is implied by VGB (and hence VBB) obfuscation, and is therefore more amenable to instantiations in the standard model.

It is possible to show via separating counterexamples that IND-CPA and PRIV-TO do not jointly suffice to give PRIV security. (Roughly speaking, one considers a malicious token-generation oracle that on specially crafted words leads to a break of PRIV security.) We will explore the feasibility of PRIV security in the next sections.

6.2 The Trojan-Obfuscate-Extract (TOX) transform

We now present a generic construction that achieves Res-PRIV security for a class of samplers that we call *concentrated*. To this end, we build on the ideas from [5,25] on converting selective to adaptive security and achieving simulation-based security from IND-CPA security for FE schemes.

The TOX transform. Given a symmetric encryption scheme SE, a general-purpose obfuscator Obf and a functional encryption FE for all circuits, our *Trojan-Obfuscate-Extract* (TOX) transform operates as follows. The master public key of the scheme is the same as that of the base FE scheme. Its master secret key includes a symmetric key k and the master secret key for the base FE scheme. To encrypt a message m we call the base FE encryption routine on $(0, 0^\lambda, \mathsf{m})$. To generate a token for a circuit C, we first generate an obfuscation $\bar{\mathsf{C}} \leftarrow_\$ \mathsf{Obf}(\mathsf{C})$, a ciphertext $\mathsf{c} \leftarrow_\$ \mathsf{SE.Enc}(\mathsf{k}, 0^n)$ and construct the following circuit.

$$\mathsf{Troj}[\bar{\mathsf{C}}, \mathsf{c}](\mathsf{b}, \mathsf{k}, \mathsf{m}) := \begin{cases} \bar{\mathsf{C}}(\mathsf{m}) \text{ if } \mathsf{b} = 0 \ ; \\ \mathsf{C}^*(\mathsf{m}) \text{ if } \mathsf{b} = 1, \text{ where } \mathsf{C}^* = \mathsf{SE.Dec}(\mathsf{k}, \mathsf{c}) \ . \end{cases}$$

Finally, we extract a token for $\mathsf{Troj}[\bar{\mathsf{C}}, \mathsf{c}]$. Evaluation simply invokes the corresponding operation in the underlying FE.

The correctness and IND-CPA security of this construction follow easily from the correctness and IND-CPA security of the underlying functional encryption scheme via straightforward reductions. Intuitively, during the normal operation of the scheme, the tokens in the construction will simply evaluate an obfuscation of the extracted circuit. In the proof of privacy, however, we will take advantage of the fact that a totally independent circuit can be hidden inside the token within the symmetric encryption ciphertext, and unlocked by a message containing the correct symmetric decryption key. For the proof to go through, the hidden circuit must be carefully selected so that the legitimacy condition is observed throughout. In order to meet this latter restriction, we consider the following constrained class of samplers.

Concentrated samplers. We say a sampler $\mathcal{S}$ is $\mathcal{S}^*$-concentrated if for all st, all CSp_λ-vectors $\mathbf{C}$ we have that

$$\Pr\left[\mathbf{C}(\mathbf{m}_0) = \mathbf{C}(\mathbf{m}_1) \neq \mathbf{C}(\mathbf{m}^*)\right] \in \text{NEGL and } \Pr\left[\mathbf{C}_0(\mathbf{m}_0) \neq \mathbf{C}^*(\mathbf{m}^*)\right] \in \text{NEGL} \ ,$$

where the probability space of these is defined by the operations $(\mathbf{C}^*, \mathbf{m}^*) \leftarrow_\$ \mathcal{S}^*(z, \mathbf{C})$ and $(\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow_\$ \mathcal{S}(st)$.

Concentration is a property independent of unpredictability and we will be relying on both in our construction. Unpredictability is used in the reduction to the DI assumption. Concentration guarantees the existence of a sampler $\mathcal{S}^*$ that generates circuits $\mathbf{C}^*$ and messages $\mathbf{m}^*$ which permit decoupling circuits and messages in the security proof. Intuitively, quantification over all $\mathbf{C}$ means that adversarially generated circuits will lead to image matrices that collide with those leaked by the sampler with overwhelming probability. The additional restriction on $\mathbf{C}^*(\mathbf{m}^*)$ guarantees that one can switch from the honest branch of challenge tokens to one corresponding to the trojan branch. Both of these properties are important to guarantee legitimacy when making a reduction to the security of the FE scheme. We however need to impose that legitimacy also holds for second-phase TGen queries as well, and this is where we need to assume Res-PRIV security: the extra legitimacy condition allows us to ensure that by moving to $\mathbf{m}^*$ the legitimacy condition is not affected in the second phase either. Finally, an important observation is that, because we are dealing with concentrated samplers, our security proof goes through assuming obfuscators that need only tolerate random auxiliary information.

Theorem 2 (TOX is Res-PRIV). *If obfuscator Obf is DI secure, SE is IND-CPA secure and FE is IND-CPA secure, then scheme $\mathsf{TOX}[\mathsf{FE}, \mathsf{Obf}, \mathsf{SE}]$ is Res-PRIV secure with respect to concentrated samplers.*

Proof. The proof proceeds via a sequence of three games as follows.

Game$_0$: This game is identical to PRIV: challenge vector $\mathbf{C}_b$ is extracted and $\mathbf{m}_b$ is encrypted for an random bit b and for all TGen queries, string 0^n is encrypted using SE in the trojan branch.

Game$_1$: In this game, instead of 0^n we encrypt the circuits queried to the (first or second-phase) TGen oracle under a symmetric key k^* in the trojan branch. In the challenge phase, we sample $(\mathbf{C}^*, \mathbf{m}^*) \leftarrow_{\$} \mathcal{S}^*(z, \mathbf{C})$, where $\mathbf{C}$ are all first-phase TGen queries, and encrypt $\mathbf{C}^*$ under k^* for the challenge circuits in the trojan branch. This transition is negligible down to IND-CPA security of SE.

Game$_2$: In this game, instead of encrypting $(0, 0, \mathbf{m}_b)$ we encrypt $(1, \mathsf{k}^*, \mathbf{m}^*)$ in the challenge phase where the latter is generated using $\mathcal{S}^*(z, \mathbf{C})$. We reduce this hop to the IND-CPA security of FE. We generate a key k^*, answer first-stage TGen queries using the provided TGen oracle and encrypt circuits under k^* in the trojan branch to get st. We run $\mathcal{S}(st)$ and get $(\mathbf{C}_0, \mathbf{m}_0, \mathbf{C}_1, \mathbf{m}_1, z)$. We then run $\mathcal{S}^*(z, \mathbf{C})$, where $\mathbf{C}$ are all first-phase TGen queries, to get $(\mathbf{C}^*, \mathbf{m}^*)$. We prepare challenges tokens by encrypting $\mathbf{C}^*$ under k^* in the trojan branch and using the provided TGen oracle we generate the challenge tokes. We query the provided LR on $(0, 0, \mathbf{m}_b)$ and $(1, \mathsf{k}^*, \mathbf{m}^*)$ and receive the corresponding vector of ciphertexts. Second-stage TGen queries are handled using provided TGen oracle and k^*. Finally, we return the same bit that the distinguisher returns. Legitimacy of first-stage TGen queries follows from the first condition on concentration that with high probability $\mathbf{C}(\mathbf{m}_b) = \mathbf{C}(\mathbf{m}^*)$. For the challenge tokens, this follows from the second concentration requirement that $\mathbf{C}_b(\mathbf{m}_b) = \mathbf{C}^*(\mathbf{m}^*)$. For the second-stage queries we rely on the restriction on the adversary. Recall that in the Res-PRIV model, any second-stage queries must have an image vector which matches one for a first-stage query. Since the first-stage images match those on $\mathbf{m}^*$ (and hence are legitimate), the second-stage ones will be also legitimate. We output $(b' = b)$ where the distinguisher outputs b'. As a result of this game, the challenge messages no longer depend on b. It is easy to see that according to the IND-CPA challenge bit this reduction interpolates between games Game$_1$ and Game$_2$.

Game$_3$: In this game we use $\mathbf{C}_1$ in challenge token generation even if $b = 0$. We show this hop in unnoticeable down to the security of the obfuscator. We sample an FE key pair and a symmetric key and simulating the first-stage TGen queries for the distinguisher as before. We define a DI sampler that outputs the circuits that the PRIV sampler outputs, but extends the circuit list to include another copy of $\mathbf{C}_1$ on both sides. This sampler also outputs as auxiliary information z' the original auxiliary information output by the PRIV sampler, extended with the random coins used to generate the FE key, the symmetric key, and to run the first stage of the adversary (this will allow the second stage DI adversary to reconstruct the keys and first stage TGen queries). It follows that this sampler is unpredictable as long as the PRIV sampler is. When we receive the obfuscations and z', we generate $(\mathbf{C}^*, \mathbf{m}^*) \leftarrow_\$ \mathcal{S}^*(z, \mathbf{C})$, where $\mathbf{C}$ are all first-phase TGen queries. We form the challenge tokens using the received obfuscations and $\mathbf{C}^*$, taking the $\mathbf{C}_1$ obfuscations from the duplicated part of the challenge, and the $\mathbf{C}_0$ obfuscations from the original part (these can now be either $\mathbf{C}_0$ or $\mathbf{C}_1$ depending on the external challenge bit). Challenge ciphertexts are generated by encrypting $\mathbf{m}^*$ (rules of Game_2). We answer the second-stage TGen queries using the FE key and the symmetric key. We return whatever the distinguisher returns. It is easy to see that according to the DI challenge bit this reduction interpolates between games Game_2 and Game_3.

In Game_3 both the challenge tokens and challenge ciphertexts are independent of the bit b and hence the advantage of any adversary is 0. □

Examples. Consider keyword samplers which output high-entropy keywords and messages with arbitrary image matrices. All such samplers are concentrated around a sampler $\mathcal{S}^*$ that outputs uniformly random keywords and messages subject to the same image pattern. The second concentration condition is immediate and the first follows from the fact that all messages and circuits have high entropy and $\mathbf{C}$ is selectively chosen. Although this argument can be extended to samplers outputting low-entropy keywords whose complete image matrix is predictable or is included in z, the latter requirement may not always be the case in general. Consider, for example, a vector $\mathbf{C}$ consisting of circuits for $\mathsf{w} = 0^n$ and messages $\mathbf{m}_0 = \mathbf{m}_1$ whose components are randomly set to 0^n and 1^n. The image matrix in this setting is unpredictable as long as a sufficiently large number of messages are output.

As another example, consider inner-product circuits $\mathsf{C}[\mathbf{v}](\mathbf{w})$ that return 1 iff $\langle \mathbf{v}, \mathbf{w} \rangle = 0 \pmod p$ for a prime p. Samplers which output n vectors $\mathbf{v}_i \in \mathbb{Z}_p^d$ and m messages $\mathbf{w}_i \in \mathbb{Z}_p^d$ where all vector entries have high entropy can be easily shown to be unpredictable. Given the corresponding $n \times m$ image matrix, whenever $d(n+m) > nm$ a high-entropy pre-image to the image matrix can be sampled as the system will be under-defined. Under this condition, the second requirement needed for concentration is met, and the first condition follows as this pre-image is high entropy and $\mathbf{C}$ is selectively chosen. This observation implies that a DI obfuscator for the hyperplane membership problem will immediately yield a private functional encryption scheme for the same functionality under arbitrary correlations via the TOX construction, a problem that was left open in [20]. In Appendix G we give a direct construction of a DI obfuscator for hyperplane membership by proving that the obfuscator of Canetti, Rothblum and Varia [23] is DI secure in the presence of random auxiliary information under a variant of the DDH assumption in the style of those used in [23,14].

6.3 The Disjunctively-Obfuscate-Extract (DOX) transform

In this section we present a specialization of the TOX transform for point functions. We are able to remove the limitation that security holds only against concentrated samplers, and achieve privacy in the presence of arbitrary correlations between searched keywords and encrypted messages. Our construction demands less from the underlying functional encryption and obfuscator and hence can potentially allow more efficient instantiations of these primitives.

The DOX transform. Let Obf be an obfuscator supporting a point circuit family CSp over message space MSp. Let FE be a functional encryption scheme supporting general circuits, and let PRP be a pseudorandom permutation (see Subsection 2.1 for the formal definition). We construct a keyword search scheme KS for keyword space $\mathsf{WSp} = \mathsf{MSp}$ via the *Disjunctively-Obfuscate-Extract* (DOX) transform as follows. The key-generation algorithm samples a PRP key $\mathsf{k} \leftarrow_{\$} \mathsf{K}(1^\lambda)$ and an FE key pair $(\mathsf{msk}, \mathsf{mpk}) \leftarrow_{\$} \mathsf{FE.Gen}(1^\lambda)$. It returns $((\mathsf{k}, \mathsf{msk}), \mathsf{mpk})$. The encryption operation is identical to that of the FE scheme. The test algorithm is identical to the evaluation algorithm of FE. The token-generation algorithm computes

$$\mathsf{FE.TGen}(\mathsf{msk}, \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{E}(\mathsf{k}, \mathsf{w})])) \ .$$

The FE-extracted circuits are two-point circuits implemented as the disjunction of two obfuscated point functions. One of the points will correspond to the searched query, whereas the other point will be pseudorandom and will be only used for proofs of security. (In a loose sense, the second point represents the second branch in the TOX construction.) As in OX, the composable VGB obfuscator of [14] for point functions and any general-purpose functional encryption scheme (such as those in [29,26]) can be used to instantiate the above construction. The supported circuit class would roughly amount to two parallel group operations and two comparisons in a DDH group.

In Appendix F.1 we show that DOX is computationally correct. The proof relies on the fact that correctness remains intact unless the adversary finds one of the hidden PRP values, and the probability of the latter can be bounded by the *one-way* security of the obfuscator.

The proof of Res-PRIV security of this construction involves an intricate game hopping argument, in order to deal with all possible correlations allowed by the Res-PRIV model (which are the same as those allowed by full PRIV). We outline it below, highlighting how various ingredients are used in the construction.

Theorem 3 **(DOX is Res-PRIV).** *If* FE *is an* IND-CPA*-secure functional encryption scheme,* PRP *is a* PRP*-secure pseudorandom permutation family and* Obf *is a* DI*-secure obfuscator then scheme* $\mathsf{DOX}[\mathsf{FE}, \mathsf{PRP}, \mathsf{Obf}]$ *is a* Res-PRIV*-secure keyword search scheme.*

Proof (Outline). The proof proceeds along six games as follows. Roughly speaking, after moving to an random permutation in Game$_1$ (and some bookkeeping in Game$_2$), in Game$_3$ we move from correlations between messages and keywords to their repetition patterns. In Game$_4$, we use obfuscation to deal with repetitions among keywords that do not match any of the messages (and were not queried to TGen in first phase). In Game$_5$, we use FE security to remove repetitions among messages that do not match any of the challenge keywords and were not queried to token-generation either (either due to legitimacy or adversarial restriction). Repetitions in all other cases can be dealt with using explicit values, the image matrix, or obfuscations. These steps make challenge ciphertexts independent of the challenge bit. In Game$_6$, using the security of the obfuscator we move to a setting where challenge tokens are also independent of the bit. In Game$_6$ advantage of any adversary is 0.

Game$_0$: This game is identical to the PRIV game.

Game$_1$: Instead of PRP, a truly random permutation is used in TGen. We simulate the random permutation via a lazily sampled table T. This transition is sound down to PRP security.

Game$_2$: We introduce a bad flag. We generate PRP values for all keywords and messages. If there are two T-values $(x_1, T(x_2))$ and $(x_2, T(x_2))$ such that $x_1 = T(x_2)$ we set bad. By the

OW security of the obfuscator, these PRP values remain hidden and bad can only be set with negligible probability.

Game$_3$: We compute the ciphertexts by encrypting $T(\mathbf{m}_b^*)$ instead of $\mathbf{m}_b^*$. This hop is reduced to the IND-CPA security of the FE, via explicit knowledge of challenge keywords and message by running the ppt sampler. Legitimacy will be violated if there is a w queried to TGen such that $\mathsf{w} = T(\mathsf{m}_b)$ or $\mathsf{m}_b = T(\mathsf{w})$. Both of these events set bad.

Game$_4$: Call a challenge keyword *unpaired* if it was not any of the challenge messages, and *new* if it is not queried to first-phase TGen. In this game, instead of T values we use *forgetful* random values for all new and unpaired keywords. We bound this hop using DI. We simulate first-phase TGen using a lazily sampled T and a msk. Next, we run the PRIV sampler *explicitly* and identify all new unpaired keywords. We define a DI sampler to sample consistent values on left and forgetful values on right (both independently of T), together with a second set consisting of sufficiently many consistent values on *both* sides. (The DI sampler does not need to respect any equality patterns.) This sampler can be shown to be statistically unpredictable. Once we receive the obfuscations, we use the first set, the explicit knowledge of challenge values and table T to form the challenge tokens and ciphertexts. For second-phase TGen queries we need to use consistent T values throughout. For values which match a first-phase query or a challenge messages we use T. If a query happens to *match a new unpaired keywords*—we can check this using the explicit knowledge of the keywords—we use a value from the second set of obfuscations. Otherwise we sample T values. We return 1 iff the adversary succeeds.

Game$_5$: Call a challenge message *unpaired* if it is not any of the challenge keywords, *LR-identical* if $\mathsf{m}_0^* = \mathsf{m}_1$, LR-differing if not equal, and *new* if it is unpaired and not queried to first-phase TGen. In this game instead of T values we use forgetful values for all unpaired LR-differing messages and all *new* LR-identical messages. We bound this hop down to IND-CPA. We will use the provided TGen oracle and only need to set T-values correctly. For first-phase TGen queries we lazily sample T. Next we run the sampler explicitly to obtain the challenges. For paired keywords or messages we use T-consistent values. For new unpaired keywords we use forgetful values (rule in Game$_4$). For unpaired messages, if LR-identical and queried to first-phase TGen (hence not new) we also use consistent T values. For LR-differing or new LR-identical messages we call LR in FE game, asking for T-consistent values on the left and independent forgetful values on the right. Note that LR-differing messages and new LR-identical messages are not queried to TGen at all due to our *restriction* on the adversary. If a second-phase TGen query matches a forgetful value generated in computing the LR query, we stop and guess that forgetful values were encrypted. (These values are information theoretically hidden if not encrypted.) Otherwise, we return 1 iff the adversary succeeds.

Game$_6$: In this game, irrespective of the bit, we use the second set of keywords for challenge token generation. We reduce this transition to the DI game. First-phase TGen queries are answered using a lazily sampled T and a generated msk. We set the DI sampler to run the PRIV sampler and on top of the output keywords, also ask for obfuscations of *messages* that are at the same time *LR-identical*, unpaired and new (it also outputs the random coins of first stage adversary, key generation and token extraction, along with a full image matrix as extra auxiliary information that will be needed for the second stage simulation). Using the symmetry of roles for keywords and messages in point functions, this sampler can be shown to be unpredictable whenever the PRIV sampler is. The obfuscations of messages will allow us to *check* if any of these messages (hidden under the obfuscation) match a first-phase TGen query. We need this as according to the rules of Game$_5$ we must use T-consistent values. For paired messages, which we can find using the image matrix, we also use T-consistent values. For unpaired keywords we use forgetful values. For all other unpaired messages (be it LR-differing or never queried to TGen) we use forgetful values (Game$_5$). Second phase TGen queries are answered using T-consistent values relying on the fact that we can use the obfuscations to check matches with paired keywords and the restriction that adversary cannot query a new unpaired LR-identical messages to TGen. We return 1 iff the adversary succeeds.

Challenge tokens in Game$_6$ are independent of the challenge bit. Due to the modifications in Game$_4$ and Game$_5$, the challenge ciphertexts are also independent of it. To see this note that ciphertexts contain on left and right: (1) identical T-consistent values that follow the correct repetition pattern for paired massages; (2) forgetful (independent) values for LR-differing messages; (3) identical T-consistent values that follow the correct repetition pattern for LR-identical messages queried in the first stage; (4) forgetful (independent) values for LR-identical messages not queried in the first stage. The adversary, therefore, has zero advantage in this game. □

6.4 The Verifiably-Obfuscate-Encrypt-Extract (VOEX) transform

We now present a fourth construction for point functions, which although simpler, conceptually relies on the observation that messages can be encoded as circuits that other circuits can evaluate. The obfuscator that we will rely on in our construction needs to be *verifiable*, meaning that there is an efficient algorithm to determine if a circuit $\overline{\mathsf{C}}$ is an obfuscation of a point function $\mathsf{C}[\mathsf{m}]$ for a message $\mathsf{m} \in \mathsf{MSp}_\lambda$. This property can be easily added by attaching a NIZK proof that there exist (m, r) such that $\overline{\mathsf{C}} = \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{m}]; \mathsf{r})$.

The VOEX transform. Let $\mathsf{NIZK} = (\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ be a non-interactive zero-knowledge proof system (see Section 2.4). Let Obf be an obfuscator supporting a circuit family $\mathsf{CSp} := \{\mathsf{CSp}^1_\lambda \cup \mathsf{CSp}^2_\lambda\}_{\lambda \in \mathbb{N}}$, where $\mathsf{CSp}^1_\lambda := \{\mathsf{C}[\mathsf{m}] : \mathsf{m} \in \mathsf{MSp}_\lambda\}$ and $\mathsf{CSp}^2_\lambda := \{\mathsf{D}[\mathsf{crs}, \mathsf{m}] : \mathsf{m} \in \mathsf{MSp}_\lambda, \mathsf{crs} \in [\mathsf{NIZK.Setup}(1^\lambda)]\}$ with

$$\mathsf{D}[\mathsf{crs}, \mathsf{w}](\overline{\mathsf{C}}, \pi) := \begin{cases} 1 & \text{if } \mathsf{NIZK.Verify}(\mathsf{crs}, \overline{\mathsf{C}}, \pi) \wedge \overline{\mathsf{C}}(\mathsf{w}) = 1 \ ; \\ 0 & \text{otherwise.} \end{cases}$$

Let $\{\mathsf{RSp}_\lambda\}_{\lambda \in \mathbb{N}}$ denote the randomness space of Obf. Let FE be a functional encryption scheme supporting general circuits. We construct a keyword search scheme $\mathsf{KS} := \mathsf{VOEX}[\mathsf{FE}, \mathsf{NIZK}, \mathsf{Obf}]$ via the *Verifiably-Obfuscate-Encrypt-Extract* (VOEX) transform for keyword space $\mathsf{WSp} = \mathsf{MSp}$ as follows.

Setup: Algorithm $\mathsf{KS.Gen}(1^\lambda)$ generates a key pair $(\mathsf{msk}, \mathsf{mpk}) \leftarrow_{\$} \mathsf{FE.Gen}(1^\lambda)$ and a common reference string $\mathsf{crs} \leftarrow_{\$} \mathsf{NIZK.Setup}(1^\lambda)$. It returns the key pair $((\mathsf{msk}, \mathsf{crs}), (\mathsf{mpk}, \mathsf{crs}))$.

Encryption: Algorithm $\mathsf{KS.Enc}((\mathsf{mpk}, \mathsf{crs}), \mathsf{m})$ generates $\overline{\mathsf{C}} \leftarrow_{\$} \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{m}]; \mathsf{r})$ for $\mathsf{r} \leftarrow_{\$} \mathsf{RSp}$. It sets $\pi \leftarrow_{\$} \mathsf{NIZK.Prove}(\mathsf{crs}, \overline{\mathsf{C}}, (\mathsf{m}, \mathsf{r}))$ and finally returns $\mathsf{FE.Enc}(\mathsf{mpk}, (\overline{\mathsf{C}}, \pi))$.

Token generation: Algorithm $\mathsf{KS.TGen}((\mathsf{msk}, \mathsf{crs}), \mathsf{w})$ generates a token for circuit $\mathsf{D}[\mathsf{crs}, \mathsf{w}]$ using the token-extraction algorithm $\mathsf{FE.TGen}$ and returns the result.

Evaluation: Algorithm $\mathsf{KS.Test}(\mathsf{c}, \mathsf{tk})$ simply runs $\mathsf{FE.Eval}(\mathsf{c}, \mathsf{tk})$.

Correctness of the construction follows from the correctness of the obfuscator and that of the functional encryption scheme, as well as the completeness of the proof system. Before presenting the theorem, we clarify the requirements on the underlying obfuscation scheme.

PRIV-restricted samplers. As shown in the work of Barak et al. [9], no 2-circuits general-purpose VBB obfuscator exists. This impossibility result can be extended to rule out general-purpose DI obfuscation as well and, in particular, DI obfuscation supporting the class of circuits we require for instantiating our construction above. Briefly, consider the circuits $\mathsf{D}[\mathsf{w}](\mathsf{C}) := \mathsf{C}(\mathsf{w})$ and a sampler $\mathcal{S}$ that outputs circuits $(\mathsf{D}[\mathsf{w}], \mathsf{C}[\mathsf{w}])$ on the left and $(\mathsf{D}[\mathsf{w}], \mathsf{C}[\overline{\mathsf{w}}])$ on the right for a uniform keyword w. This sampler can be shown to be unpredictable. However, the DI game can be won by evaluating (an obfuscation of) the first challenge circuit on an obfuscation of the second challenge circuit.

For our particular construction, however, we rely on a weaker form of obfuscation that is only required to support samplers that output circuits and messages that are restricted by the PRIV legitimacy condition (this is a result of our reduction strategy). Concretely, such circuits and messages will result on image matrices that are identical on the left and right, which completely rules out attacks akin to those in [9]. We call this class of DI samplers PRIV*-restricted*. Formally, a DI sampler $\mathcal{S}$ is PRIV-restricted for circuit class CSp if for a legitimate PRIV sampler $\mathcal{S}'$ and a non-interactive zero-knowledge proof system NIZK it operates as follows.

$$\begin{aligned} \mathcal{S}(1^\lambda, st, \mathsf{crs}) : {} & (\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow_{\$} \mathcal{S}'(1^\lambda, st);\\ & \text{if } \mathsf{crs} \notin [\mathsf{NIZK.Setup}(1^\lambda)] \text{ return } ([], [], \epsilon)\\ & \text{else return } ((\mathsf{D}[\mathsf{crs}, \mathbf{C}_0], \mathsf{C}[\mathbf{m}_0]), (\mathsf{D}[\mathsf{crs}, \mathbf{C}_1], \mathsf{C}[\mathbf{m}_1]), (z, \mathbf{C}_0(\mathbf{m}_0))) \end{aligned}$$

The PRIV security of the VOEX construction is established in the following theorem.

Theorem 4 (VOEX is PRIV secure). *If* FE *is* IND-CPA *secure,* NIZK *is perfectly sound and computationally zero-knowledge, and obfuscator* Obf *is* DI *secure with respect to* PRIV*-restricted samplers, then scheme* $\mathsf{VOEX}[\mathsf{FE}, \mathsf{Obf}, \mathsf{NIZK}]$ *is* PRIV *secure.*

Proof (Outline). The proof follows a sequence of games as follows.

Game$_0$: This is the PRIV game for the VOEX construction.

Game$_1$: We say a message $\mathbf{m}_b[i]$ is *unpaired* if $\mathbf{m}_b[i] \notin \mathbf{w}_b$. Note that for all legitimate samplers if $\mathbf{m}_0[i]$ is unpaired, so is $\mathbf{m}_1[i]$. In this game, the LR oracle replaces all unpaired messages (on both sides) which are LR-differing (that is, when $\mathbf{m}_0[i] \neq \mathbf{m}_0[i]$) with random and independently sampled values. The distance to the previous game can be upper bounded using the IND-CPA security of the FE scheme. The legitimacy of the algorithm playing the IND-CPA game in the reduction is guaranteed because: (1) replaced messages are LR-differing and therefore the adversary cannot ask tokens for those in the PRIV game (and hence also the IND-CPA hame); (2) replacements are random and information-theoretically hidden from the adversary when the original messages are encrypted, and if the adversary asks for token for one of the random replacements, it can be only because the ciphertexts are leaking one of these replacements.

Game$_2$: In this game we use Sim to generate simulated proofs in the LR oracle without using the explicit knowledge of the messages. The distance to the previous game can be bounded by the zero-knowledge property of the NIZK proof system.

Game$_3$: In this game, regardless of bit b, we use the second set of keywords and messages to generate the challenge. We reduce this transition to the DI game. We set the DI sampler to take a crs along with the state st required to run the PRIV sampler; it runs the PRIV sampler to obtain keywords and messages, and it outputs a D circuit (with a hardwired crs) for every keyword and a C circuit for every message (after carefully replacing LR-differing unpaired messages with random values as in Game_1). This DI sampler is by definition PRIV-restricted and it is unpredictable whenever the underlying PRIV sampler is unpredictable. The proof of this fact relies on the perfect soundness of the proof system under the binding crs, whose validity we assume can be efficiently checked [34]. The PRIV predictor uses its oracle to answer the DI predictor's queries (if a query contains an obfuscated point circuit and the attached proof verifies, the unbounded PRIV predictor can reverse-engineer the obfuscated circuit to recover its underlying point, and query its own oracle on it).

The adversary against the DI game simulates the environment of Game_2 as follows. It generates a key pair $(\mathsf{msk}, \mathsf{mpk})$ and simulated $(\mathsf{crs}, \mathsf{tp})$ for the NIZK, and then it runs the first stage of the PRIV adversary until it obtains state st for the LR oracle call. It then calls its own LR oracle on (st, crs), obtaining a set of obfuscations. As before, the trapdoor tp is used to produce simulated proofs of the obfuscations of C circuits corresponding to messages, resulting in well-formed challenge ciphertexts. It then runs the second stage of the PRIV adversary, answering its token extraction queries using the master secret key and, when this adversary returns a bit b', it uses it as its own guess.

In Game_3, the challenge is independent of bit b and therefore the adversary has zero advantage. □

7 Concluding Remarks

The main open problem we leave for future work is to construct functional encryption schemes that achieve full PRIV security under milder assumptions or are more efficient under restricted versions of the PRIV model. For general circuits, a possible path towards a solution to this open problem would be to consider the FE construction of Garg et al. [26]. There, a token for a circuit C is (roughly speaking) an indistinguishability obfuscation of the circuit $\mathsf{C}(\mathsf{PKDec}(\mathsf{sk}, \cdot))$ for a PKE decryption circuit PKDec. A natural question is whether this construction already achieves some form of privacy under the conjecture that the indistinguishability obfuscator achieves VGB obfuscation [15, Section 1.1]. For specific classes, one can follow the various constructions presented here and explore variations and optimizations of their underlying primitives. Indeed, since Res-PRIV constitutes a very mild weakening of PRIV, it could be that a modification of it allows the proof of security to be extended to the PRIV model. Finally, we note that our work leaves open the task of formalizing and realizing privacy notions for more expressive cryptographic primitives such as multi-input or randomized FE schemes.

Acknowledgements. A. Arriaga is supported by the National Research Fund, Luxembourg (AFR Grant No. 5107187).

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *J. Cryptology*, 21(3):350–391, 2008.
2. S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. On the practical security of inner product functional encryption. In *PKC 2015*, vol. 9020 of LNCS, pp. 777–798. Springer, 2015. IACR Cryptology ePrint Archive, Report 2013/744, 2013.
3. S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In *CRYPTO 2013*, vol. 8043 of LNCS, pp. 500–518. Springer, 2013.

4. P. Ananth, D. Boneh, S. Garg, A. Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. IACR Cryptology ePrint Archive, Report 2013/689, 2013.
5. P. Ananth, Z. Brakerski, G. Segev, and V. Vaikuntanathan. From selective to adaptive security in functional encryption. IACR Cryptology ePrint Archive, Report 2014/917, 2014.
6. A. Arriaga, Q. Tang, and P. Ryan. Trapdoor privacy in asymmetric searchable encryption schemes. In *AFRICACRYPT 2014*, vol. 8469 of LNCS, pp. 31–50. Springer, 2014.
7. B. Barak, N. Bitansky, R. Canetti, Y. T. Kalai, O. Paneth, and A. Sahai. Obfuscation for evasive functions. In *TCC 2014*, vol. 8349 of LNCS, pp. 26–51. Springer, 2014.
8. B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai. Protecting obfuscation against algebraic attacks. In *EUROCRYPT 2014*, vol. 8441 of LNCS, pp. 221–238. Springer, 2014.
9. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO 2001*, vol. 2139 of LNCS, pp. 1–18. Springer, 2001.
10. M. Barbosa and P. Farshim. On the semantic security of functional encryption schemes. In *PKC 2013*, vol. 7778 of LNCS, pp. 143–161. Springer, 2013.
11. M. Bellare, R. Dowsley, and S. Keelveedhi. How secure is deterministic encryption? In *PKC 2015*, vol. 9020 of LNCS, pp. 52–73. Springer, 2015.
12. M. Bellare, I. Stepanovs, and S. Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In *ASIACRYPT 2014*, vol. 8874 of LNCS, pp. 102–121. Springer, 2014.
13. M. Bellare, I. Stepanovs and S. Tessaro. Contention in cryptoland: obfuscation, leakage and UCE. IACR Cryptology ePrint Archive, Report 2015/487, 2015.
14. N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. *J. Cryptology*, 27(2):317–357, 2014.
15. N. Bitansky, R. Canetti, Y. T. Kalai, and O. Paneth. On virtual grey box obfuscation for general circuits. In *CRYPTO 2014*, vol. 8617 of LNCS, pp. 108–125. Springer, 2014.
16. Z. Brakerski and G. N. Rothblum. Black-box obfuscation for d-CNFs. IACR Cryptology ePrint Archive, Report 2013/557, 2013.
17. Z. Brakerski and G. N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. IACR Cryptology ePrint Archive, Report 2013/563, 2013.
18. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search In *EUROCRYPT 2004*, vol. 3027 of LNCS, pp. 506-522, Springer, 2004.
19. D. Boneh, A. Raghunathan, and G. Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In *CRYPTO 2013*, vol. 8043 of LNCS, pp. 461–478. Springer, 2013.
20. D. Boneh, A. Raghunathan, and G. Segev. Function-private subspace-membership encryption and its applications. In *ASIACRYPT 2013*, vol. 8269 of LNCS, pp. 255–275. Springer, 2013.
21. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *TCC 2011*, vol. 6597 of LNCS, pp. 253–27. Springer, 2011.
22. C. Brzuska and A. Mittelbach. Indistinguishability obfuscation versus multi-bit point obfuscation with auxiliary input. In *ASIACRYPT 2014*, vol. 8874 of LNCS, pp. 142–161. Springer, 2014.
23. R. Canetti, G. N. Rothblum and M. Varia. Obfuscation of hyperplane membership. In *TCC 2010*, vol. 5978 of LNCS, pp. 72–89. Springer, 2010.
24. R. Canetti, V. Vaikuntanathan. Obfuscating branching programs using black-box pseudo-free groups. IACR Cryptology ePrint Archive, Report 2013/500, 2013.
25. A. De Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth and G. Persiano. On the achievability of simulation-based security for functional encryption. In *CRYPTO 2013*, vol. 8043 of LNCS, pp. 519–535. Springer, 2013.
26. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS 2013*, pp. 40–49. IEEE Computer Society, 2013.
27. O. Goldreich. The foundations of cryptography, vol. 2, Basic Applications. Cambridge University Press, 2004.
28. S. Goldwasser and Y. T. Kalai. On the impossibility of obfuscation with auxiliary input. In *FOCS 2005*, pp. 553–562. IEEE Computer Society, 2005.
29. S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC 2013*, pp. 555–564. ACM, 2013.
30. S. Goldwasser and G. N. Rothblum. On best-possible obfuscation. *J. Cryptology*, 27(3):480–505, 2014.
31. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO 2012*, vol. 7417 of LNCS, pp. 162–179. Springer, 2012.
32. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. IACR Cryptology ePrint Archive, Report 2015/029, 2015.

33. V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In *TCC 2011*, vol. 6597 of LNCS, pp. 182–200. Springer, 2011.
34. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, vol. 4965 of LNCS, pp. 415432. Springer, 2008.
35. Y. Ishai, O. Pandey, and A. Sahai. Public-coin differing-inputs obfuscation and its applications. In *TCC 2015*, vol. 9015 of LNCS, pp. 668–697. Springer, 2015.
36. A. O'Neill. Definitional issues in functional encryption. IACR Cryptology ePrint Archive, Report 2010/556, 2010.
37. H. Wee. Zero knowledge in the random oracle model, revisited. In *ASIACRYPT 2009*, vol. 5912 of LNCS, pp. 417–434. Springer, 2009.

A Taxonomy of Samplers

We define a number of special classes of samplers by imposing structural restrictions on their internal operation.

Stateless. The sampler does not keep any internal state and uses independent set of coins on each invocation. All samplers will be stateless in this work unless stated otherwise.

(t, s)-bounded. For polynomials t and s, with overwhelming probability $|\mathbf{C}_0| = |\mathbf{C}_1| \leq t(\lambda)$ and $|\mathbf{m}_0| = |\mathbf{m}_1| \leq s(\lambda)$.

Circuits-only. The sampler outputs no messages with overwhelming probability, i.e. it is $(\cdot, 0)$-bounded.

One-sided. $\mathbf{C}_0 = \mathbf{C}_1$ and $\mathbf{m}_0 = \mathbf{m}_1$ with overwhelming probability. In this case we will simply write $(\mathbf{C}, \mathbf{m}, z) \leftarrow_\$ \mathcal{S}(1^\lambda, st)$ for the sampling operation. Note that every one-sided sampler is trivially unpredictable.

Input-independent. For any 1^λ and st, $\mathcal{S}(1^\lambda, st) = \mathcal{S}(1^\lambda, \varepsilon)$ with overwhelming probability.

Aux-free. With overwhelming probability $z = \varepsilon$.

Simple. If the sampler is both aux-free and input-independent.

Random-aux. For a polynomial poly and a ppt algorithm $\mathcal{S}'$ the sampler takes the form

$$\begin{aligned} \mathcal{S}(1^\lambda, st) : \; & z \leftarrow_\$ \{0,1\}^{\mathsf{poly}(\lambda)}; \\ & (\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1) \leftarrow_\$ \mathcal{S}'(1^\lambda, z, st); \\ & \text{return } (\mathbf{C}_0, \mathbf{C}_1, \mathbf{m}_0, \mathbf{m}_1, z) \; . \end{aligned}$$

Differing-inputs. With overwhelming probability z contains the sampler's output circuits $(\mathbf{C}_0, \mathbf{C}_1)$. Note that statistical unpredictability would imply that the sampled circuits are *functionally equivalent*, whereas computational unpredictability would lead to a notion of differing-inputs samplers used to formulate differing-inputs obfuscation [4,12].

Block-source. A t-block-source is a random variable $X = (X_1, \ldots, X_t)$ where for every $j \in [t]$ and $x_1, \ldots, x_{j-1}$ it holds that $X_j|_{X_1=x_1,\ldots,X_{j-1}=x_{j-1}}$ has high min-entropy. There is therefore *sufficient* decorrelation between different components in such a distribution. We can model block sources in our framework by restricting attention to ppt samplers that take the form

$$\begin{aligned} \mathcal{S}(1^\lambda, st) : \; & (\mathbf{C}_0, \mathbf{C}_1) \leftarrow_\$ \mathcal{S}'(1^\lambda, st); \\ & j \leftarrow_\$ [t]; \\ & \text{return } ((\mathbf{C}_0[j], \mathbf{C}_1[j]), (\mathbf{C}_0[1..(j-1)], \mathbf{C}_1[1..(j-1)])) \end{aligned}$$

where $\mathcal{S}'$ is a $(t, 0)$-bounded sampler. The rationale here is that any indistinguishability-based security definition that imposes an adversary to output two block sources, and later on distinguish some computation performed on the sampled values, e.g. [19], would remain the same if a sampler such as the one above was used instead (note that in this case, the adversary can only have an advantage when outputting distributions that component-wise differ with non-negligible probability).

B Proof of Proposition 2: CVGB $\implies$ DI

Proof. Let $(\mathcal{S}, \mathcal{A})$ be a DI adversary against the obfuscator. We show that the advantage of $\mathcal{A}$ must be negligible if $\mathcal{S}$ is unpredictable and the obfuscator is CVGB secure. Also, let RSp_λ denote the randomness space of $\mathcal{A}$. Consider a one-sided circuit sampler $\mathcal{S}'$ that selects $\mathsf{r} \leftarrow_\$ \mathsf{RSp}_\lambda$, runs $\mathcal{A}(1^\lambda; \mathsf{r})$ until it outputs st, runs $\mathcal{S}(1^\lambda, st)$, chooses a bit b uniformly at random, and outputs the left or right outputs of $\mathcal{S}$ according to the bit b, along with auxiliary information z and coins r. Let $\mathcal{B}$ be a CVGB-Real adversary that runs $\mathcal{A}$ on the same coins and answers SAM oracle query with its challenge vector of obuscations. $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. By the CVGB property, for $(\mathcal{S}', \mathcal{B})$ there is a (possibly unbounded) simulator Sim such that:

$$\mathbf{Adv}^{\mathrm{cvgb}}_{\mathsf{Obf},\mathcal{S}',\mathcal{B},\mathsf{Sim}}(\lambda) = \left|\Pr\left[\text{CVGB-Real}^{\mathcal{S}',\mathcal{B}}_{\mathsf{Obf}}(\lambda)\right] - \Pr\left[\text{CVGB-Ideal}^{\mathcal{S}',\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)\right]\right| \ .$$

Note that

$$\mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S},\mathcal{A}}(\lambda) = \Pr\left[\text{CVGB-Real}^{\mathcal{S}',\mathcal{B}}_{\mathsf{Obf}}(\lambda)|b=1\right] - \Pr\left[\text{CVGB-Real}^{\mathcal{S}',\mathcal{B}}_{\mathsf{Obf}}(\lambda)|b=0\right] \ .$$

Hence,

$$\mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S},\mathcal{A}}(\lambda) \leq \left|\Pr\left[\text{CVGB-Ideal}^{\mathcal{S}',\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)|b=1\right] - \Pr\left[\text{CVGB-Ideal}^{\mathcal{S}',\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)|b=0\right]\right| + \\ + 2 \cdot \mathbf{Adv}^{\mathrm{cvgb}}_{\mathsf{Obf},\mathcal{S}',\mathcal{B},\mathsf{Sim}}(\lambda) \ .$$

Let $Q(\lambda)$ denote the number of queries of Sim. We claim that there is a predictor $\mathcal{P}$ making at most $Q(\lambda)$ queries such that

$$\left|\Pr\left[\text{CVGB-Ideal}^{\mathcal{S}',\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)|b=1\right] - \Pr\left[\text{CVGB-Ideal}^{\mathcal{S}',\mathsf{Sim}}_{\mathsf{Obf}}(\lambda)|b=0\right]\right| \leq Q(\lambda) \cdot \mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda) \ .$$

From this it follows that

$$\mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S},\mathcal{A}}(\lambda) \leq Q(\lambda) \cdot \mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathrm{cvgb}}_{\mathsf{Obf},\mathcal{S}',\mathcal{B},\mathsf{Sim}}(\lambda) \ .$$

We prove the claim via unpredictability of the sampler. Observe that the views of Sim in the CVGB-Ideal game for $b = 0$ and $b = 1$ are identical unless Sim queries its oracle on a point that results in different outputs for the left and right circuits. This event, however, immediately leads to a break of unpredictability. Consider a (possibly unbounded) predictor $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2)$ as follows. $\mathcal{P}_1$ selects random coins $\mathsf{r} \leftarrow_\$ \mathsf{RSp}_\lambda$ and runs $\mathcal{A}(1^\lambda; \mathsf{r})$ until it outputs st. $\mathcal{P}_1$ then outputs (st, r). $\mathcal{P}_2(1^\lambda, \epsilon, z, \mathsf{r})$ chooses a random index $i \leftarrow_\$ [Q(\lambda)]$ indicating a guess for the first query of Sim that leads to a break of unpredictability. It runs $\mathsf{Sim}(z||\mathsf{r})$ and answers its oracle queries using its own provided oracle (which always respond for left circuits $b = 0$). At query i algorithm $\mathcal{P}_2$ stops and outputs the queried value. With probability $1/Q(\lambda)$ this is the first query that the bad event occurs. Hence $\mathcal{P}_2$ runs Sim perfectly until query i, at which point it wins the unpredictability game.

This concludes the proof as the above holds for any poly (which in turn implies that the left hand side is negligible). □

C Proof of Proposition 3: DI $\implies$ OW

Proof. We show that OW is a weakening of DI for point circuits. Given an OW adversary $\mathcal{A}$ we construct a sampler $\mathcal{S}$ and a distinguisher $\mathcal{D}$ attacking DI security as follows. First, we partition each set CSp_λ into two sets (of super-polynomial size) CSp_λ^0 and CSp_λ^1, such that $|\mathsf{CSp}_\lambda^0| = |\mathsf{CSp}_\lambda^1| + \mathsf{negl}(\lambda)$. This partition can be based on some lexicographic criterion (e.g., the most significant bit of the point), as long as one can efficiently decide membership in each partition. Our sampler $\mathcal{S}$ samples two point circuits C_0 and C_1, uniformly at random from CSp_λ^0 and CSp_λ^1, respectively. It then outputs two t-sized vectors $\mathbf{C}_0 = (\mathsf{C}_0, \dots, \mathsf{C}_0)$ and $\mathbf{C}_1 = (\mathsf{C}_1, \dots, \mathsf{C}_1)$. (Here t is the length parameter initially output by the one-wayness adversary $\mathcal{A}$.) (Recall that auxiliary information z is empty.) It is clear that $\mathcal{S}$ is unpredictable, and therefore legitimate as a DI sampler. On obtaining the obfuscations, the distinguisher $\mathcal{D}$ runs adversary $\mathcal{A}$ on the same inputs and recover a circuit C'. Observe that the distribution of the obfuscations provided to $\mathcal{A}$ is statistically close to the correct distribution given the *combined* action of $\mathcal{S}$ and the challenge bit in the DI game. Distinguisher $\mathcal{D}$ then returns 0 if $\mathsf{C}' \in \mathsf{CSp}_\lambda^0$ and 1 otherwise. It is straightforward to establish that a non-negligible advantage for $\mathcal{A}$ in the OW game translates to a non-negligible advantage for $(\mathcal{S}, \mathcal{D})$ in the DI game. □

D Proof of Proposition 4: PRIV-TO $\implies$ 1-DI

Proof. We first describe the operation of the required obfuscator. Given a circuit C, the required obfuscator Obf generates an FE key pair $(\mathsf{mpk}, \mathsf{msk})$ and uses the master secret key to extract a token tk for C. It then defines the obfuscated circuit to be one that first encrypts m under mpk using trivial random coins, and then evaluates the resulting ciphertext using tk, i.e., the circuit

$$\overline{\mathsf{C}}[\mathsf{mpk}, \mathsf{tk}](\cdot) := \mathsf{FE.Eval}(\mathsf{FE.Enc}(\cdot, \mathsf{mpk}; 0^{\mathsf{poly}(\lambda)}), \mathsf{tk}) \ .$$

The correctness of this obfuscator follows from that of the FE scheme. The proof of DI security for this construction with respect to samplers that output a single circuit pair is a direct reduction to PRIV-TO. We construct a PRIV-TO adversary that uses the DI sampler without change and does nothing in the first stage. In the second stage, on obtaining the challenge token tk, constructs $\overline{\mathsf{C}}[\mathsf{mpk}, \mathsf{tk}](\cdot)$ and passes it on to the DI distinguisher. It will return whatever the distinguisher outputs. This simulation is easily seen to be perfect. □

Remark. For arbitrary DI samplers the argument above fails. This is due to the fact that communication between the sampler and the distinguisher is restricted (by the unpredictability condition) and hence hybrid arguments cannot be made to go through.

E PRIV-TO and IND-CPA Security of OX Transform

Proof. The proof is straightforward and results from direct reductions to the underlying components used in the construction. We start by proving that $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$ is PRIV-TO-secure for a circuits family CSp and (circuits-only) sampler class $\mathbb{S}$ if Obf is DI-secure for CSp and $\mathbb{S}$. Given an adversary $(\mathcal{S}, \mathcal{A}_1)$ against PRIV-TO security of $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$, we construct an adversary $(\mathcal{S}', \mathcal{B}_1)$ against the DI security of Obf as follows. We set $\mathcal{S}'$ to be the same as $\mathcal{S}$. Algorithm $\mathcal{B}_1$ runs $\mathsf{FE.Gen}(1^\lambda)$ to generate on its own a master secret key and master public key pair $(\mathsf{msk}, \mathsf{mpk})$. Then, $\mathcal{B}_1$ runs $\mathcal{A}_1$

on mpk, answering all its token-generation queries by running $\mathsf{FE.TGen}(\mathsf{msk}, \cdot)$, until $\mathcal{A}_1$ calls LR on some state *st*. At this point, $\mathcal{B}_1$ calls its own LR oracle on *st* and receives as a challenge a vector of obfuscated circuits. $\mathcal{B}_1$ generates a token for each circuit, and forwards the result to $\mathcal{A}_1$. Thereafter, $\mathcal{B}_1$ continues running $\mathcal{A}_1$, answering its second-stage token-generation queries as before until $\mathcal{A}_1$ outputs a bit b', which $\mathcal{B}_1$ outputs as its own guess. The simulation is perfect and $\mathcal{S}'$ is unpredictable because $\mathcal{S}$ is unpredictable.

We now prove the second part of the theorem, i.e. that $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$ is IND-CPA-secure if FE is. Let $\mathcal{A}_2$ be an adversary against IND-CPA security of $\mathsf{OX}[\mathsf{FE}, \mathsf{Obf}]$. We construct an adversary $\mathcal{B}_2$ against IND-CPA security of FE. Algorithm $\mathcal{B}_2(\mathsf{mpk})$ runs $\mathcal{A}_2(\mathsf{mpk})$, answering its first-stage $\textsc{TGen}(\mathsf{C})$ queries by first computing an obfuscation $\overline{\mathsf{C}}$ of circuit C, placing a token-generation query on $\overline{\mathsf{C}}$ to its own $\textsc{TGen}$ oracle, and forwarding the token to $\mathcal{A}_2$. When $\mathcal{A}_2$ asks to be challenged on messages $(\mathsf{m}_0, \mathsf{m}_1)$, $\mathcal{B}_2$ calls its own LR oracle on these messages and forwards the challenge ciphertext to $\mathcal{A}_2$. Second-stage $\textsc{TGen}$ queries are answered as before. Finally, $\mathcal{B}_2$ outputs $\mathcal{A}_2$'s guess b' as its own guess. Here again, the simulation is perfect and legitimacy of $\mathcal{B}_2$ follows from the legitimacy of $\mathcal{A}_2$ and the fact that the obfuscator preserves the functionality of the circuit, which means that $\mathcal{B}_2$ has precisely the same advantage as $\mathcal{A}_2$. Therefore, we conclude that

$$\mathbf{Adv}^{\text{priv-to}}_{\mathsf{OX}[\mathsf{FE},\mathsf{Obf}],\mathcal{S},\mathcal{A}_1}(\lambda) = \mathbf{Adv}^{\text{di}}_{\mathsf{Obf},\mathcal{S},\mathcal{B}_1}(\lambda), \text{ and that}$$
$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{OX}[\mathsf{FE},\mathsf{Obf}],\mathcal{A}_2}(\lambda) = \mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_2}(\lambda) \; .$$

□

F Analysis of DOX Transform

F.1 Computational correctness

Theorem (DOX is computationally correct). Let Obf be an obfuscator and let FE be a computationally correct FE scheme. Then $\mathsf{DOX}[\mathsf{FE}, \mathsf{PRP}, \mathsf{Obf}]$ is computationally correct if the underlying PRP is pseudorandom and Obf is OW secure. More precisely, for any adversary $\mathcal{A}$ in game CC against $\mathsf{DOX}[\mathsf{FE}, \mathsf{PRP}, \mathsf{Obf}]$, placing at most t queries to $\textsc{TGen}$, there exist adversaries $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ such that

$$\mathbf{Adv}^{\text{cc}}_{\mathsf{KS},\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\text{cc}}_{\mathsf{FE},\mathcal{B}_3}(\lambda) + (t+1) \cdot \mathbf{Adv}^{\text{ow}}_{\mathsf{Obf},\mathcal{B}_2}(\lambda) + \mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{B}_1}(\lambda) + \frac{t+1}{|\mathsf{WSp}_\lambda|} \; .$$

Proof. The proof is simple and follows two game hops as follows.

Game$_0$: This is the CC game with respect to FE and PRP.
Game$_1$: In this game instead of a PRP a truly random permutation (simulated via lazy sampling) is used in the calculation of tokens in $\textsc{TGen}$ oracle and preparing tp.
Game$_2$: In this game, if a token generation query or one of $\mathcal{A}$'s output words matches any of the randomly generated words (via lazy sampling) the game aborts.

The analyses of the game transitions are straightforward. The transition from Game$_0$ to Game$_1$ relies on the security of the PRP. The transition from Game$_1$ to Game$_2$ is down to the one-way security of the obfuscator (note that the only information leaked to the adversary about each of the random keywords is via an obfuscated circuit included in the extracted tokens). Finally, the advantage of the adversary in Game_2 can be bounded down to the correctness of FE. We give the details next.

Game_0 TO Game_1. Any adversary $\mathcal{A}$ with visible advantage difference in these two games can be converted to an adversary $\mathcal{B}_1$ against the security of the PRP. Assume that lazy sampling is implemented using a table T, i.e., $T[\mathsf{w}]$ indicates the random value assigned to w. Algorithm $\mathcal{B}_1$ starts by generating an FE key pair. It handles a queries w of $\mathcal{A}$ to TGEN by first computing $\mathsf{w}' \leftarrow \mathrm{FN}(\mathsf{w})$ via its FN oracle, obfuscating the circuits associated with these keywords, and finally generating a token for the disjunction of the obfuscated circuits using the master secret key. Token generation after $\mathcal{A}$ terminates is handled similarly, and the remaining operations in of the CC game can be simulated using mpk. $\mathcal{B}_1$ will finally check if $\mathcal{A}$ succeeded in breaking correctness. If so, then its output will be 0. Else, it will be 1.

Note that when the FN oracle implements the PRP, Game_0 is simulated for $\mathcal{A}$, and when it implements a random permutation Game_1 is simulated. A simple probability analysis yields,

$$\Pr[\mathrm{Game}_0(1^\lambda)] - \Pr[\mathrm{Game}_1(1^\lambda)] = \mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{B}_1}(\lambda) \ .$$

Game_1 TO Game_2. Let us consider that the game is aborted if, at the end of the execution of Game_2, one considers all keywords explicitly output by the adversary (i.e., all w^* in the list of keywords queried from TGEN plus the challenge keyword and message output by the adversary when it terminates), and for some keyword w in table T we have:

$$\mathsf{w}^* = T[\mathsf{w}] \ .$$

We bound the probability that this bad flag is set via the one-way security of the obfuscation. We build the required $\mathcal{B}_2$ against the $(t+1)$-OW security of Obf as follows. $\mathcal{B}_2$ first guesses the query i in which $\mathcal{A}$ first produces w by choosing an index $i \leftarrow_{\$} [t+1]$, where t is an upper bound on the number of TGEN queries that $\mathcal{A}$ makes and the extra 1 accounts for the challenge keyword it produces on termination. $\mathcal{B}_2$ then generates an FE key pair, runs $\mathcal{A}$ and answers its TGEN queries using the master secret key and constructing $T[\mathsf{w}]$ as before, except when the i-th query comes (and all future w queries). In the latter case, $\mathcal{B}_2$ uses a new challenge obfuscated circuit it receives in the one-wayness game. Note that we have implicitly programmed $T[\mathsf{w}]$ to be an unknown value, which leads to an inconsistency with probability at most $(t+1)/|\mathsf{WSp}|$: an upper bound on the probability that this value collides with one of the values in T during the entire game. When the bad event is detected, and if $\mathcal{B}_2$'s guess was correct, then $\mathcal{B}_2$ can recognize the faulty keyword by checking the obfuscated circuits it received for a match, and it can win the one-wayness game. Hence,

$$\Pr[\mathrm{Game}_1(1^\lambda)] - \Pr[\mathrm{Game}_2(1^\lambda)] \leq (t+1) \cdot \mathbf{Adv}^{\mathrm{ow}}_{\mathsf{Obf},t,\mathcal{B}_2}(\lambda) + \frac{t+1}{|\mathsf{WSp}_\lambda|} \ .$$

ANALYSIS OF Game_2. In this game we use $\mathcal{A}$ to build an adversary against the computational correctness of the underlying FE scheme. Note that if Game_2 does not abort, then $\mathsf{m} \neq T(\mathsf{w})$ when $\mathcal{A}$ terminates. We show that if $\mathcal{A}$ wins without any aborts we can build an adversary $\mathcal{B}_3$ which wins the FE correctness game. Algorithm $\mathcal{B}_3$ gets mpk and runs $\mathcal{A}$ on it. It answers $\mathcal{A}$'s TGEN queries using its own oracle, still lazily sampling $T[\mathsf{w}]$ and asking for a trapdoor on the disjunction of the obfuscated circuits. When $\mathcal{A}$ returns (m, w), algorithm $\mathcal{B}_3$ also returns these. Note that this is winning pair iff it is a winning pair in the FE game. We therefore have

$$\Pr[\mathrm{Game}_2(1^\lambda)] = \mathbf{Adv}^{\mathsf{cc}}_{\mathsf{FE},\mathcal{B}_3}(\lambda) \ .$$

□

F.2 Res-PRIV security

Theorem (Res-PRIV **security of DOX).** If FE is an IND-CPA secure functional encryption scheme, PRP is pseudorandom and Obf is a DI-secure obfuscator then scheme $\mathsf{DOX}[\mathsf{FE},\mathsf{PRP},\mathsf{Obf}]$ is Res-PRIV secure. More precisely, for any adversary $(\mathcal{S},\mathcal{A})$ in game Res-PRIV against scheme $\mathsf{DOX}[\mathsf{FE},\mathsf{PRP},\mathsf{Obf}]$, in which $\mathcal{A}$ places at most q queries to TGen oracle and $\mathcal{S}$ outputs a tuple $(\mathbf{w}_0,\mathbf{w}_1,\mathbf{m}_0,\mathbf{m}_1,z)$ such that $|\mathbf{w}_0| = |\mathbf{w}_1| = t$ and $|\mathbf{m}_0| = |\mathbf{m}_1| = s$, there exists adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, $(\mathcal{S}_4,\mathcal{B}_4)$, $\mathcal{B}_5$, $(\mathcal{S}_6,\mathcal{B}_6)$ such that

$$\begin{aligned}\mathbf{Adv}^{\text{res-priv}}_{\mathsf{DOX},\mathcal{S},\mathcal{A}}(\lambda) \leq {} & 2\cdot\mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{B}_1}(\lambda) + 2\cdot(t+s+q)\cdot\mathbf{Adv}^{\text{ow}}_{\mathsf{Obf},\mathcal{B}_2}(\lambda) + \\ & 2\cdot\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_3}(\lambda) + 2\cdot\mathbf{Adv}^{\text{di}}_{\mathsf{Obf},\mathcal{S}_4,\mathcal{B}_4}(\lambda) + 2\cdot(\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_5}(\lambda) + \\ & \frac{s\cdot q}{|\mathsf{WSp}_\lambda|}) + \mathbf{Adv}^{\text{di}}_{\mathsf{Obf},\mathcal{S}_6,\mathcal{B}_6}(\lambda)\ .\end{aligned}$$

Proof. The proof follows from a sequence of six game hops. We refer the reader to Fig. 11 for a formal description of each game in a code-based language. Since the definition of Res-PRIV composes for stateless samplers, we assume $\mathcal{A}$ calls LR oracle exactly once.

Game$_0$: This game is identical to the Res-PRIV game.

Game$_1$: Instead of a PRP, a truly random permutation (simulated via lazy sampling) is used in token generation. The table used to maintain the lazy sampling, which we denote by T, has at most $(t+q)$ entries. The distance to the previous game can be bounded using the security of the PRP.

Game$_2$: When sampler $\mathcal{S}$ outputs, we generate PRP values of all messages in $\mathbf{m}_b$ as well. Since Game$_1$, these are now simulated via lazy sampling, which causes the expansion of table T to at most $(t+s+q)$ entries. All keywords and messages whose PRP value was generated are registered in list. Before setting the outcome of the game, if there are values w_1 and w_2 in list such that $\mathsf{w}_1 = T[\mathsf{w}_2]$, game aborts. Throughout the game $T[\mathsf{w}]$ is obfuscated, so the distance to the previous game can be upper bounded by the one-wayness property of the obfuscator.

Game$_3$: LR oracle computes the vector of ciphertexts $\mathbf{c}$ by encrypting $T[\mathbf{m}_b]$ instead of $\mathbf{m}_b$. The distance to the previous game can be upper bounded using the IND-CPA security property of the underlying FE scheme.

Game$_4$: We say a keyword w is *unpaired* if $\mathsf{w} \in \mathbf{w}_b$ and $\mathsf{w} \notin \mathbf{m}_b$. All first-phase queries to TGen oracle are recorded in $\mathsf{FirstPhase}$ list, i.e. all keywords $\mathcal{A}$ queries to TGen oracle before calling LR. During the simulation of LR oracle and second-phase TGen oracle, if w is an unpaired keyword not in $\mathsf{FirstPhase}$ list, we extract its token from circuit $(\mathsf{Obf}(1^\lambda,\mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda,\mathsf{C}[\mathsf{r}]))$, where r is a *fresh* random value uniformly sampled from WSp_λ. We precise that by *fresh* we mean that a new and independent random value is sampled each time, even in case of multiple token extractions of the same keyword. The distance to the previous game can be upper bounded to the DI security of the obfuscator.

Game$_5$: Analogously, we say a message m is *unpaired* if $\mathsf{m} \in \mathbf{m}_b$ and $\mathsf{m} \notin \mathbf{w}_b$. During the simulation of LR oracle, if m is an unpaired message not in $\mathsf{FirstPhase}$ list, we encrypt r instead of $T[\mathsf{m}]$, where r is a *fresh* random value uniformly sampled from WSp_λ. We precise that by *fresh* we mean that a new and independent random value is sampled each time, even in case of repetitions of the same message. We bound this hop down to IND-CPA.

Game$_6$: In this game, irrespective of the bit b, we use the second set of keywords for challenge token generation.

We now analyze the transitions between each game and the reduction of Game$_5$ to DI game.

Game$_0$ TO Game$_1$. Any adversary $(\mathcal{S}, \mathcal{A})$ with visible advantage difference in these two games can be converted to an adversary $\mathcal{B}_1$ against the security of PRP. Assume that lazy sampling is implemented using a table T, i.e., $T[\mathsf{w}]$ indicates the random value assigned to w. Algorithm $\mathcal{B}_1$ runs adversary $(\mathcal{S}, \mathcal{A})$ inside it, simulating all the details of Game$_0$, bar the computation of the PRP. For this, algorithm $\mathcal{B}_1$ uses its FN oracle. When $\mathcal{A}$ terminates, $\mathcal{B}_1$ checks if $\mathcal{A}$ succeeded in winning the game. If so, it outputs 0. Otherwise, it outputs 1.

When the FN oracle implements the PRP, Game$_0$ is simulated for $\mathcal{A}$, and when FN implements a random permutation, Game$_1$ is simulated. Therefore,

$$\Pr[\mathrm{Game}_0(1^\lambda)] - \Pr[\mathrm{Game}_1(1^\lambda)] = \mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{B}_1}(\lambda)\ .$$

Game$_1$ TO Game$_2$. Both games are exactly the same unless the bad event that causes abortion is triggered. Game$_2$ aborts if there are values w_1 and w_2 in list such that $\mathsf{w}_1 = T[\mathsf{w}_2]$. We show that an adversary $(\mathcal{S}, \mathcal{A})$ that triggers the bad event in Game$_2$ can be converted to an adversary $\mathcal{B}_2$ against the one-wayness property of Obf. For an intuition on this game hop, observe that all occurrences of $T[\mathsf{w}]$ are obfuscated and $T[\mathsf{w}]$ is uniformly distributed.

During the simulation of Game$_2$, table T expands up to $(t + s + q)$ entries. Algorithm $\mathcal{B}_2$ receives in its challenge $(t + q)$ obfuscated copies of a random point circuit. At the beginning of its execution, $\mathcal{B}_2$ randomly guesses the first occurrence of w_2 in the game, by sampling i is uniformly from $\{1, ..., (t + s + q)\}$. (Keyword w_2 is of course unknown to $\mathcal{B}_2$ at this point, the guess reflects a prediction of when such keyword involved in the bad event will be added to table T.) Then, $\mathcal{B}_2$ simulates for adversary $(\mathcal{S}, \mathcal{A})$ all the details of Game$_2$ until a new keyword comes that will cause table T to expand to i entries. Instead of sampling $T[\mathsf{w}_2]$, $\mathcal{B}_2$ embeds one of its challenge circuits in the computation of $\mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}_2]])$. (If w_2 is a message, nothing needs to be done.) Thenceforth, $\mathcal{B}_2$ embeds a new circuit from its challenge each time it needs to extract a token for w_2. In any case, $\mathcal{B}_2$ never needs more than $(t + q)$ challenge circuits to complete its simulation.

At the end of the game, if $\mathcal{B}_2$'s guess is correct, which happens with probability $1/(t + s + q)$, there is $\mathsf{w}_1 \in \mathsf{list}$ such that $\mathsf{w}_1 = T[\mathsf{w}_2]$. This equality can be checked by evaluating w_1 on one of $\mathcal{B}_2$'s obfuscated circuits. If so, $\mathcal{B}_2$ outputs w_1 and wins the game. Hence,

$$\Pr[\mathrm{Game}_1(1^\lambda)] - \Pr[\mathrm{Game}_2(1^\lambda)] \leq (t + s + q) \cdot \mathbf{Adv}^{\mathsf{ow}}_{\mathsf{Obf},\mathcal{B}_2}(\lambda)\ .$$

Game$_2$ TO Game$_3$. Any legitimate adversary $(\mathcal{S}, \mathcal{A})$ with visible advantage difference in these two games can be converted to an adversary $\mathcal{B}_3$ against IND-CPA security of FE. For an intuition on this reduction, observe that all tokens are extracted from circuits of the form $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))$, which return 1 when evaluated on both w and $T[\mathsf{w}]$. Illegitimate tokens that would allow to distinguish encryptions of m from encryption of $T[\mathsf{m}]$ have been excluded in Game$_2$, given that the game aborts if adversary $(\mathcal{S}, \mathcal{A})$ outputs a value sampled for the simulation of the random permutation.

Algorithm $\mathcal{B}_3$ runs adversary $(\mathcal{S}, \mathcal{A})$ inside it, simulating all the details common to Game$_2$ and Game$_3$. $\mathcal{B}_3$ receives mpk and runs adversary $\mathcal{A}$ with it. For token-generation and encryption, $\mathcal{B}_3$ relies on its oracles. When $\mathcal{B}_3$ needs to compute a token for some keyword w, it queries its own TGen oracle with circuit $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))$. When $\mathcal{B}_3$ needs to compute the encryption of some message m in Game$_2$ or $T[\mathsf{m}]$ in Game$_3$, it queries its own LR oracle with $(\mathsf{m}, T[\mathsf{m}])$. The ciphertexts output by LR oracle in game IND-CPA allow $\mathcal{B}_3$ to interpolate between the simulation of Game$_2$ and Game$_3$. The simulation is perfect. Eventually, $\mathcal{A}$ outputs b', which $\mathcal{B}_3$ forwards as its own guess.

Now, let's analyze legitimacy of $\mathcal{B}_3$. Legitimacy condition of IND-CPA requires that for all C queried to TGen and all $(\mathsf{m}_0, \mathsf{m}_1)$ queried to LR, we have that $\mathsf{C}(\mathsf{m}_0) = \mathsf{C}(\mathsf{m}_1)$. In the execution of $\mathcal{B}_3$, queried circuits are of the form $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))$ and queried messages of the form $(\mathsf{m}, T[\mathsf{m}])$. More precisely, legitimacy requires that $\forall \mathsf{w} \in \mathsf{TList}, \forall \mathsf{m} \in \mathsf{MList}$,

$$\begin{aligned}
(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))(\mathsf{m}) &= (\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))(T[\mathsf{m}]) \\
\Leftrightarrow (\mathsf{C}[\mathsf{w}] \vee \mathsf{C}[T[\mathsf{w}]])(\mathsf{m}) &= (\mathsf{C}[\mathsf{w}] \vee \mathsf{C}[T[\mathsf{w}]])(T[\mathsf{m}]) \quad /\!/ \text{ functionality preserving} \\
\Leftrightarrow \mathsf{C}[\mathsf{w}](\mathsf{m}) \vee \mathsf{C}[T[\mathsf{w}]](\mathsf{m}) &= \mathsf{C}[\mathsf{w}](T[\mathsf{m}]) \vee \mathsf{C}[T[\mathsf{w}]](T[\mathsf{m}]) \\
\Leftrightarrow \mathsf{C}[\mathsf{w}](\mathsf{m}) &= \mathsf{C}[T[\mathsf{w}]](T[\mathsf{m}]) \quad /\!/ \text{ bad event in Game}_2 \\
\Leftrightarrow (\mathsf{w} \overset{?}{=} \mathsf{m}) &= (T[\mathsf{w}] \overset{?}{=} T[\mathsf{m}]) \\
\Leftrightarrow & \mathsf{True}\ .
\end{aligned}$$

Therefore, $\mathcal{B}_3$ is a legitimate adversary against IND-CPA and we have that

$$\Pr[\text{Game}_2(1^\lambda)] - \Pr[\text{Game}_3(1^\lambda)] = \mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_3}(\lambda)\ .$$

Game$_3$ to Game$_4$. Any legitimate adversary $(\mathcal{S}, \mathcal{A})$ with visible advantage difference in these two games can be converted to an adversary $(\mathcal{S}_4, \mathcal{B}_4)$ against DI security of Obf. The intuition here is the following: Without a ciphertext that encrypts $T[\mathsf{w}]$, the adversary cannot detect if tokens for w are extracted from $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))$ or from $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}]))$, where r is a fresh random value uniformly sampled from WSp_λ. Details of adversary $(\mathcal{S}_4, \mathcal{B}_4)$ are shown in Fig. 8.

Sampler $\mathcal{S}_4$ computes two $t \times (t + q)$ matrices $\mathbf{M}_0$ and $\mathbf{M}_1$. Each row in $\mathbf{M}_0$ contains $(t + q)$ repetitions of a *unique* random point circuit. $\mathbf{M}_1$ contains $t \times (t + q)$ fresh random point circuits. $\mathcal{S}_4$ is clearly unpredictable. Algorithm $\mathcal{B}_4$ runs $\mathcal{S}$ and $\mathcal{A}$ inside it, simulating all the details common to Game$_3$ and Game$_4$, which only differ on unpaired keywords not in $\mathsf{FirstPhase}$ list. For those, $\mathcal{B}_4$ carefully picks circuits from its challenge matrix of obfuscated circuits: A new row is assigned to a new keyword; a circuit is picked from a new column in case of repetitions. If $\mathbf{M}_0$ is selected in game DI, algorithm $\mathcal{B}_4$ will simulate Game$_3$. On the other hand, if $\mathbf{M}_1$ is selected in game DI, algorithm $\mathcal{B}_4$ will simulate Game$_4$. Finally, when $\mathcal{A}$ outputs its guess, $\mathcal{B}_4$ checks if $\mathcal{A}$ succeeded in winning the game. If so, $\mathcal{B}_4$ outputs 0. Otherwise, it outputs 1. Therefore, we have that

$$\Pr[\text{Game}_3(1^\lambda)] - \Pr[\text{Game}_4(1^\lambda)] = \mathbf{Adv}^{\text{di}}_{\mathsf{Obf},\mathcal{S}_4,\mathcal{B}_4}(\lambda)\ .$$

Game$_4$ to Game$_5$. Any legitimate adversary $(\mathcal{S}, \mathcal{A})$ with visible advantage difference in these two games can be converted to an adversary $\mathcal{B}_5$ against IND-CPA security of FE. The intuition here is

```
$\mathcal{S}_4(1^\lambda, st)$:
for $i \in \{1, ..., t\}$
    $T[i] \leftarrow\$ \ \mathsf{WSp}_\lambda \setminus T$
    for $j \in \{1, ..., (t+q)\}$
        $\mathbf{M}_0[i][j] \leftarrow \mathsf{C}[T[i]]$
        $\mathsf{r} \leftarrow\$ \ \mathsf{WSp}_\lambda$
        $\mathbf{M}_1[i][j] \leftarrow \mathsf{C}[\mathsf{r}]$
return $(\mathbf{M}_0, \mathbf{M}_1, \epsilon)$
```

```
$\mathcal{B}_4(1^\lambda)$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \ \{0, 1\}$
$\overline{\mathbf{M}} \leftarrow\$ \ \mathrm{DI.S{\small AM}}(\epsilon)$
$b' \leftarrow\$ \ \mathcal{A}^{\mathrm{LR,TG{\small EN}}}(\mathsf{mpk})$
return $(b = b')$

$\mathrm{TG{\small EN}}(\mathsf{w})$:
$\overline{\mathsf{C}}_\mathsf{L} \leftarrow\$ \ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
    $T[\mathsf{w}] \leftarrow\$ \ \mathsf{WSp}_\lambda \setminus T$
    $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
if $(\mathsf{w} \in \mathbf{w}_b \wedge \mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$
    $\overline{\mathsf{C}}_\mathsf{R} \leftarrow \mathsf{Select}(\mathsf{w})$
else
    $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_\mathsf{L} \vee \overline{\mathsf{C}}_\mathsf{R}))$
return $\mathsf{tk}$
```

```
$\mathrm{LR}(st)$:
$\mathsf{FirstPhase} \leftarrow \mathsf{List}$
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
    if $T[\mathsf{w}] = \perp$ then
        $T[\mathsf{w}] \leftarrow\$ \ \mathsf{WSp}_\lambda \setminus T$
        $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\mathbf{c} \leftarrow\$ \ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathbf{m}_b])$
for all $\mathsf{w} \in \mathbf{w}_b$
    $\overline{\mathsf{C}}_\mathsf{L} \leftarrow\$ \ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
    if $(\mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$ then
        $\overline{\mathsf{C}}_\mathsf{R} \leftarrow \mathsf{Select}(\mathsf{w})$
else
    $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
    $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_\mathsf{L} \vee \overline{\mathsf{C}}_\mathsf{R}))$
return $(\mathbf{tk}, \mathbf{c}, z)$
```

```
$\mathsf{Select}(\mathsf{w})$ :
if $(\mathsf{Row}[\mathsf{w}] = \perp)$ then
    // assign next available row to w
    $\mathsf{Row}[\mathsf{w}] \leftarrow |\mathsf{Row}| + 1$
    // initialize counter for w
    $\mathsf{Column}[\mathsf{Row}[\mathsf{w}]] \leftarrow 0$
// increase counter for w
$\mathsf{Column}[\mathsf{Row}[\mathsf{w}]] \leftarrow \mathsf{Column}[\mathsf{Row}[\mathsf{w}]] + 1$
// return fresh obfuscated circuit
return $(\overline{\mathbf{M}}[\mathsf{Row}[\mathsf{w}]][\mathsf{Column}[\mathsf{Row}[\mathsf{w}]]])$
```

Fig. 8. DI adversary $\mathcal{B}_4$ and circuit sampler $\mathcal{S}_4$ constructed from PRIV adversary $\mathcal{A}$ and keyword sampler $\mathcal{S}$.

simple: Without a token for m, the adversary cannot detect if we encrypt a fresh random value r instead if $T[\mathsf{m}]$. Details of adversary $\mathcal{B}_5$ are shown in Fig. 9.

Let d denote the challenge bit in the IND-CPA game for FE. Let d' denote $\mathcal{B}_5$'s output. By definition, we have that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_5}(\lambda) = \Pr[d' = 0 | d = 0] - \Pr[d' = 0 | d = 1].$$

Also, let Ter be the event that $\mathcal{B}_5$ terminates because $\mathcal{A}$ queried $\mathsf{w} \in \mathsf{RList}$. We have that

$$\begin{aligned}
\Pr[d' = 0 | d = 0] &= \Pr[d' = 0 | d = 0 \wedge \neg\mathsf{Ter}] \cdot \Pr[\neg\mathsf{Ter}] + \Pr[d' = 0 | d = 0 \wedge \mathsf{Ter}] \cdot \Pr[\mathsf{Ter}] \\
&= \Pr[d' = 0 | d = 0 \wedge \neg\mathsf{Ter}] \cdot \Pr[\neg\mathsf{Ter}] + 0 \quad /\!\!/ \ \mathcal{B}_5 \text{ never outputs 0 if } \mathsf{Ter} \\
&= \Pr[d' = 0 | d = 0 \wedge \neg\mathsf{Ter}] \cdot \Big(1 - \frac{s \cdot q}{|\mathsf{WSp}_\lambda|}\Big) \ /\!\!/ \ d = 0, \text{ so } \mathsf{RList} \text{ is hidden from } \mathcal{A} \\
&\geq \Pr[d' = 0 | d = 0 \wedge \neg\mathsf{Ter}] - \frac{s \cdot q}{|\mathsf{WSp}_\lambda|} = \Pr[\mathrm{Game}_4(1^\lambda)] - \frac{s \cdot q}{|\mathsf{WSp}_\lambda|}
\end{aligned}$$

$$\begin{aligned}
\Pr[d' = 0 | d = 1] &= \Pr[d' = 0 | d = 1 \wedge \neg\mathsf{Ter}] \cdot \Pr[\neg\mathsf{Ter}] + \Pr[d' = 0 | d = 1 \wedge \mathsf{Ter}] \cdot \Pr[\mathsf{Ter}] \\
&= \Pr[d' = 0 | d = 1 \wedge \neg\mathsf{Ter}] \cdot \Pr[\neg\mathsf{Ter}] + 0 \quad /\!\!/ \ \mathcal{B}_5 \text{ never outputs 0 if } \mathsf{Ter} \\
&\leq \Pr[d' = 0 | d = 1 \wedge \neg\mathsf{Ter}] = \Pr[\mathrm{Game}_5(1^\lambda)].
\end{aligned}$$

We now analyze legitimacy of $\mathcal{B}_5$. Legitimacy condition of IND-CPA requires that for all C queried to TGEN and all $(\mathsf{m}_0, \mathsf{m}_1)$ queried to LR, we have that $\mathsf{C}(\mathsf{m}_0) = \mathsf{C}(\mathsf{m}_1)$. In the execution of $\mathcal{B}_5$, queried circuits are of the form $(\mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}]) \vee \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]]))$ and queried messages of the form $(T[\mathsf{m}], \mathsf{r})$. More precisely, legitimacy requires that $\forall \mathsf{w} \in \mathsf{TList}, \forall \mathsf{m} \in \mathbf{m}_b$ s.t. $\mathsf{m} \notin \mathbf{w}_b \wedge \mathsf{m} \notin \mathsf{FirstPhase}, \forall \mathsf{r} \in \mathsf{RList}$, we have that

$$\begin{aligned}
(\mathsf{Obf}(1^\lambda,\mathsf{C}[\mathsf{w}])\vee\mathsf{Obf}(1^\lambda,\mathsf{C}[T[\mathsf{w}]]))(T[\mathsf{m}]) &= (\mathsf{Obf}(1^\lambda,\mathsf{C}[\mathsf{w}])\vee\mathsf{Obf}(1^\lambda,\mathsf{C}[T[\mathsf{w}]]))(\mathsf{r})\\
\Leftrightarrow (\mathsf{C}[\mathsf{w}]\vee\mathsf{C}[T[\mathsf{w}]])(T[\mathsf{m}]) &= (\mathsf{C}[\mathsf{w}]\vee\mathsf{C}[T[\mathsf{w}]])(\mathsf{r}) \quad /\!/ \text{ functionality preserving}\\
\Leftrightarrow \mathsf{C}[\mathsf{w}](T[\mathsf{m}])\vee\mathsf{C}[T[\mathsf{w}]](T[\mathsf{m}]) &= \mathsf{C}[\mathsf{w}](\mathsf{r})\vee\mathsf{C}[T[\mathsf{w}]](\mathsf{r})\\
\Leftrightarrow \mathsf{C}[T[\mathsf{w}]](T[\mathsf{m}]) &= \mathsf{C}[\mathsf{w}](\mathsf{r})\vee\mathsf{C}[T[\mathsf{w}]](\mathsf{r}) \quad /\!/ \text{ bad event in Game}_2\\
\Leftrightarrow \mathsf{C}[T[\mathsf{w}]](T[\mathsf{m}]) &= \mathsf{C}[T[\mathsf{w}]](\mathsf{r}) \quad /\!/\ \mathcal{B}_5 \text{ outputs 1, } \mathsf{Ter} \text{ event}\\
\Leftrightarrow \mathsf{C}[\mathsf{w}](\mathsf{m}) &= \mathsf{C}[T[\mathsf{w}]](\mathsf{r})\\
\Leftrightarrow 0 &= \mathsf{C}[T[\mathsf{w}]](\mathsf{r}) \quad /\!/ \text{ Res-PRIV restriction}\\
\Leftrightarrow 0 &= 0 \quad /\!/ \text{ with overwhelming probability } \mathsf{C}[T[\mathsf{w}]](\mathsf{r}) = 0\ .
\end{aligned}$$

Therefore,

$$\Pr[\text{Game}_4(1^\lambda)] - \Pr[\text{Game}_5(1^\lambda)] \le \mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{FE},\mathcal{B}_5}(\lambda) + \frac{s\cdot q}{|\mathsf{WSp}_\lambda|}.$$

```
$\mathcal{B}_5(1^\lambda, \mathsf{mpk})$:
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
if $(\exists\ \mathsf{w}_1, \mathsf{w}_2 \in \mathsf{List}$ s.t. $\mathsf{w}_1 = T[\mathsf{w}_2])$ abort
return $(\neg(b = b'))$
```

```
$\mathrm{LR}(st)$:
$\mathsf{FirstPhase} \leftarrow \mathsf{List}$
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
      $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
for all $\mathsf{m} \in \mathbf{m}_b$
   if $(\mathsf{m} \notin \mathbf{w}_b \wedge \mathsf{m} \notin \mathsf{FirstPhase})$ then
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\mathsf{RList} \leftarrow \mathsf{RList} : \mathsf{r}$
      $\mathsf{c} \leftarrow\$ \mathrm{IND\text{-}CPA.LR}(T[\mathsf{m}], \mathsf{r})$
   else
      $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathsf{m}])$
   $\mathbf{c} \leftarrow \mathbf{c} : \mathsf{c}$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_\mathsf{L} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   if $(\mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$ then
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
   else
      $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathrm{IND\text{-}CPA.TGen}((\overline{\mathsf{C}}_\mathsf{L} \vee \overline{\mathsf{C}}_\mathsf{R}))$
return $(\mathbf{tk}, \mathbf{c}, z)$
```

```
$\mathrm{TGen}(\mathsf{w})$:
if $\mathsf{w} \in \mathsf{RList}$ exit 1 // $\mathcal{B}_5$ terminates and outputs 1
$\overline{\mathsf{C}}_\mathsf{L} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
if $(\mathsf{w} \in \mathbf{w}_b \wedge \mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$
   $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
   $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
else
   $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathrm{IND\text{-}CPA.TGen}((\overline{\mathsf{C}}_\mathsf{L} \vee \overline{\mathsf{C}}_\mathsf{R}))$
return $\mathsf{tk}$
```

Fig. 9. IND-CPA adversary $\mathcal{B}_5$ constructed from PRIV adversary $\mathcal{A}$ and keyword sampler $\mathcal{S}$.

Game$_5$ TO Game$_6$. In this game, irrespective of the bit, we use the second set of keywords for challenge token generation. We construct an adversary $(\mathcal{S}_6, \mathcal{B}_6)$ against DI as follows. $\mathcal{B}_6$ generates by itself a master secret key and master public key pair $(\mathsf{msk}, \mathsf{mpk})$, then runs $\mathcal{A}(\mathsf{mpk})$. First-phase TGen queries are answered using a lazily sampled T and a generated msk. We set the DI sampler $\mathcal{S}_6$ to run the PRIV sampler $\mathcal{S}$ and on top of the output keywords, also ask for obfuscations of *messages* that match a first-phase query. By legitimacy of $\mathcal{A}$, these messages must be LR-identical. Using the symmetry of roles for keywords and messages in point functions, this sampler can be shown to be unpredictable whenever the PRIV sampler is. We find paired messages using the image matrix. The obfuscations of messages will allow us to *check* if any of these messages (hidden under the obfuscation) match a first-phase TGen query. For messages that were queried during the first stage, we select the correct T-value. For messages that are at the same time new and paired, we sample a new T-value the first time and answer consistently throughout the game. For all unpaired messages that were never queried to TGen, we use forgetful values (rules of Game$_5$). Second phase TGen queries are answered using T-consistent values relying on the fact that we can use the obfuscations to check matches with paired keywords and the restriction that adversary cannot query a new unpaired message to TGen. Finally, we output whatever the PRIV adversary $\mathcal{A}$ outputs. More details on how to construct $\mathcal{S}_6, \mathcal{B}_6$ are available in Fig. 10.

$$\Pr[\mathrm{Game}_5(1^\lambda)] - \Pr[\mathrm{Game}_6(1^\lambda)] \leq \Pr[\mathrm{Game}_5(1^\lambda)] - \frac{1}{2} = \frac{1}{2} \cdot \mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S}_6,\mathcal{B}_6}(\lambda) \ .$$

It remains to show that $\mathcal{S}_6$ is unpredictable if $\mathcal{S}$ is. For this, we build a predictor $\mathcal{Q}$ against sampler $\mathcal{S}$, from a predictor $\mathcal{P}$ against sampler $\mathcal{S}_6$ (bottom of Fig. 10). Since $\mathcal{S}_6$ outputs the same vector of circuits as $\mathcal{S}$ plus circuits that are LR-identical, a distinguishing message for the output of $\mathcal{S}_6$ is also a distinguishing message for the output of $\mathcal{S}$. Therefore, we have that

$$\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{Q}}(\lambda) = \mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S}_6,\mathcal{P}}(\lambda) \ .$$

To conclude our proof, we put everything together:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{res\text{-}priv}}_{\mathsf{DOX},\mathcal{S},\mathcal{A}}(\lambda) &:= 2 \cdot \Pr[\mathrm{Game}_0(1^\lambda)] - 1 \\ &\leq 2 \cdot \mathbf{Adv}^{\mathsf{prp}}_{\mathsf{PRP},\mathcal{B}_1}(\lambda) + 2 \cdot (t+s+q) \cdot \mathbf{Adv}^{\mathrm{ow}}_{\mathsf{Obf},\mathcal{B}_2}(\lambda) + \\ &\quad 2 \cdot \mathbf{Adv}^{\mathrm{ind\text{-}cpa}}_{\mathsf{FE},\mathcal{B}_3}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S}_4,\mathcal{B}_4}(\lambda) + 2 \cdot (\mathbf{Adv}^{\mathrm{ind\text{-}cpa}}_{\mathsf{FE},\mathcal{B}_5}(\lambda) + \\ &\quad \frac{s \cdot q}{|\mathsf{WSp}_\lambda|}) + \mathbf{Adv}^{\mathrm{di}}_{\mathsf{Obf},\mathcal{S}_6,\mathcal{B}_6}(\lambda) \ .\end{aligned}$$

□

```
$\underline{\mathcal{S}_6(1^\lambda, st):}$
$(st', \mathsf{FirstPhase}) \leftarrow st$
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z') \leftarrow\$ \mathcal{S}(1^\lambda, st)$
for $i \in \{1, \ldots, s\}$
   if $(\mathbf{m}_0[i] \in \mathsf{FirstPhase})$
      // by legitimacy of $\mathcal{A}$, $\mathbf{m}_0[i] \in \mathsf{FirstPhase} \Rightarrow \mathbf{m}_0[i] = \mathbf{m}_1[i]$
      // the following line ensures unpredictability of $\mathcal{S}_6$
      if $(\mathbf{m}_0[i] \neq \mathbf{m}_1[i])$ return $([], [], \perp)$
      $\mathbf{m}^\star \leftarrow \mathbf{m}^\star : \mathbf{m}_0[i]$
   else
      $\mathbf{m}^\star \leftarrow \mathbf{m}^\star : \perp$ // $\mathsf{C}[\perp](.) := 0$ is the zero circuit
$z \leftarrow (z', \mathbf{C}[\mathbf{w}_0](\mathbf{m}_0))$
return $(\mathbf{w}_0 : \mathbf{m}^\star, \mathbf{w}_1 : \mathbf{m}^\star, z)$
```

```
$\underline{\mathcal{B}_6(1^\lambda):}$
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR}, \mathrm{TGEN}}(\mathsf{mpk})$
return $b'$
```

```
$\underline{\mathrm{LR}(st):}$
$\mathsf{FirstPhase} \leftarrow \mathsf{List}$
$(\overline{\mathbf{C}}, z) \leftarrow\$ \mathrm{DI.SAM}((st, \mathsf{FirstPhase}))$
$(z', \mathsf{ImgMatrix}) \leftarrow z$
for $i \in \{1, \ldots, s\}$
   $\mathsf{flag} \leftarrow 0$
   if $(\mathsf{ImgMatrix}[i][] \neq [0, \ldots, 0])$
      $\mathsf{flag} \leftarrow 1$ // $\mathbf{m}_b[i] \in \mathbf{w}_b$
   for $\mathsf{w} \in \mathsf{FirstPhase}$
      if $(\overline{\mathbf{C}}[(t+i)](\mathsf{w}) = 1)$
         $\mathsf{flag} \leftarrow 2$ // $\mathbf{m}_b[i] \in \mathsf{FirstPhase}$
         $\mathsf{m}^\star \leftarrow \mathsf{w}$ // $\mathbf{m}_b[i] = \mathsf{w}$
   if $(\mathsf{flag} = 0)$ // encrypt random message
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{r})$
   if $(\mathsf{flag} = 1)$ // encrypt $T$-consistent
      $(\ldots)$
   if $(\mathsf{flag} = 2)$ // encrypt $T[\mathsf{m}^\star]$
      $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathsf{m}^\star])$
   $\mathbf{c} \leftarrow \mathbf{c} : \mathsf{c}$
$(\ldots)$
return $(\mathbf{tk}, \mathbf{c}, z')$
```

```
$\underline{\mathrm{TGEN}(\mathsf{w}):}$
$\overline{\mathsf{C}}_\mathsf{L} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\mathsf{flagRandom} \leftarrow 0$
for $i \in \{1, \ldots, t\}$
   if $(\overline{\mathbf{C}}[i](\mathsf{w}) = 1)$
      // $\mathsf{w} \in \mathbf{w}_b$
      if $(\mathsf{ImgMatrix}[][i] = [0, \ldots, 0])$
         // $\mathsf{w} \notin \mathbf{m}_b$
         if $\mathsf{w} \notin \mathsf{FirstPhase}$
            $\mathsf{flagRandom} \leftarrow 1$
if $(\mathsf{flagRandom} = 1)$
   $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
   $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
else
   $\overline{\mathsf{C}}_\mathsf{R} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_\mathsf{L} \vee \overline{\mathsf{C}}_\mathsf{R}))$
return $\mathsf{tk}$
```

```
$\underline{\mathcal{Q}_1(1^\lambda):}$
$(st, st') \leftarrow\$ \mathcal{P}_1(1^\lambda)$
$(st'', \mathsf{FirstPhase}) \leftarrow st$
return $(st'', (st', \mathsf{FirstPhase}))$
```

```
$\underline{\mathcal{Q}_2^{\mathrm{FUNC}}(1^\lambda, \mathbf{C}[\mathbf{w}_0](\mathbf{m}_0), z', (st', \mathsf{FirstPhase})):}$
$\mathsf{m} \leftarrow\$ \mathcal{P}_2^{\mathrm{FUNC}'}(1^\lambda, \epsilon, (z', \mathbf{C}[\mathbf{w}_0](\mathbf{m}_0)), st')$
return $\mathsf{m}$

$\underline{\mathrm{FUNC}'(\mathsf{m}, -):}$
$(\mathbf{C}[\mathbf{w}_0](\mathsf{m}), -) \leftarrow \mathrm{FUNC}(\mathsf{m}, -)$
if $(\mathsf{m} \in \mathsf{FirstPhase})$
   $(-, \mathsf{C}[\mathsf{m}](\mathbf{m}_0)) \leftarrow \mathrm{FUNC}(-, \mathsf{m})$ // if this query is not legit, $\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S}_6, \mathcal{P}}(\lambda) = 0$
   return $(\mathbf{C}[\mathbf{w}_0](\mathsf{m}) : \mathsf{C}[\mathsf{m}](\mathbf{m}_0))$
else
   return $(\mathbf{C}[\mathbf{w}_0](\mathsf{m}) : [0, \ldots, 0])$
```

Fig. 10. Top: Circuit sampler $\mathcal{S}_6$ built from PRIV sampler $\mathcal{S}$. **Middle**: DI adversary $\mathcal{B}_6$ built from PRIV adversary $\mathcal{A}$. **Bottom**: Predictor $\mathcal{Q}$ against $\mathcal{S}$, built from predictor $\mathcal{P}$ against $\mathcal{S}_6$.

```
$\underline{\mathrm{Game}_0(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$\mathsf{k} \leftarrow\$ \mathsf{KSp}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathbf{m}_b)$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{E}(\mathsf{k}, \mathsf{w})])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
$\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{E}(\mathsf{k}, \mathsf{w})])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

```
$\underline{\mathrm{Game}_1(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
~~$\mathsf{k} \leftarrow\$ \mathsf{KSp}(1^\lambda)$~~
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathbf{m}_b)$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
$\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

```
$\underline{\mathrm{Game}_2(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
if $(\exists\ \mathsf{w}_1, \mathsf{w}_2 \in \mathsf{List}$ s.t. $\mathsf{w}_1 = T[\mathsf{w}_2])$ abort
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
      $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathbf{m}_b)$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

```
$\underline{\mathrm{Game}_3(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
if $(\exists\ \mathsf{w}_1, \mathsf{w}_2 \in \mathsf{List}$ s.t. $\mathsf{w}_1 = T[\mathsf{w}_2])$ abort
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
      $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathbf{m}_b])$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

```
$\underline{\mathrm{Game}_4(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
if $(\exists\ \mathsf{w}_1, \mathsf{w}_2 \in \mathsf{List}$ s.t. $\mathsf{w}_1 = T[\mathsf{w}_2])$ abort
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$\mathsf{FirstPhase} \leftarrow \mathsf{List}$
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
      $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
$\mathbf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathbf{m}_b])$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   if $(\mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$ then
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
   else
      $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
if $(\mathsf{w} \in \mathbf{w}_b \wedge \mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$
   $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
else
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

```
$\underline{\mathrm{Game}_5(1^\lambda)}$:
$(\mathsf{msk}, \mathsf{mpk}) \leftarrow\$ \mathsf{FE.Gen}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{LR},\mathrm{TGen}}(\mathsf{mpk})$
if $(\exists\ \mathsf{w}_1, \mathsf{w}_2 \in \mathsf{List}$ s.t. $\mathsf{w}_1 = T[\mathsf{w}_2])$ abort
return $(b = b')$

$\underline{\mathrm{LR}(st)}$:
$\mathsf{FirstPhase} \leftarrow \mathsf{List}$
$(\mathbf{w}_0, \mathbf{w}_1, \mathbf{m}_0, \mathbf{m}_1, z) \leftarrow\$ \mathcal{S}(st)$
for all $\mathsf{w} \in \mathbf{w}_b \cup \mathbf{m}_b$
   if $T[\mathsf{w}] = \perp$ then
      $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
      $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
for all $\mathsf{m} \in \mathbf{m}_b$
   if $(\mathsf{m} \notin \mathbf{w}_b \wedge \mathsf{m} \notin \mathsf{FirstPhase})$ then
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{r})$
   else
      $\mathsf{c} \leftarrow\$ \mathsf{FE.Enc}(\mathsf{mpk}, T[\mathsf{m}])$
   $\mathbf{c} \leftarrow \mathbf{c} : \mathsf{c}$
for all $\mathsf{w} \in \mathbf{w}_b$
   $\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
   if $(\mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$ then
      $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
      $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
   else
      $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
   $\mathbf{tk} \leftarrow \mathbf{tk} : \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $(\mathbf{tk}, \mathbf{c}, z)$

$\underline{\mathrm{TGen}(\mathsf{w})}$:
$\overline{\mathsf{C}}_{\mathsf{L}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{w}])$
if $T[\mathsf{w}] = \perp$ then
   $T[\mathsf{w}] \leftarrow\$ \mathsf{WSp}_\lambda \setminus T$
   $\mathsf{List} \leftarrow \mathsf{List} : \mathsf{w}$
if $(\mathsf{w} \in \mathbf{w}_b \wedge \mathsf{w} \notin \mathbf{m}_b \wedge \mathsf{w} \notin \mathsf{FirstPhase})$
   $\mathsf{r} \leftarrow\$ \mathsf{WSp}_\lambda$
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[\mathsf{r}])$
else
   $\overline{\mathsf{C}}_{\mathsf{R}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathsf{C}[T[\mathsf{w}]])$
$\mathsf{tk} \leftarrow\$ \mathsf{FE.TGen}(\mathsf{msk}, (\overline{\mathsf{C}}_{\mathsf{L}} \vee \overline{\mathsf{C}}_{\mathsf{R}}))$
return $\mathsf{tk}$
```

Fig. 11. Sequence of games in proof of Theorem F.2.

G Distributional Indistinguishability for Hyperplane Membership

Hyperplane membership. Let $\rho = \{\rho(\lambda)\}$ be a family of primes such that, for all $\lambda \in \mathbb{N}$, we have $2^{\lambda-1} \leq \rho(\lambda) < 2^{\lambda}$. Let $\mathcal{F}^d_\rho = \{\mathcal{F}^d_{\rho(\lambda)}\}$ be the family of hyperplane sets that is defined, at each value of the security parameter as follows. Take each element $\boldsymbol{a} \in \mathbb{F}^d_{\rho(\lambda)}$ such that the first coordinate $\boldsymbol{a}[1] = 1_{\mathbb{F}_{\rho(\lambda)}}$. The set of vectors $x \in \mathbb{F}^d_{\rho(\lambda)}$ that are orthogonal to $\boldsymbol{a}$ defines a hyperplane. We let $\mathcal{F}^d_{\rho(\lambda)}$ be the set of all such hyperplanes, and take $\boldsymbol{a}$ as its canonical representation. Given a vector $\boldsymbol{a} \in \mathbb{F}^d_{\rho(\lambda)}$, we define the hyperplane membership functionality as the circuit that, given some vector $\boldsymbol{x} \in \mathbb{F}^d_{\rho(\lambda)}$, returns 1 if and only if $\langle \boldsymbol{x}, \boldsymbol{a} \rangle = 0$. We also assume that such circuits are canonically represented by $\boldsymbol{a}$ itself.

VBB obfuscator. Canetti, Rothblum and Varia [23] presented a virtual black-box obfuscator for the hyperplane membership functionality (i.e., inner products), which works as follows. Let $\mathcal{G} = \{\mathcal{G}_\lambda\}$ be a family of (multiplicative) groups where $|\mathcal{G}_\lambda| = \rho(\lambda)$, for all λ. To obfuscate the hyperplane membership circuit represented by a vector $\boldsymbol{a}$, sample a generator g uniformly at random from $\mathcal{G}_\lambda \setminus 1_{\mathcal{G}_\lambda}$, compute $g_i \leftarrow g^{\boldsymbol{a}[i]}$ for $1 \leq i \leq d$, and construct the circuit that, given a vector $\boldsymbol{x}$, returns 1 if and only if $\prod_{i=1}^{d} g_i^{\boldsymbol{x}[i]} = 1_{\mathcal{G}_\lambda}$. We assume that the obfuscated circuit $\mathsf{Obf}_{\mathcal{G}}(\boldsymbol{a})$ is canonically represented by $(g, \ldots, g_d)$, generated as described above. We will now prove that this same construction satisfies distributional indistinguishability under a variant of the DDH assumption.

Unpredictable hyperplane membership samplers. We begin by refining the notion of unpredictable samplers to the case of hyperplane membership circuits. In general, sampler for the hyperplane membership functionality will output two lists of message vectors corresponding to candidate hyperplane members, and two lists of hyperplane defining vectors, plus some auxiliary information z, which in this paper we will assume to be a random string of polynomial size $\mathsf{poly}(\lambda)$. However, since we are dealing with obfuscation, we will consider samplers where no messages are produced. We recall the unpredictability experiment for this special case in Figure 12, where notation $\langle \boldsymbol{a}, \boldsymbol{x} \rangle$ denotes the vector that results from computing inner products between $\boldsymbol{a}$ and all the elements in $\boldsymbol{x}$.

```
$\mathrm{Pred}^{\mathcal{P}}_{\mathcal{S}}(1^\lambda)$:
$st \leftarrow\$ \mathcal{P}_1(1^\lambda)$
$z \leftarrow\$ \{0,1\}^{\mathsf{poly}(\lambda)}$
$(\mathbf{a}_0, \mathbf{a}_1) \leftarrow\$ \mathcal{S}(1^\lambda, z, st)$
$\mathbf{x} \leftarrow\$ \mathcal{P}_2^{\textsc{Func}}(1^\lambda, z, st)$
return $(\langle \mathbf{a}_0, \mathbf{x} \rangle \neq \langle \mathbf{a}_1, \mathbf{x} \rangle)$

$\textsc{Func}(\mathbf{x})$:
return $(\langle \mathbf{a}_0, \mathbf{x} \rangle)$
```

Fig. 12. Game defining single-instance unpredictability of a sampler $\mathcal{S}$ against $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2)$.

Computational Assumption. Our computational assumption is a vectorized version of the DDH variant introduced in [23], in the style of the assumption that is used in [14] to establish the DI property of a point function obfuscator. The assumption states that, for every unpredictable sampler, any distinguishing adversary has a negligible advantage in the game in Figure 13. We note that the unpredictability restriction on the sampler essentially excludes any challenge where a polynomial-size set of black-box linear tests (i.e., inner product computations) could be used by a semi-bounded predictor to distinguish the hidden bit. This is a natural restriction, since the adversary is given enough information to trivially perform such tests on its own. The assumption therefore states that ppt adversaries cannot do better than what can be achieved with such linear tests. In particular, we note that such linear tests can be used to extract coefficient equality patterns that might permit trivial distinguishing attacks by checking group element repetitions in the received obfuscations.

```
$\mathsf{Assumption}_{\mathcal{S},\mathcal{A},\mathcal{G},t,d,\mathsf{poly}}(1^\lambda)$:
$b \leftarrow_{\$} \{0,1\}$; $z \leftarrow_{\$} \{0,1\}^{\mathsf{poly}(\lambda)}$
$(\mathbf{a}_0, \mathbf{a}_1) \leftarrow_{\$} \mathcal{S}(z, \epsilon)$
$\boldsymbol{g} \leftarrow_{\$} \mathcal{G}_\lambda^t$
$M_b \leftarrow \begin{bmatrix} \boldsymbol{g}[1]^{\mathbf{a}_b[1][1]} & \dots & \boldsymbol{g}[1]^{\mathbf{a}_b[1][d]} \\ \dots & \dots & \dots \\ \boldsymbol{g}[t]^{\mathbf{a}_b[t][1]} & \dots & \boldsymbol{g}[t]^{\mathbf{a}_b[t][d]} \end{bmatrix}$
$b' \leftarrow_{\$} \mathcal{A}(M_b, z)$
return $b = b'$
```

Fig. 13. Game defining a DDH-style computational assumption.

DI obfuscation for hyperplane membership. The following theorem can be trivially proven using a direct reduction.

Theorem 5. *The hyperplane membership obfuscator of Canetti, Rothblum and Varia [23] is DI secure in the presence of random auxiliary information if the assumption in Figure 13 holds in $\mathcal{G}$. More precisely, for every unpredictable hyperplane membership sampler $\mathcal{S}$, any DI adversary $\mathcal{A}$ that breaks the DI property can be used (without change) to break the underlying assumption with the same advantage.*