Dual-Mode NIZKs from Obfuscation

Dennis Hofheinz[1], Bogdan Ursu[1]

Karlsruhe Institute of Technology
{dennis.hofheinz,bogdan.ursu}@kit.edu

Abstract. Two standard security properties of a non-interactive zero-knowledge (NIZK) scheme are soundness and zero-knowledge. But while standard NIZK systems can only provide one of those properties against unbounded adversaries, *dual-mode NIZK systems* allow to choose dynamically and adaptively which of these properties holds unconditionally. The only known dual-mode NIZK schemes are Groth-Sahai proofs (which have proved extremely useful in a variety of applications), and the FHE-based NIZK constructions of Canetti et al. and Peikert et al, which are concurrent and independent to this work. However, all these constructions rely on specific algebraic settings.

Here, we provide a generic construction of dual-mode NIZK systems for all of NP. The public parameters of our scheme can be set up in one of two indistinguishable ways. One way provides unconditional soundness, while the other provides unconditional zero-knowledge. Our scheme relies on subexponentially secure indistinguishability obfuscation and subexponentially secure one-way functions, but otherwise only on comparatively mild and generic computational assumptions. These generic assumptions can be instantiated under any one of the DDH, k-LIN, DCR, or QR assumptions.

As an application, we reduce the required assumptions necessary for several recent obfuscation-based constructions of multilinear maps. Combined with previous work, our scheme can be used to construct multilinear maps from obfuscation and a group in which the strong Diffie-Hellman assumption holds. We also believe that our work adds to the understanding of the construction of NIZK systems, as it provides a conceptually new way to achieve dual-mode properties.

Keywords: non-interactive zero-knowledge, dual-mode proof systems, indistinguishability obfuscation.

1 Introduction

Obfuscation and structured assumptions. Indistinguishability obfuscation (iO) is a powerful cryptographic object, and along with one-way functions, it implies almost every cryptographic primitive, from deniable encryption [42] to functional encryption [26] and fully-homomorphic encryption [18]. However, it is not currently known whether iO gives rise to structures in which algebraic assumptions hold (such as DDH, DCR, LWE etc.). In this work, we are motivated by the following open problem:

Can structured objects (such as DDH groups) be bootstrapped from unstructured objects (like generic one-way functions and iO)?

We make progress in this direction by developing the first construction of dual-mode non-interactive zero-knowledge (NIZK) proof systems from unstructured assumptions (iO, one-way functions and lossy trapdoor functions). This dual-mode NIZK can be used in the constructions from [1,2,21], allowing us to answer this question in the affirmative.

Zero-knowledge proof systems. Zero-knowledge (ZK) proof systems [28,29] are (implicitly or explicitly) at the heart of countless cryptographic constructions. In a ZK proof system, a prover P tries to convince a verifier V of the validity of a statement x. "Validity" usually means that $x \in L$ for some language $L \in \mathsf{NP}$. In this case, P obtains a witness w to $x \in L$. For security, we require *soundness*, which means that no dishonest prover can convince V of a false statement $x \notin L$. Additionally, we may want to protect P (and in particular the used witness w) in several ways. For instance, the protocol is *zero-knowledge* if it is possible to efficiently simulate (transcripts of) protocol runs even without w. Alternatively, we can require the protocol to be *witness-hiding* or *witness-indistinguishable* [23].

ZK proof systems can be interactive or non-interactive (the latter of which means that the prover sends only one message to the verifier). In this work, we are interested in non-interactive ZK (NIZK) proof systems [10]. There exist already various NIZK proof systems, ranging from generic [22,24,42] to highly efficient constructions based on concrete number-theoretic assumptions [24, 32, 44]. Some of these systems only allow to prove $x \in L$ for specific languages L, while others can be used to prove statements from arbitrary languages $L \in \mathsf{NP}$.

Dual-mode proof systems. Some NIZK systems enjoy statistical security, i.e., information-theoretic soundness or zero-knowledge guarantees. However, interestingly, no NIZK system can be statistically sound *and* statistically zero-knowledge simultaneously. Hence, a NIZK system can be secure only *either* against unbounded malicious provers *or* against unbounded malicious verifiers.

Fortunately, there is a compromise that combines the best of both worlds: Groth-Sahai proofs [32] are statistically sound or statistically zero-knowledge depending on the choice of public parameters crs. Furthermore, both choices of parameters are computationally indistinguishable. This "dual-mode" property leads to comparatively simple proofs for complex protocols (e.g., for anonymous credentials [4] or payment systems [33]). In the case of [2, 21], a proof without using dual-mode properties in fact does not seem obvious at all.[1]

Until recently, only Groth-Sahai proofs [32] (and their variants, e.g., [9, 20, 35]) were known to possess this dual-mode property.[2] These proof systems all rely on a very specific and structured algebraic setting (pairing-friendly cyclic groups). In contrast, we rely on generic rather than algebraic techniques, resulting in a fundamentally new way of obtaining dual-mode proof systems.

Concurrent work Concurrently and independently to this work, [19, 39] have put forward breakthrough approaches to obtain dual-mode NIZKs from the LWE assumption. These constructions rely on rich algebraic structures and are non-blackbox. In contrast, our techniques are generic and our perspective is closer to computational complexity, in that we investigate whether the existence of a powerful non-algebraic object (iO) can lead to algebraic ones.

Our contribution In this paper, we give the first generic construction of dual-mode NIZK proofs from (the combination of) the following ingredients:

- subexponentially secure indistinguishability obfuscation (iO, [3, 26]),
- subexponentially secure one-way functions,
- a (selectively) subexponentially secure functional encryption scheme,
- lossy encryption [5, 40], and
- lossy functions (LFs), a relaxation of lossy trapdoor functions [41] which we introduce in this paper.

We stress that some of our ingredients are implied by (a combination of) others: Functional encryption can be constructed from iO and one-way functions [26]. Conversely, subexponentially secure functional encryption implies subexponentially secure iO and one-way functions (e.g., [8] and the references therein). Furthermore, both LFs and lossy encryption are implied by lossy trapdoor functions [41].

As a side note, we remark that thus, a subexponential variant of any of the DDH, k-LIN, QR, DCR, or LWE assumptions, along with subexponential iO implies all of our ingredients.[3]

Of course, since we assume iO, our construction is far from practical. Still, it has interesting theoretical applications. For instance, it allows to instantiate dual-mode NIZK proofs in the recent works [1, 2, 21] without any additional assumptions, and in particular without pairing-friendly groups. (Incidentally, these works already assume what we need for our construction.)

[1] A bit more technically, dual-mode NIZK proofs allow to use both witness extraction or simulation trapdoors in different stages of the proof, depending on the chosen mode. (This is helpful in case of [4,33] and crucial in [2,21].) Furthermore, in complex settings with mutually dependent statements and witnesses, statistical properties are easier seen to compose.

[2] We do not consider NIZK proofs in the random oracle model (such as [37]) here.

[3] See [11, 25, 41] for the corresponding instantiations of lossy trapdoor functions from these concrete assumptions.

In particular, combining our results with the scheme from [1], shows that it is possible to obtain a very structured object (namely, a cyclic group in which Diffie-Hellman and similar assumptions hold) solely from an unstructured and generic object (iO), and a mildly structured object (a lossy trapdoor function).[4]

Similarly, implementing [2, 21] with our system (instead of with Groth-Sahai proofs) yields a pairing-friendly group (with non-unique representation) from iO and a DDH group (both subexponentially secure). Therefore, we also give an answer to the following open problem:

Can bilinear groups be bootstrapped from DDH groups and iO?

Previous work	This work + [1, 2, 21]
[2] iO + Pairings + SDDH ⇒ Multilinear Maps	iO + SDDH ⇒ Multilinear Maps
[21] iO + Pairings + SDDH ⇒ Graded Encoding Schemes	iO + SDDH ⇒ Graded Encoding Schemes
[1] iO + Pairings ⇒ Interactively Secure Groups	iO + LTDF ⇒ Interactively Secure Groups

Fig. 1. Some implications on previous results. "iO", "LTDF" and "SDDH" denote subexponential versions of indistinguishability obfuscation, lossy trapdoor functions and the "Strong DDH" (a q-type variant of the Diffie-Hellman assumption).

Open problems. We note that the groups from [1, 2, 21] all enjoy non-unique representations of group elements. That is, equality of group elements can be tested, but there does not exist a canonical form. Removing this limitation remains an open problem.

Our techniques

Existing generic approaches. Before explaining our main ideas, we first mention that generic constructions of NIZKs from iO already exist. Namely, [42] present a NIZK construction that only assumes iO and one-way functions. Their construction is (even perfectly) zero-knowledge. However, proofs are in their case simply signatures of the corresponding statement x. Thus, their construction is inherently limited to computational soundness, in the sense that it is not clear how to tweak this construction to obtain statistical soundness.

Secondly, it is possible to construct a notion of trapdoor permutations from iO that is in turn sufficient to construct statistically sound NIZK proofs [17] (cf. [6, 7, 22, 30]). However, it is not clear how to tweak this NIZK construction to obtain statistical zero-knowledge.

The hidden bits model. Similarly to [17], our starting point is also the generic NIZK construction from [22]. This work presents a statistically sound and perfectly zero-knowledge NIZK protocol in an ideal model of computation called the "hidden bits model" (HBM).[5] It will be helpful to first recall the HBM before going further. In a nutshell, the HBM gives the prover P access to an ideal random bitstring $\mathsf{hrs} = (\mathsf{hrs}_1, \ldots, \mathsf{hrs}_t) \in \{0, 1\}^t$. Next, P selects a subset $\mathcal{I} \subseteq [t]$ and a proof π. Then, the verifier V is activated with $\mathcal{I}, \pi$, the subset $(\mathsf{hrs}_i)_{i \in \mathcal{I}}$ of hrs that is selected by $\mathcal{I}$, and of course the instance x. Finally, V is supposed to output a verdict $b \in \{0, 1\}$.

Two ways to implement the HBM. Note that the power of the HBM stems from the fact that hrs is ideally random (and cannot be tampered with by P), but only revealed in part to V. When implementing the HBM, we will necessarily have to compromise on some of these properties. However, it will be interesting to see what the consequences of such compromises are. Specifically, when implementing the HBM in the HBM-based NIZK protocol of [22], we can observe the following:

[4] Indeed, except for a dual-mode NIZK proof system, all assumptions in [1] can be instantiated from subexponentially secure iO and a subexponentially secure lossy trapdoor function. We note, however, that [1] construct a group in which elements have only a non-unique representation and no canonical form. Hence, their group might not be considered a "standard group", but still has a rich algebraic structure.

[5] Since their protocol is formulated in an ideal model of computation, it does not contradict our remark above about the impossibility of simultaneously achieving statistical soundness and statistical zero-knowledge. One of the two statistical properties will be lost when implementing this ideal model.

(a) if we implement the HBM such that hrs is truly random (or selected from a small set of possible hrs values, each of which is individually truly random), then the resulting NIZK protocol is statistically sound and computationally zero-knowledge,

(b) if we implement the HBM such that the unopened bits $(\mathsf{hrs}_i)_{i \notin \mathcal{I}}$ are statistically hidden from V, then the resulting NIZK protocol is statistically zero-knowledge and computationally sound.

Known implementations of the HBM (e.g., [22,30,31]) follow (a), and thus enjoy statistical soundness guarantees. Our main strategy will be to build a dual-mode NIZK proof system by implementing the HBM in a way that allows to switch (by switching public parameters) between (a) and (b).

A first approach. Our first step will be to set up the hidden string hrs as

$$\mathsf{hrs} = \mathsf{H}(X) \oplus \mathsf{crs}$$

for a value X chosen freely by P, a yet-to-be-defined function H, and a truly random "randomizing string" crs fixed in the public parameters. If H is a pseudorandom generator (that admits a suitable partial opening process, see [31] for an explicit formulation), this yields the core of existing HBM implementations. In particular, if H has a small image, then we are in case (a) above, and the resulting NIZK is statistically sound.

However, suppose we can switch (in a computationally indistinguishable way) $\mathsf{H}(X)$ to have a large image, such that in fact $\mathsf{H}(X) \in \{0,1\}^t$ is close to uniformly distributed for random X. We call such a "switchable" object a lossy function (LF). An LF can be easily constructed, e.g., by universally hashing the output of a lossy trapdoor function F. For suitable choices of parameters, $\mathsf{H}(X) := h(F(X))$ is close to uniform if F is injective (and X random), and has a small range if F does.

With $\mathsf{H}(X)$ close to uniform, we are in case (b) above, assuming that the process itself of revealing $\mathsf{hrs}_{\mathcal{I}}$ does not reveal additional information about other bit positions. Hence, we obtain a statistically zero-knowledge NIZK protocol, and in summary even a dual-mode NIZK that can be switched between statistically sound and statistically zero-knowledge modes of operation.

Managing the opening process. The main problem with our first approach is that it is not clear how to *partially* open a subset $\mathsf{hrs}_{\mathcal{I}}$ of hrs to a verifier V. Previous HBM implementations (e.g., [22,31]) devised elaborate ways to partially open suitably designed pseudorandom generators (in the role of H above). We cannot use those techniques for two reasons. First, their opening process might reveal statistical information about the unopened parts of hrs. Second, these techniques require specific H functions, and do not appear to work with "switchable" functions H as we need. Hence, we use the strong ingredients mentioned above to design our own opening process.

We will use a functional encryption scheme FE. We will publicize a truly random crs, a statement Z from a language L' that is hard to decide, along with an FE public key fmpk, and a corresponding secret key $\mathsf{sk_f}$ for the following function f:

$$\mathsf{f}(X, \mathcal{I}, z, T) := \begin{cases} (T, \mathcal{I}) & \text{if } z \text{ is a witness to } Z \in L' \\ (\mathsf{H}(X)_{\mathcal{I}}, \mathcal{I}) & \text{else.} \end{cases}$$

An opening consists of an encryption

$$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (X, \mathcal{I}, 0, 0))$$

that will decrypt to $\mathsf{f}(X, \mathcal{I}, 0, 0) = \mathsf{H}(X)_{\mathcal{I}}$ under $\mathsf{sk_f}$. The verifier will receive this opening, retrieve $\mathsf{H}(X)_{\mathcal{I}}$ with $\mathsf{sk_f}$, and compute $\mathsf{hrs}_{\mathcal{I}} = \mathsf{H}(X)_{\mathcal{I}} \oplus \mathsf{crs}_{\mathcal{I}}$.

Observe that this process has the following properties:

– If $Z \notin L'$, then $\mathsf{sk_f}(C) = (\mathsf{H}(X)_{\mathcal{I}}, \mathcal{I})$ always. Hence, if additionally H has a small range, we are in case (a) above, and the corresponding NIZK protocol is statistically sound.

– If $Z \in L'$ with witness z, then any prover who knows z can efficiently open $\mathsf{hrs}_{\mathcal{I}}$ arbitrarily, by encrypting $(0, \mathcal{I}, z, T)$ for $T = \mathsf{crs}_{\mathcal{I}} \oplus \mathsf{hrs}_{\mathcal{I}}$ and the desired $\mathsf{hrs}_{\mathcal{I}}$. Furthermore, such openings obviously do not contain any information about potential other positions of hrs. This means we are in case (b) above, and the corresponding NIZK protocol is statistically zero-knowledge.

By using FE's security, it is possible to show that these two types of openings are indistinguishable to a verifier. However, as formulated, they are of course not indistinguishable to a prover yet. Hence, we will additionally publicize an obfuscated algorithm PC that will get as input a statement x with witness w, and random coins r. Depending on the mode (sound or zero-knowledge), $\mathsf{PC}(x, w, r)$ will then either encrypt $(X, \mathcal{I}, 0, 0)$ or $(0, \mathcal{I}, z, T)$, for pseudorandom X and T derived from r.

A taste of the security proof. For security, we will show that the public parameters in both modes are computationally indistinguishable. The security proof is somewhat technical, and we would like to highlight only one interesting theme here. Namely, observe that the prover algorithm PC is inherently probabilistic. In the proof, we need to modify PC's behavior, and in particular decouple its output distribution from its input w. Specifically, when aiming at statistical soundness, the output of PC will encrypt, and thus depend on w. But when trying to achieve zero-knowledge, PC's output should not reveal (in a statistical sense) which witness w has been used.[6]

This decoupling process is particularly cumbersome to go through because PC itself is public and can be run on arbitrary inputs. Any change that essentially makes PC ignore its w input will be easily detectable. Hence, we add an indirection that helps to remove dependencies on w. Specifically, we let PC first compute $a = \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$ using a lossy encryption scheme LE. If the corresponding public key lpk is injective (i.e., leads to decryptable ciphertexts), then a determines w. Hence, any case distinction (or hybrid argument) we make for different values of w can alternatively be made for different values of a. On the other hand, if lpk is lossy, then a will be statistically independent of the plaintext (x, w).

Hence, a can be used as a single value that (a) can serve as a "fingerprint" of (or in some sense even as a substitute for) w in the proof, but (b) can be easily made independent of w by switching lpk into lossy mode. Equipped with this gadget, we will structure the proof as a large hybrid argument over all values of a (encrypted at this point with an injective lpk). In each step, we modify PC's behavior for one particular value of (x, w), and change the corresponding FE ciphertext C from an encryption of $(X, \mathcal{I}, 0, 0)$ to $(0, \mathcal{I}, z, T)$ for a pseudorandom value T derived from a.

Roadmap. After recalling some preliminaries in Sec. 2, we present our proof system in Sec. 3, followed by its analysis in Sec. 4. The appendix contains a schematic overview over our main proof (App. A), a proof of a technical lemma (App. B), a recap of the HBM-based NIZK from [22] (App. C), and an analysis of the (statistical) extractability of our scheme (App. D).

2 Preliminaries

Notation. Throughout this paper, λ denotes the security parameter. For a natural number $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\}$. A probabilistic polynomial time algorithm (PPT, also denoted *efficient* algorithm) runs in time polynomial in the (implicit) security parameter λ. A positive function f is *negligible* if for any polynomial p there exists a bound $B > 0$ such that, for any integer $k \geq B$, $f(k) \leq 1/|p(k)|$. An event depending on λ occurs with *overwhelming probability* when its probability is at least $1 - \mathsf{negl}(\lambda)$ for a negligible function negl. Given a finite set S, the notation $x \leftarrow_{\mathrm{R}} S$ means a uniformly random assignment of an element of S to the variable x. If A is a probabilistic algorithm, $y \leftarrow_{\mathrm{R}} A(\cdot)$ denotes the process of running A on some appropriate input and assigning its output to y. The notation $\mathcal{A}^{\mathcal{O}}$ indicates that the algorithm $\mathcal{A}$ is given oracle access to $\mathcal{O}$. We denote $a \leftarrow A; b \leftarrow B(a); \ldots$ for running the experiment where a is chosen from A, after which b is chosen from B, which might depend on a and so on. This determines a probability distribution over the outputs and we write $\Pr[a \leftarrow A; b \leftarrow B(a); \ldots : C(a, b, \ldots)]$ for the probability of the condition $C(a, b, \ldots)$ being satisfied after running the experiment. For two distributions D_1, D_2, we denote by $\Delta(D_1, D_2)$ the statistical distance. We also write $D_1 \equiv D_2$ when the distributions are identical, $D_1 \approx_{\mathrm{C}} D_2$ when the distributions are computationally indistinguishable and $D_1 \approx_{\epsilon} D_2$ when $\Delta(D_1, D_2) \leq \epsilon$.

[6] Formally, to achieve zero-knowledge, we must achieve witness-indistinguishability.

2.1 Puncturable Pseudorandom Function

A pseudorandom function (PRF) originally introduced in [27], is a tuple of PPT algorithms $\mathsf{PRF} = (\mathsf{PRF.KeyGen}, \mathsf{PRF.Eval})$. Let $\mathcal{K}$ denote the key space, $\mathcal{X}$ denote the domain, and $\mathcal{Y}$ denote the range. The key generation algorithm $\mathsf{PRF.KeyGen}$ on input of 1^λ, outputs a random key from $\mathcal{K}$ and the evaluation algorithm $\mathsf{PRF.Eval}$ on input of a key K and $x \in \mathcal{X}$, evaluates the function $F \colon \mathcal{K} \times \mathcal{X} \mapsto \mathcal{Y}$. The core property of PRFs is that, on a random choice of key K, no probabilistic polynomial-time adversary should be able to distinguish $F(K, \cdot)$ from a truly random function, when given black-box access to it. Puncturable PRFs (pPRFs) have the additional property that some keys can be generated *punctured* at some point, so that they allow to evaluate the PRF at all points except for the punctured point. As observed in [13, 14, 36], it is possible to construct such punctured keys for the original construction from [27], which can be based on any one-way functions [34].

Definition 1 (Puncturable Pseudorandom Function [13, 14, 36]). *A* puncturable pseudorandom function (pPRF) *is with punctured key space $\mathcal{K}_p$ is a triple of PPT algorithms* $(\mathsf{PRF.KeyGen}, \mathsf{PRF.Puncture}, \mathsf{PRF.Eval})$ *such that:*

- $\mathsf{PRF.KeyGen}(1^\lambda)$ *outputs a random key $K \in \mathcal{K}$,*
- $\mathsf{PRF.Puncture}(K, x)$*, on input $K \in \mathcal{K}$, $x \in \mathcal{X}$, outputs a punctured key $K\{x\} \in \mathcal{K}_p$,*
- $\mathsf{PRF.Eval}(K', x')$*, on input a key K' (punctured or not), and a point x', outputs an evaluation of the PRF.*

We require PRF *to meet the following conditions:*

Functionality preserved under puncturing. *For all $\lambda \in \mathbb{N}$, for all $x \in \mathcal{X}$,*

$$\begin{aligned}&\Pr[K \leftarrow_{\scriptscriptstyle \mathrm{R}} \mathsf{PRF.KeyGen}(1^\lambda), K\{x\} \leftarrow_{\scriptscriptstyle \mathrm{R}} \mathsf{PRF.Puncture}(K, x) \colon \\ &\quad \forall x' \in \mathcal{X} \setminus \{x\} \colon \mathsf{PRF.Eval}(K, x') = \mathsf{PRF.Eval}(K\{x\}, x')] = 1.\end{aligned}$$

Pseudorandom at punctured points. *For every stateful* PPT *adversary $\mathcal{A}$ and every security parameter $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ in* $\mathsf{Exp\text{-}s\text{-}pPRF}$ *(described in Figure 2) is negligible, namely:*

$$\mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\lambda, \mathcal{A}) := \left| \Pr[\mathsf{Exp\text{-}s\text{-}pPRF}(1^\lambda, \mathcal{A}) = 1] - \tfrac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

```
Experiment $\mathsf{Exp\text{-}s\text{-}pPRF}(1^\lambda, \mathcal{A})$
Experiment $\mathsf{Exp\text{-}s\text{-}pPRF}_{\mathcal{A}}(\lambda)$
$x^* \leftarrow_{\mathrm{R}} \mathcal{A}(1^\lambda)$, $b \leftarrow_{\mathrm{R}} \{0,1\}$,
$K \leftarrow_{\mathrm{R}} \mathsf{PRF.KeyGen}(1^\lambda)$,
$K\{x^*\} \leftarrow_{\mathrm{R}} \mathsf{PRF.Puncture}(K, x^*)$,
$y_0 \leftarrow \mathsf{PRF.Eval}(K, x^*)$, $y_1 \leftarrow_{\mathrm{R}} \mathcal{Y}$
$b' \leftarrow_{\mathrm{R}} \mathcal{A}(K\{x^*\}, y_b)$
Return $b = b'$
```

Fig. 2. Experiment $\mathsf{Exp\text{-}s\text{-}pPRF}_{\mathcal{A}}(\lambda)$ for the pseudo-randomness at punctured points.

Sub-exponential security We say that PRF is sub-exponentially secure when it satisfies Definition 1 and in addition it satisfies: for every PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\lambda, \mathcal{A}) \leq \frac{1}{2^{\lambda^\epsilon}}$, for some positive constant $0 < \epsilon < 1$.

Definition 1 corresponds to a selective security notion for puncturable pseudorandom functions; adaptive security could be considered, but will not be required in our work. For ease of notation we often write $F(\cdot, \cdot)$ instead of $\mathsf{PRF.Eval}(\cdot, \cdot)$.

2.2 Lossy functions

We generalize the notion of LTDF (lossy trapdoor function) due to [41] and introduce lossy functions. LTDFs (Lossy trapdoor functions) can be sampled in two indistinguishable modes: an injective and a lossy mode.

When sampling injective functions, the setup also provides a trapdoor which can be used to invert the function. Unlike LTDFs, for lossy functions we require that functions can be sampled in two modes, but in which one mode is "more lossy" than the other. Thus, instead of an injective and a lossy mode, we have a "less lossy" and a "more lossy" mode, which we denote as "dense" and "lossy" modes. Since we do not necessarily have injectivity in the dense setting, we also do not have trapdoors as in LTDFs.

Definition 2 (Lossy Functions). *A tuple* $\mathsf{LF} = (\mathsf{Setup}, \mathsf{Eval})$ *of* PPT *algorithms is a family of* (n, k, m, i)*-lossy functions if the following properties hold:*

- *Sampling functions: Both* $\mathsf{Setup}(1^\lambda, \text{dense})$ *of dense functions and* $\mathsf{Setup}(1^\lambda, \text{lossy})$ *of lossy functions output a function index* s*. We require that* $\mathsf{Eval}(s, \cdot)$ *is a deterministic function on* $\{0,1\}^n \to \{0,1\}^m$ *in both cases. In the following, we use the shorthand notation* $s(\cdot) \coloneqq \mathsf{Eval}(s, \cdot)$*.*
- *Dense functions have images statistically close to uniformly random: for all* $s \leftarrow_{\mathrm{R}} \mathsf{LF}(1^\lambda, \text{dense})$*, we have that:*
$$\Delta((s(U_n), s), (U_m, s)) \leq \tfrac{1}{2^i}.$$
- *Lossy functions have small image size: The image size of lossy functions is bounded by* 2^k*. In particular, for all* $s \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{lossy})$*,*
$$|\{\mathsf{Eval}(s, x) : x \in \{0,1\}^n\}| \leq 2^k.$$
- *Indistinguishability: The outputs of* $\mathsf{Setup}(1^\lambda, \text{lossy})$ *and* $\mathsf{Setup}(1^\lambda, \text{dense})$ *are computationally indistinguishable, i.e.* $\{\mathsf{Setup}(1^\lambda, \text{lossy})\} \approx_{\mathrm{C}} \{\mathsf{Setup}(1^\lambda, \text{dense})\}$

We can generalise Definition 2 even further. Instead of asking that in dense mode the evaluation of the function is statistically close to a uniformly random, we may instead define the dense mode as having $H_\infty(\mathsf{Eval}(s, U_n)) \geq m + 2\log\left(\frac{1}{\epsilon}\right)$. Then, by the leftover hash lemma, we can combine LF with a 2-universal hash function to ensure that the output is statistically close to uniformly random as in Definition 2. For clarity, we do not use this generalization in our proofs.

Concrete instantiations: The lossy trapdoor functions from [41] are also lossy functions in the sense of Definition 2. Moreover, composed with 2-universal hash functions, they satisfy the necessary parameters in our construction (see Section 3). This would yield suitable lossy functions based on DDH and LWE.

2.3 Lossy Encryption

Definition 3. [5, 40]*: A lossy public-key encryption scheme is a tuple* $\mathsf{LE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *of polynomial-time algorithms such that*

- $\mathsf{Gen}(1^\lambda, \text{inj})$ *outputs keys* $(\mathsf{pk}, \mathsf{sk})$*, keys generated by* $\mathsf{Gen}(1^\lambda, \text{inj})$ *are called injective keys.*
- $\mathsf{Gen}(1^\lambda, \text{lossy})$ *outputs keys* $(\mathsf{pk}_{\text{lossy}}, \perp)$*, keys generated by* $\mathsf{Gen}(1^\lambda, \text{lossy})$ *are called lossy keys.*
- $\mathsf{Enc}(\mathsf{pk}, \cdot, \cdot) : M \times R \to C$*.*

Additionally, the algorithms must satisfy the following properties:

1. *Correctness on injective keys. For all plaintexts* $x \in X$*,*
$$\Pr[(\mathsf{pk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Gen}(1^\lambda, \text{inj}); r \leftarrow R : \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, x, r)) = x] = 1.$$
2. *Indistinguishability of keys. Public keys* pk *are computationally indistinguishable in lossy and injective modes. Specifically, if* $\text{proj} : (\mathsf{pk}, \mathsf{sk}) \to \mathsf{pk}$ *is the projection map, then:*
$$\{\text{proj}(\mathsf{Gen}(1^\lambda, \text{inj}))\} \approx_{\mathrm{C}} \{\text{proj}(\mathsf{Gen}(1^\lambda, \text{lossy}))\}.$$
3. *Lossiness of lossy keys. For all* $(\mathsf{pk}_{\text{lossy}}, \perp) \leftarrow_{\mathrm{R}} \mathsf{Gen}(1^\lambda, \text{lossy})$*, and all* $x_0, x_1 \in M$*, the two distributions* $\{r \leftarrow_{\mathrm{R}} R : (\mathsf{pk}_{\text{lossy}}, \mathsf{Enc}(\mathsf{pk}_{\text{lossy}}, x_0, r))\}$ *and* $\{r \leftarrow_{\mathrm{R}} R : (\mathsf{pk}_{\text{lossy}}, \mathsf{Enc}(pk_{\text{lossy}}, x_1, r))\}$ *are statistically close, i.e. the statistical distance is negligible in* λ*.*

We define a lossy encryption scheme LE *to be* μ*-lossy if for all* $(\mathsf{pk}_{\text{lossy}}, \perp) \leftarrow_{\mathrm{R}} \mathsf{Gen}(1^\lambda, \text{lossy})$ *and for all* x_0, x_1*, we have that:*

$$\{r \leftarrow_{\mathrm{R}} R : (\mathsf{pk}_{\text{lossy}}, \mathsf{Enc}(\mathsf{pk}_{\text{lossy}}, x_0, r))\} \approx_\mu \{r \leftarrow_{\mathrm{R}} R : (\mathsf{pk}_{\text{lossy}}, \mathsf{Enc}(pk_{\text{lossy}}, x_1, r))\}$$

2.4 Functional Encryption

Definition 4. [12,38,43] *A functional encryption scheme for a class of functions $\mathcal{F} = \mathcal{F}(1^\lambda)$ over message space $\mathcal{M} = \mathcal{M}_\lambda$ consists of four polynomial time algorithms $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:*

1. $\mathsf{Setup}(1^\lambda)$ *– on input the security parameter λ outputs master public key* mpk *and master secret key* msk*.*
2. $\mathsf{KeyGen}(\mathsf{msk}, f)$ *– on input the master secret key* msk *and a description of function $f \in \mathcal{F}$ and outputs a corresponding secret key* sk_f*.*
3. $\mathsf{Enc}(\mathsf{mpk}, x)$ *– on input the master public key* mpk *and a string x, outputs a ciphertext* ct*.*
4. $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$ *– on inputs the secret key* sk_f *and a ciphertext encrypting message $m \in M$, outputs $f(m)$.*

A functional encryption scheme is perfectly correct for $\mathcal{F}$ if for all $f \in \mathcal{F}$ and all messages $m \in \mathcal{M}$:

$$\Pr[(\mathsf{mpk}, \mathsf{msk}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda) : \mathsf{Dec}(\mathsf{KeyGen}(\mathsf{msk}, f), \mathsf{Enc}(\mathsf{mpk}, m)) = f(m)] = 1$$

In addition, for the proof of theorem 14, we need a stronger property from the functional encryption schemes we use in our construction, which we call special correctness of decryption keys. Special correctness requires that decrypting any (potentially maliciously generated) ciphertext under the decryption key sk_f yields a result which lies in the range of the function f. The functional encryption scheme based on iO and one-way functions from [26] satisfies this property.

Definition 5 (Special correctness of decryption keys). *A functional encryption scheme satisfies special correctness if for all $\lambda \in \mathbb{N}$, for all* ct*, for all $f \in \mathcal{F}$, there exists $m \in \mathcal{M}$, such that:*

$$\Pr\left[\begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda), \\ \mathsf{sk}_f \leftarrow_{\mathrm{R}} \mathsf{KeyGen}(\mathsf{msk}, f)\end{array} : \mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct}) \in \{f(m), \bot\}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Definition 6 (Selectively Indistinguishable Security). *A functional encryption scheme* FE *is selectively indistinguishable secure (* $\mathsf{SEL\text{-}IND\text{-}FE\text{-}CPA}$*) if for all stateful* PPT *adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the experiment* $\mathsf{Exp\text{-}}s\mathsf{\text{-}IND\text{-}FE\text{-}CPA}$ *described in Figure 3 is negligible, namely:*

$$\mathsf{Adv}^{\mathsf{FE}}_{\mathsf{Exp}\text{-}s\text{-}\mathsf{IND\text{-}FE\text{-}CPA}}(\lambda, \mathcal{A}) := \left|\Pr[\mathsf{Exp}\text{-}s\text{-}\mathsf{IND\text{-}FE\text{-}CPA}^{\mathsf{FE}}(1^\lambda, \mathcal{A}) = 1] - \tfrac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

```
Experiment $\mathsf{Exp}\text{-}s\text{-}\mathsf{IND\text{-}FE\text{-}CPA}^{\mathsf{FE}}(1^\lambda, \mathcal{A})$
$(m_0, m_1) \leftarrow_{\mathrm{R}} \mathcal{A}(1^\lambda)$;
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow_{\mathrm{R}} \mathsf{FE.Setup}(1^\lambda)$
$b \leftarrow_{\mathrm{R}} \{0, 1\}$
$\mathsf{ct} \leftarrow_{\mathrm{R}} \mathsf{FE.Enc}(\mathsf{mpk}, m_b)$
$b' \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{FE.KeyGen}(\mathsf{msk}, \cdot)}(\mathsf{mpk}, \mathsf{ct})$
Return $b = b'$
```

Fig. 3. Experiment $\mathsf{Exp}\text{-}s\text{-}\mathsf{IND\text{-}FE\text{-}CPA}$ for the selective indistinguishable security of FE. The queries of $\mathcal{A}$ to oracle $\mathsf{FE.KeyGen}(\mathsf{msk}, \cdot)$ are restricted to functions f such that $f(m_0) = f(m_1)$.

Definition 7 (Sub-exponential Selectively Indistinguishability Security). *A functional encryption scheme* FE *is sub-exponentially selectively indistinguishability secure if it satisfies Definition 6 and in addition: for all* PPT *adversaries $\mathcal{A}$:*

$$\mathsf{Adv}^{\mathsf{FE}}_{\mathsf{Exp}\text{-}s\text{-}\mathsf{IND\text{-}FE\text{-}CPA}}(\lambda, \mathcal{A}) \leq \tfrac{1}{2^{\lambda^\epsilon}}, \textit{ for some positive constant } 0 < \epsilon < 1.$$

2.5 Indistinguishability Obfuscation

Definition 8 ([3, 26] Indistinguishability Obfuscator). *A uniform* PPT *machine* iO *is called an indistinguishability obfuscator for a circuit class $\mathcal{C}_\lambda$ if the following conditions are satisfied:*

- *For all security parameters $\lambda \in \mathbb{N}$, for all $C \in \mathcal{C}_\lambda$, for all inputs x, we have:*

$$Pr[C'(x) = C(x) : C' \leftarrow_{\mathrm{R}} \mathsf{iO}(\lambda, C)] = 1$$

- *For any (not necessarily uniform)* PPT *distinguisher $\mathcal{A}$, for all security parameters $\lambda \in \mathbb{N}$, for all pairs of circuits $C_0, C_1 \in \mathcal{C}_\lambda$, we have that if $C_0(x) = C_1(x)$ for all inputs x, then:*

$$\mathsf{Adv}^{\mathsf{iO}}(\lambda, \mathcal{A}) := |\Pr[\mathcal{A}(\mathsf{iO}(\lambda, C_0)) = 1] - \Pr[\mathcal{A}(\mathsf{iO}(\lambda, C_1)) = 1]| \leq \mathsf{negl}(\lambda)$$

Sub-exponential security We say that iO is sub-exponentially secure when it satisfies Definition 8 and also it satisfies that: for every (not necessary uniform) PPT distinguisher $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{iO}}(\lambda, \mathcal{A})$ is bounded by $\frac{1}{2^{\lambda^\epsilon}}$, for some positive constant $0 < \epsilon < 1$.

2.6 Dual-Mode NIWI Proof Systems

A dual-mode non-interactive witness indistinguishable (DM-NIWI) proof system [32] is a special type of non-interactive witness indistinguishable (NIWI) proof system, in which the common reference string (CRS) generation is dual-mode. The dual-mode property means that these systems have common reference string algorithms which generate indistinguishable CRS in "binding" or "hiding" modes. The system satisfies statistical soundness and extractability in binding mode and statistical witness indistinguishability in hiding mode.

Definition 9. *A binary relation R is polynomially bounded if it is decidable in polynomial time and there is a polynomial p such that $|w| \leq p(|x|)$, for all $(x, w) \in R$. For any such relation and any x we set $L_R = \{x | \ \exists w \ s.t. \ (x, w) \in R\}$.*

Definition 10 ([32] Dual-mode non-interactive witness indistinguishable proof systems). *Let R be a polynomially-bounded binary relation R. A* dual-mode non-interactive witness indistinguishable (DM-NIWI) proof system *for language $\mathcal{L}_R \in \mathsf{NP}$ is a tuple of PPT algorithms* $\mathsf{DM\text{-}NIWI} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Extract})$.

$\mathsf{Setup}(1^\lambda, \text{binding})$ *on input the security parameter, outputs a common reference string* crs *which we call binding. It also outputs the corresponding extraction trapdoor* $\mathsf{td}_{\mathsf{ext}}$.
$\mathsf{Setup}(1^\lambda, \text{hiding})$ *on input the security parameter, outputs a common reference string* crs*, which we call a hiding* crs.
$\mathsf{Prove}(\mathsf{crs}, x, w)$*, on input* crs*, a statement x and a witness w, outputs a proof π.*
$\mathsf{Verify}(\mathsf{crs}, x, \pi)$*, on input* crs*, a statement x and a proof π, outputs either 1 or 0.*
$\mathsf{Extract}(\mathsf{td}_{\mathsf{ext}}, x, \pi)$ *on input the extraction trapdoor* $\mathsf{td}_{\mathsf{ext}}$*, a statement x and a proof π, it outputs a witness w.*

We require the $\mathsf{DM\text{-}NIWI}$ *to meet the following properties:*

CRS indistinguishability. *Common reference strings generated via* $\mathsf{Setup}(1^\lambda, \text{binding})$ *and* $\mathsf{Setup}(1^\lambda, \text{hiding})$ *are computationally indistinguishable. More formally, for all non-uniform* PPT *adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the experiment* $\mathsf{Exp\text{-}CRS\text{-}IND}$ *described in Figure 4 is negligible, namely:*

$$\mathsf{Adv}^{\mathsf{DM\text{-}NIWI}}_{\mathsf{Exp\text{-}CRS\text{-}IND}}(\lambda, \mathcal{A}) := \Big| \Pr[\mathsf{Exp\text{-}CRS\text{-}IND}_0^{\mathsf{DM\text{-}NIWI}}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathsf{Exp\text{-}CRS\text{-}IND}_1^{\mathsf{DM\text{-}NIWI}}(1^\lambda, \mathcal{A}) = 1] \Big| \leq \mathsf{negl}(\lambda)$$

```
Experiment $\mathsf{Exp\text{-}CRS\text{-}IND}_b^{\mathsf{DM\text{-}NIWI}}(1^\lambda, \mathcal{A})$
if $b = 0$ then
    $(\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding})$
else
    $(\mathsf{crs}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{hiding})$
$b' \leftarrow_{\mathrm{R}} \mathcal{A}(\mathsf{crs})$
Return $b = b'$
```

Fig. 4. Experiment $\mathsf{Exp\text{-}CRS\text{-}IND}_b^{\mathsf{DM\text{-}NIWI}}$ for CRS indistinguishability.

Perfect completeness in both modes. *For every $(x, w) \in R$, we have that:*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding}), \\ \pi \leftarrow_{\mathrm{R}} \mathsf{Prove}(\mathsf{crs}, x, w)\end{array} : \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1\right] = 1.$$

The same holds when instead of $\mathsf{crs} \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding})$*, we have* $\mathsf{crs} \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{hiding})$.

Statistical soundness in binding mode. *The system is statistically sound if for every (possibly unbounded) adversary $\mathcal{A}$, we have that*

$$\Pr\left[\begin{array}{l}(\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding}), \\ (x, \pi) \leftarrow_{\mathrm{R}} \mathcal{A}(\mathsf{crs})\end{array} : \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L}_R\right] = \mathsf{negl}(\lambda).$$

Statistical extractability in binding mode *For any (x, π), it holds that:*

$$\Pr\left[\begin{array}{l}(\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding}), \\ w \leftarrow_{\mathrm{R}} \mathsf{Extract}(\mathsf{crs}, \mathsf{td}_{\mathsf{ext}}, x, \pi)\end{array} : \left(\begin{array}{c}\mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \\ \implies (x, w) \in R\end{array}\right)\right] = 1 - \mathsf{negl}(\lambda).$$

Note: In binding mode, statistical extractability implies statistical soundness.

Statistical witness-indistinguishability in hiding mode *We say that the* DM-NIWI *system is statistically witness-indistinguishable if for every x, w_0, w_1 with both $(x, w_0) \in R$ and $(x, w_1) \in R$, proofs of x with witness w_0 are indistinguishable from proofs of x with witness w_1. More formally, for every interactive (potentially unbounded) adversary $\mathcal{A}$:*

$$\left|\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{hiding}), \\ (x, w_0, w_1) \leftarrow_{\mathrm{R}} \mathcal{A}(\mathsf{crs}), \\ b \leftarrow_{\mathrm{R}} \{0,1\}, \\ \pi \leftarrow_{\mathrm{R}} \mathsf{Prove}(\mathsf{crs}, x, w_b)\end{array} : \mathcal{A}(\mathsf{crs}, \pi) = b\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ is restricted to choosing (x, w_0, w_1), such that both $(x, w_0) \in R$ and $(x, w_1) \in R$.

Remark. Like with the original presentation of Groth and Sahai [32], we focus our presentation on *witness-indistinguishable* (and not zero-knowledge) proof systems. Unlike zero-knowledge, witness-indistinguishability has useful compositional properties (see [23]). If zero-knowledge is desired, however, a simple transformation is possible: instead of proving $x \in L$, prove $x \in L \vee \hat{x} \in \hat{L}$ with our system, where $\hat{L}$ is any fixed hard-to-decide language, and $\hat{x}$ is a fixed instance determined in crs. In binding mode, set up $\hat{x} \notin \hat{L}$, so that $x \in L \vee \hat{x} \in \hat{L}$ implies $x \in L$. In hiding mode, set up $\hat{x} \in \hat{L}$, in which case a witness to this fact can be used as a simulation trapdoor to efficiently simulated proofs that achieve statistical zero-knowledge.

2.7 Hidden Bits Non-Interactive Zero-Knowledge

In our construction, we rely on a NIZK protocol in the hidden bits model. The hidden-bits model was introduced by [22] and is an idealized setting in which the bits of the common reference string are hidden from the verifier (but not from the prover). We call this the hidden reference string hrs.

When the prover computes a proof, it can choose which bits of hrs to reveal to the verifier. Denote the revealed bit set by $\mathcal{I}$, then by $\mathsf{hrs}_{\mathcal{I}}$ we will refer to the corresponding revealed bits of the hrs. Our construction can be based on the hidden-bits NIZK from [22], which proves graph Hamiltonicity and therefore covers any NP statement. Nevertheless, our construction is generic enough to be based on any hidden-bits NIZK with statistical soundness and perfect zero-knowledge (if we only had statistical ZK, then we would only get statistical correctness of DM-NIWI). The hidden-bits NIZK from [22] satisfies both statistical soundness and perfect ZK and we briefly recap it in Appendix C.

Definition 11. [22] *A pair of* PPT *algorithms* $\mathsf{NIZK}_H = (\mathsf{P}_H, \mathsf{V}_H)$ *is a NIZK proof system in the hidden-bits model if it satisfies the following properties:*

1. *Completeness: there exists a polynomial r denoting the length of the hidden random string, such that for every $(x, w) \in \mathcal{R}$ we have that:*

$$\Pr_{\mathsf{P}_H, \mathsf{hrs} \leftarrow \{0,1\}^{t(|x|,\lambda)}}[(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs}) : \mathsf{V}_H(x, \mathsf{hrs}_{\mathcal{I}}, \mathcal{I}, \pi) = 1] = 1$$

   where $\mathcal{I} \subseteq [t(|x|, \lambda)]$ and $\mathsf{hrs}_{\mathcal{I}} = \{\mathsf{hrs}[i] : i \in \mathcal{I}\}$.

2. *Statistical Soundness: for every $x \notin \mathcal{L}$ we have that:*

$$\Pr_{\mathsf{hrs}\leftarrow\{0,1\}^{t(|x|,\lambda)}}[\exists \pi, \mathcal{I} : \mathsf{V}_H(x, \mathsf{hrs}_{\mathcal{I}}, \mathcal{I}, \pi) = 1] < \tfrac{1}{2^{\lambda+|x|}}.$$

3. *Perfect Zero-Knowledge: there exists a* PPT *algorithm* S_H *such that:*

$$D_0 \coloneqq \{(\mathsf{hrs}_{\mathcal{I}}, \pi, \mathcal{I}) : \mathsf{hrs} \leftarrow \{0,1\}^{t(|x|,\lambda)}, (\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})\}_{(x,w)\in\mathcal{R}} \equiv$$
$$\equiv \{\mathsf{S}_H(x)\}_{(x,w)\in\mathcal{R}} =: D_1$$

For ease of notation, we denote by $\Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda) \coloneqq \Delta(D_0, D_1)$ *the statistical distance between distributions* D_0 *and* D_1*. In the case of perfect ZK,* $\Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda) \coloneqq \Delta(D_0, D_1) = 0$.

3 Construction

In Figure 5, we describe our $\mathsf{DM\text{-}NIWI}$ candidate. Our scheme uses a hidden-bits NIZK proof system $\mathsf{NIZK}_H = (\mathsf{P}_H, \mathsf{V}_H)$ as a building block. To distinguish common reference strings and proofs between the two proof systems, we denote by lowercase (π, hrs) the proofs and hidden reference strings for NIZK_H. In contrast, the common reference string and proofs of $\mathsf{DM\text{-}NIWI}$ are denoted as CRS and $\mathit{\Pi}$, respectively.

The CRS of $\mathsf{DM\text{-}NIWI}$ contains the public key lpk of a lossy encryption scheme LE, a lossy function H, uniformly random Z and crs, a functional decryption function sk_f and an obfuscated program PC. Prover program $\mathsf{PC}(x, w, r)$ first encrypts (x, w) using randomness r to obtain $a = \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$. Then it computes either a $\mathsf{HidingProof}$ or a $\mathsf{BindingProof}$ depending on the mode and outputs as proof a FE ciphertext C and a hidden-bits proof π. The verifier decrypts C using sk_f and then uses the hidden-bits verifier to check proof π.

Notation and parameters For security parameter λ, we denote by $p(|x| + \lambda)$ the ciphertext size of LE. By $p_2(|x|, \lambda)$, we denote the size of the randomness needed to compute FE ciphertexts, while $p_3(|x|, \lambda)$ denotes the size of the random tape needed by the hidden-bits simulator S_H. Recall that $t(|x|, \lambda)$ is the polynomial from Definition 11. Then LF must be a $\big(p_1(|x|, \lambda), \lambda, t(|x|, 2\lambda + |x|), p(|x| + \lambda) + \lambda\big)$-lossy function. Consider the subexponential security level of iO, FE and PRF to be $\frac{1}{2^{\kappa^{\epsilon}}}$, for some constant $0 < \epsilon < 1$. Then κ must be chosen as $(p(|x| + \lambda) + \lambda)^{(1/\epsilon)}$.

4 Security Proof

Theorem 12. *Let* PRF *be a subexponentially-secure puncturable pseudo-random function,* iO *be a subexponentially-secure obfuscator,* PRG *a secure pseudo-random generator,* LE *a secure lossy encryption scheme and* FE *a subexponentially-secure selectively-*$\mathsf{IND\text{-}CPA}$ *functional encryption scheme, then the scheme* $\mathsf{DM\text{-}NIWI} = (\mathsf{DM\text{-}NIWI.Setup}, \mathsf{DM\text{-}NIWI.Prover}, \mathsf{DM\text{-}NIWI.Verifier})$ *described in Figure 5 is a secure dual-mode non-interactive witness-indistinguishable system.*

4.1 Completeness

Lemma 13. *The* $\mathsf{DM\text{-}NIWI}$ *system in Figure 5 is perfectly complete.*

Proof. Completeness follows from the completeness of the hidden-bits NIZK_H, the perfect ZK of NIZK_H, the perfect correctness of FE and the functionality of iO (the fact that for all programs C, we have that $\mathsf{iO}(C)$ is functionally equivalent to C). Consider any $(x, w) \in R$ and $(C, \pi) = \mathsf{DM\text{-}NIWI.Prover}(\mathsf{CRS}, x, w, r)$. We want to show that $\mathsf{DM\text{-}NIWI.Verifier}(C, \pi, \mathsf{CRS}) = 1$ with probability 1.

Case 1: $\mathsf{CRS} \leftarrow_{\mathbf{R}} \mathsf{DM\text{-}NIWI.Setup}(\mathbf{1}^{\lambda}, \text{binding})$ Since $(C, \mathit{\Pi})$ is a proof computed by the honest prover, we know that $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$, where hrs is derived from a, the lossy encryption of (x, w). From the perfect correctness of FE, we have that indeed $(T \oplus \mathsf{crs})_{\mathcal{I}} = \mathsf{hrs}_I$. Therefore, from the perfect correctness of NIZK_H, it follows that $\mathsf{V}_H(\mathcal{I}, (T \oplus \mathsf{crs})_{\mathcal{I}}, x, \pi)$ accepts with probability 1.

```
$\mathsf{Setup}(1^\lambda, \text{mode})$
  $\mathsf{PRG} \leftarrow_R \mathsf{PRG.Setup}(1^\lambda)$
  if mode = binding then
    $\mathsf{H} \leftarrow_R \mathsf{LF.Setup}(1^\lambda, \text{lossy})$
  else
    $\mathsf{H} \leftarrow_R \mathsf{LF.Setup}(1^\lambda, \text{dense})$
  $(\mathsf{lpk}, \mathsf{lsk}) \leftarrow_R \mathsf{LE.Setup}(1^\lambda, \text{lossy})$
  $K_1, K_2, K_3 \leftarrow_R \mathsf{PRF.KeyGen}(1^\kappa)$
  $(\mathsf{fmpk}, \mathsf{fmsk}) \leftarrow_R \mathsf{FE.Setup}(1^\kappa)$
  $\mathsf{sk_f} \leftarrow_R \mathsf{FE.KeyGen}(\mathsf{fmsk}, \mathsf{f})$
  $\mathsf{crs} \leftarrow_R \{0,1\}^{t(|x|, 2\lambda+|x|)}$
  $z \leftarrow_R \{0,1\}^\lambda$
  if mode = binding then
    $Z \leftarrow_R \{0,1\}^{2\lambda+|x|}$
  else
    $Z \leftarrow \mathsf{PRG}(z)$
  $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{\text{mode},\mathsf{crs}})$
  $\mathsf{CRS} := (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})$
  if mode = binding then
    Return $(\mathsf{CRS}, \mathsf{td_{ext}} := \mathsf{fmsk})$
  Return $\mathsf{CRS}$

$\mathsf{Prover}(\mathsf{PC}, x, w, r)$
  Return $\Pi := \mathsf{PC}(x, w, r)$

$\mathsf{Verifier}(\mathsf{CRS}, x, \Pi := (C, \pi))$
  $(T, \mathcal{I}) \leftarrow \mathsf{FE.Dec}(\mathsf{sk_f}, C)$
  $\mathsf{hrs}_\mathcal{I} \leftarrow T \oplus \mathsf{crs}_\mathcal{I}$
  return $\mathsf{V}_H(x, \mathsf{hrs}_\mathcal{I}, \mathcal{I}, \pi)$

$\mathsf{ProgProv}_{\text{mode},\mathsf{crs}}(x, w, r)$
  Hardcoded: Keys $K_1, K_2, K_3, z$
  if $(x, w) \notin R$
    Return $\perp$
  $a \leftarrow_R \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$
  if mode = binding then
    $(C, \pi) = \mathsf{BindingProof_{crs}}(x, w, a)$
  else
    $(C, \pi) = \mathsf{HidingProof_{crs}}(x, a)$
  Return $\Pi := (C, \pi)$

$\mathsf{BindingProof_{crs}}(x, w, a)$
  Hardcoded : Keys $K_1, K_2$
  $X \leftarrow \mathsf{PRF}(K_1, a)$
  $\mathsf{hrs} \leftarrow \mathsf{H}(X) \oplus \mathsf{crs}$
  $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
  $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
  $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (X, \mathcal{I}, 0, 0); r_2)$
  Return $\Pi := (C, \pi)$

$\mathsf{HidingProof_{crs}}(x, a)$
  Hardcoded : Keys $K_2, K_3$
  $r_3 \leftarrow \mathsf{PRF}(K_3, a)$
  $(\mathsf{hrs}_\mathcal{I}, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$
  $T \leftarrow \mathsf{hrs}_\mathcal{I} \oplus \mathsf{crs}_\mathcal{I}$
  $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
  $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}, z, T); r_2)$
  Return $\Pi := (C, \pi)$

$\mathsf{f}(C = \mathsf{FE.Enc}(\mathsf{fmpk}, (X, \mathcal{I}, z, T)))$
  Hardcoded : Parameters $Z, \mathsf{H}$
  if $\mathsf{PRG}(z) = Z$ then return $(T, \mathcal{I})$.
  else return $(\mathsf{H}(X)_\mathcal{I}, \mathcal{I})$
```

Fig. 5. Dual-mode NIWI scheme $\mathsf{DM\text{-}NIWI} = (\mathsf{Setup}, \mathsf{Prover}, \mathsf{Verifier})$. LF is a class of lossy functions, $\mathsf{PRG.Setup}$ outputs pseudo-random generators from $\{0.1\}^\lambda$ to $\{0,1\}^{2\lambda+|x|}$, FE is a functional encryption scheme, LE is a lossy encryption scheme, iO is an indistinguishability obfuscator and $(\mathsf{P}_H, \mathsf{V}_H)$ is the hidden-bits model NIZK from [22]. Parameter κ is chosen so that the sub-exponential security level is sufficient.

Case 2: $\mathsf{CRS} \leftarrow_R \mathsf{DM\text{-}NIWI.Setup}(\mathbf{1}^\lambda, \text{hiding})$ Since (C, Π) is a proof computed by the honest prover, we know that $(\mathsf{hrs}_\mathcal{I}, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$, where r_3 is the random tape used by the hidden-bits simulator S_H. By the perfect correctness of FE, decrypting C yields indeed $\mathsf{hrs}_\mathcal{I} \oplus \mathsf{crs}_\mathcal{I}$, therefore we can recover $\mathsf{hrs}_\mathcal{I}$. Now, since NIZK_H has perfect zero-knowledge, it follows that $\mathsf{V}_H(\mathcal{I}, (T \oplus \mathsf{crs})_\mathcal{I}, x, \pi)$ accepts with probability 1 (or otherwise simulated proofs would not be identically distributed to real ones).

4.2 Soundness

Theorem 14. *When in binding mode, the* $\mathsf{DM\text{-}NIWI}$ *system in Figure 5 is statistically sound.*

Proof. Here we use the soundness of the hidden-bits scheme, coupled with the lossiness of function H.

Since crs is uniformly random, computing $\mathsf{hrs} := \mathsf{H}(\mathsf{PRF}(K_1, a)) \oplus \mathsf{crs}$ will yield another uniformly random string and will allow us to use the soundness of the hidden-bits system. Moreover, we leverage the lossiness of H to ensure that an adversary cannot influence the hrs sufficiently enough as to be able to cheat. This is because the honest verifier applies H automatically when it functionally decrypts ciphertext C.

More formally, fix some $x \in \{0,1\}^n \setminus \mathcal{L}$. We prove that with overwhelming probability over the common reference string, there is no proof Π which will be accepted by the verifier. This is a selective notion which we later amplify to obtain the security notion from Definition 10.

We want to bound $\Pr_{(\mathsf{CRS},\mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda,\text{binding})}[\exists \Pi : \mathsf{Verifier}(\Pi, \mathsf{CRS}) = 1]$. We can rewrite this probability as:

$$\Pr_{\substack{Z \leftarrow_{\mathrm{R}} \{0,1\}^{2\lambda+|x|} \\ \mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)} \\ \mathsf{H},\mathsf{PC},\mathsf{fmpk},\mathsf{fmsk},\mathsf{sk_f}}}[\exists(\pi, C) : \mathsf{Verifier}((\pi, C), (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})) = 1]$$

Now, we condition on the event E that Z does not have a PRG preimage, which happens with probability $1 - \frac{1}{2^{\lambda+|x|}}$. If Z has no preimage, then from the functionality of iO and the special correctness of the FE scheme (see definition 5), the adversary must produce a ciphertext which decrypts in the same way as ciphertext $C = \mathsf{FE.Enc}(X, \mathcal{I}, \cdot, \cdot)$. Note that both the functional equivalence of iO and the special correctness of the functional encryption scheme are statistical properties. Therefore, the probability above is less or equal than:

$$\Pr_{\mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)}}[\exists(\pi, X, \mathcal{I}) : \mathsf{V}_H(x, (\mathsf{crs} \oplus \mathsf{H}(X))_{\mathcal{I}}, \mathcal{I}, \pi) = 1]$$

The next step is to bound the number of possible values of hrs. Recall that $\mathsf{hrs} := \mathsf{H}(\mathsf{PRF}(K_1, a)) \oplus \mathsf{crs}$. From the lossiness of H, we know that there are at most 2^k images of H, where k is the second parameter of H (see Definition 2). Thus, we can compute an union bound over all these images $\mathsf{H}(X)$, bounding the above probability by:

$$2^k \times \Pr_{\mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)}}[\exists(\pi, \mathcal{I}) : \mathsf{V}_H(x, (\mathsf{crs} \oplus \mathsf{H}(X))_{\mathcal{I}}, \mathcal{I}, \pi) = 1]$$

Now, recall that we denote $\mathsf{crs} \oplus H(X)$ as hrs. Since crs is uniformly randomly distributed, so is hrs, and we can rewrite the probability above as:

$$2^k \times \Pr_{\mathsf{hrs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)}}[\exists(\pi, \mathcal{I}) : \mathsf{V}_H(x, \mathsf{hrs}_{\mathcal{I}}, \mathcal{I}, \pi) = 1]$$

Finally, by using the soundness of the hidden-bits NIZK, we know that:

$$\Pr_{\mathsf{hrs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)}}[\exists(\pi, \mathcal{I}) : \mathsf{V}_H(x, \mathsf{hrs}_{\mathcal{I}}, \mathcal{I}, \pi) = 1] \leq \frac{1}{2^{2\lambda+|x|}}$$

Therefore, we can conclude that:

$$\Pr_{(\mathsf{CRS},\mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda,\text{binding})}[\exists \Pi : \mathsf{Verifier}(\Pi, \mathsf{CRS}) = 1] \leq \tfrac{1}{2^{2\lambda+|x|-k}}.$$

The only remaining step is to amplify the security from the selective variant we have just proven to the adaptive one from Definition 10. We eliminate the restriction that x is fixed by computing a union bound over all possible values of x. In particular, for H parameter $k = \lambda$, we conclude that for every unbounded adversary $\mathcal{A}$:

$$\Pr\left[\begin{array}{l}(\mathsf{CRS}, \mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^\lambda, \text{binding}), \\ (x, \Pi) \leftarrow_{\mathrm{R}} \mathcal{A}(\mathsf{CRS})\end{array} : \mathsf{Verifier}(\mathsf{CRS}, x, \Pi) = 1 \wedge x \notin \mathcal{L}_R\right] = \frac{1}{2^\lambda}.$$

As a last check, we must ensure that event $\neg E$ still happens with negligible probability. If we compute the same union bound as above, the probability of $\neg E$ is now bounded by $\frac{1}{2^\lambda}$. Therefore, the system is statistically sound.

4.3 Witness Indinstinguishability

Theorem 15. *In hiding mode, the* $\mathsf{DM\text{-}NIWI}$ *system from Figure 5 is statistically witness-indistinguishable.*

Proof. By using the statistical lossiness of LE, we show that no (potentially unbounded) adversary $\mathcal{A}$ can break the witness-indistinguishability of $\mathsf{DM\text{-}NIWI}$. Recall that the lossiness of LE implies that for all $(\mathsf{lpk}, \perp) \leftarrow \mathsf{LE.Gen}(1^\lambda, \text{lossy})$, and for all x, w_0, w_1, encryptions of (x, w_0) are statistically indistinguishable from encryptions of (x, w_1). More formally:

$$D_0 \coloneqq \{r \leftarrow R : (\mathsf{lpk}, \mathsf{LE.Enc}(\mathsf{lpk}, (x, w_0), r))\} \approx_{\frac{1}{2^\lambda}}$$
$$\approx_{\frac{1}{2^\lambda}} \{r \leftarrow R : (\mathsf{lpk}, \mathsf{LE.Enc}(\mathsf{lpk}, (x, w_1), r))\} \eqqcolon D_1.$$

The goal is to show that for every hiding CRS and for every (x, w_0, w_1), with both $(x, w_0) \in R$ and $(x, w_1) \in R$, proofs for (x, w_0) are statistically indistinguishable from proofs for (x, w_1). Fix (x, w_0, w_1) and let D'_b be the following distribution:

$$D'_b \coloneqq \left\{\mathsf{CRS} \leftarrow_{\text{R}} \mathsf{DM\text{-}NIWI.Setup}(1^\lambda, \text{hiding}) : \pi \leftarrow_{\text{R}} \mathsf{DM\text{-}NIWI.Prove}(\mathsf{CRS}, x, w_b)\right\} \tag{1}$$

We want to prove that we have that $D'_0 \approx_{\frac{1}{2^\lambda}} D'_1$. To achieve this, we exhibit a probabilistic function F which on input D_b outputs D'_b, i.e. $F(D_b) = D'_b$, without needing to know bit b. If such an F exists, then $D_0 \approx_{\frac{1}{2^\lambda}} D_1$ implies that $F(D_0) \approx_{\frac{1}{2^\lambda}} F(D_1)$. Function F works as follows:

1. F obtains public key lpk from D_b. Then F esentially computes $\mathsf{DM\text{-}NIWI.Setup}(1^\lambda)$ and chooses all the parameters itself, except for lpk which comes from D_b.
   In more detail, F chooses the PRG, a dense function H, keys K_1, K_2, K_3, master keys $(\mathsf{fmpk}, \mathsf{fmsk})$ and functional key $\mathsf{sk_f}$ just as in $\mathsf{DM\text{-}NIWI.Setup}(1^\lambda)$. It also draws uniformly random strings z and crs. It then sets $Z = \mathsf{PRG}(z)$ and uses all these parameters to construct program $\mathsf{ProgProv}_{\text{hiding},\mathsf{crs}}$, which it obfuscates obtaining PC.
2. For hiding CRS, we have that PC obfuscates $\mathsf{ProgProv}_{\text{hiding},\mathsf{crs}}$. Therefore, F can compute the output of $\mathsf{DM\text{-}NIWI.Prove}(\mathsf{CRS}, x, w_b)$ even without knowing bit b: F has access to ciphertext ct from distribution D_b. Ciphertext ct can originate from either (x, w_0) or (x, w_1). F simply computes $(C, \pi) \leftarrow_{\text{R}} \mathsf{HidingProof}_{\mathsf{crs}}(x, \mathsf{ct})$ and uses (C, π) to construct distribution D'_b. Observe that this is only possible because $\mathsf{HidingProof}_{\mathsf{crs}}(x, \mathsf{ct})$ crucially only has x and ct as inputs and does not directly depend on witnesses w_0, w_1 themselves.

We have shown that $F(D_0) \approx_{\frac{1}{2^\lambda}} F(D_1)$, for every (x, w_0, w_1) and for all hiding $\mathsf{CRS} \leftarrow_{\text{R}} \mathsf{DM\text{-}NIWI.Setup}(1^\lambda, \text{hiding})$. This concludes witness-indistinguishability as defined in Definition 10. (In Definition 10, the adversary can choose (x, w_0, w_1) after seeing the CRS, but since $F(D_0) \approx_{\frac{1}{2^\lambda}} F(D_1)$ for every (x, w_0, w_1) and for every hiding CRS, the adversary will not have advantage greater that $\frac{1}{2^\lambda}$).

4.4 CRS Indistinguishability

Theorem 16. *The* $\mathsf{DM\text{-}NIWI}$ *system from Figure 5 satisfies computational indistinguishability between common reference strings generated in binding mode and common reference strings generated in hiding mode.*

Proof. The proof proceeds by a sequence of games where G_0 is defined exactly as $\mathsf{Exp\text{-}CRS\text{-}IND}_0(1^\lambda, \mathcal{A})$ (see Figure 4). G_0 corresponds to the experiment in which adversary $\mathcal{A}$ against crs indistinguishability receives common reference strings in binding mode. A high-level summary is provided in Figure 6. For any game G_i, we denote by $\mathsf{Adv}_i(A)$ the advantage of $\mathcal{A}$ in G_i, that is, $\Pr[\mathsf{G}_i(1^\lambda, \mathcal{A}) = 1]$, where the probability is taken over the random coins of G_i and $\mathcal{A}$. At a high level, we use four hybrid games $\mathsf{G}_0, \mathsf{G}_1, \mathsf{G}_2$ and G_3. The proof is in three phases:

1. In the first phase, we transition from G_0 to G_1. Game G_1 is defined to be the same as G_0, except for the following two changes: First, we switch the mode of the lossy function H from lossy to dense. This is done with the end goal of ensuring that the output of H is uniformly distributed at specific values of a.
   Secondly, we use the security of the PRG to change Z from being uniformly random to being in the image of the PRG. This is done by setting $Z = \mathsf{PRG}(z)$. To anticipate, this will provide us with a trapdoor for replacing functional ciphertext encoding X with ciphertexts encoding hrs_I. The fact that $\mathsf{G}_0 \approx_{\mathrm{C}} \mathsf{G}_1$ is proven in Lemma 17.
2. In the second phase, we transition from G_1 to G_2. Game G_2 is defined to be precisely the same as G_1, except that $\mathsf{DM\text{-}NIWI.Setup}(1^\lambda)$ computes $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{\mathrm{hiding},\mathsf{crs}})$. This transition only makes changes in the program $\mathsf{ProgProv}$. By iterating over all values of a, for each a we replace real proofs by simulated proofs from the hidden-bits simulator S_H.
   We carefully leverage PRF security, the injective mode of LE and the density of H to ensure that for a specific a^*, its corresponding hrs^* is of the form $\beta \oplus \mathsf{crs}$, for uniformly random β. Then we use functional encryption security to replace the functional ciphertext corresponding to a^* to one which only leaks $\mathsf{hrs}_\mathcal{I}$. But at this stage, since only $\mathsf{hrs}_\mathcal{I}$ is encoded in the ciphertext, we can use the zero knowledge of the hidden-bits NIZK to replace real proofs by simulated ones. We formally prove that $\mathsf{G}_1 \approx_{\mathrm{C}} \mathsf{G}_2$ in Theorem 19.
3. In the third stage, we define G_3 to be the same as $\mathsf{Exp\text{-}CRS\text{-}IND}_1(1^\lambda, \mathcal{A})$. The only difference between G_2 and G_3 is that in the later, the public key of the lossy encryption scheme LE is switched from injective to lossy mode. We prove that $\mathsf{G}_2 \approx_{\mathrm{C}} \mathsf{G}_3$ in Lemma 18.

Game	$(\mathsf{lpk}, \mathsf{lsk})$	H	Z	PC	Mode or Remark
G_0	$\mathsf{LE.Setup}(1^\lambda, \mathrm{inj})$	$\mathsf{LF.Setup}(1^\lambda, \mathrm{lossy})$	$Z \leftarrow_{\mathrm{R}} \{0,1\}^{2\lambda+\lvert x\rvert}$	$\mathsf{iO}(\mathsf{ProgProv}_{\mathrm{binding}})$	Binding
G_1	$\mathsf{LE.Setup}(1^\lambda, \mathrm{inj})$	$\mathsf{LF.Setup}(1^\lambda, \mathrm{dense})$	$Z \leftarrow \mathsf{PRG}(z)$	$\mathsf{iO}(\mathsf{ProgProv}_{\mathrm{binding}})$	Lemma 17
G_2	$\mathsf{LE.Setup}(1^\lambda, \mathrm{inj})$	$\mathsf{LF.Setup}(1^\lambda, \mathrm{dense})$	$Z \leftarrow \mathsf{PRG}(z)$	$\mathsf{iO}(\mathsf{ProgProv}_{\mathrm{hiding}})$	Theorem 19
G_3	$\mathsf{LE.Setup}(1^\lambda, \mathrm{lossy})$	$\mathsf{LF.Setup}(1^\lambda, \mathrm{dense})$	$Z \leftarrow \mathsf{PRG}(z)$	$\mathsf{iO}(\mathsf{ProgProv}_{\mathrm{hiding}})$	Lemma 18 Hiding

Fig. 6. An overview of the games used in the proof of Theorem 16, changes between consecutive games are highlighted with gray boxes.

Lemma 17 (From G_0 to G_1). *For every* PPT *adversary* $\mathcal{A}$*, it holds that* $|\mathsf{Adv}_0(\mathcal{A}) - \mathsf{Adv}_1(\mathcal{A})| \leq \mathsf{negl}(\lambda)$.

Proof. The only differences between G_0 and G_1 are the fact that Z is changed from $Z \leftarrow_{\mathrm{R}} \{0,1\}^{2\lambda+|x|}$ to $Z \leftarrow \mathsf{PRG}(z)$ and function H is changed from $\mathsf{H} \leftarrow \mathsf{LF.Setup}(1^\lambda, \mathrm{lossy})$ to $\mathsf{H} \leftarrow \mathsf{LF.Setup}(1^\lambda, \mathrm{dense})$. The lemma follows from the security of the PRG and from the computational indistinguishability of the modes of the lossy function LF. Namely, if $\mathcal{A}$ can distinguish between G_0 and G_1, there exists either a PPT adversary $\mathcal{B}_1$ that can break the security of the PRG or a PPT adversary $\mathcal{B}_2$ that can distinguish with non-negligible advantage between the lossy and dense modes of LF.

Lemma 18 (From G_2 to G_3). *For every* PPT *adversary* $\mathcal{A}$*, it holds that* $|\mathsf{Adv}_2(\mathcal{A}) - \mathsf{Adv}_3(\mathcal{A})| \leq \mathsf{negl}(\lambda)$.

Proof. The only change between G_2 and G_3 is that the $(\mathsf{lpk}, \mathsf{lsk})$ keys of LE are changed from injective to lossy. The lemma follows directly from the fact that $\{\mathrm{proj}(\mathsf{LE.Gen}(1^\lambda, \mathrm{inj}))\} \approx_{\mathrm{C}} \{\mathrm{proj}(\mathsf{LE.Gen}(1^\lambda, \mathrm{lossy}))\}$, where $\mathrm{proj} : (\mathsf{lpk}, \mathsf{lsk}) \to \mathsf{lpk}$ and from the fact that lsk is not used anywhere in the construction.

Theorem 19 (From G_1 to G_2). *For every* PPT *adversary* $\mathcal{A}$*, there exist* PPT *adversaries* $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$*, such that:*

$$|\mathsf{Adv}_0(\mathcal{A}) - \mathsf{Adv}_1(\mathcal{A})| \leq$$
$$2^{p(|x|+\lambda)}\left(8 \cdot \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}_1) + 4 \cdot \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B}_2) + \mathsf{Adv}^{\mathsf{FE}}_{\mathsf{Exp\text{-}}s\mathsf{\text{-}IND\text{-}FE\text{-}CPA}}(\kappa, \mathcal{B}_3) + \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda) + \frac{1}{2^{p(|x|+\lambda)+\lambda}}\right).$$

Proof. The proof strategy is to iterate over all values of $a = \mathsf{LE.Enc}(\mathsf{lpk}, (x, w), r)$ and make changes to the obfuscation of the program $\mathsf{ProgProv}$. We define a series of hybrids H_{1,a^*}, for all $a^* \in \{0,1\}^{p(|x|+\lambda)}$ in Figure 7. Briefly, hybrid H_{1,a^*} is defined as follows:

```
Hybrid $\mathsf{H}_{1,a^*}$

$\mathsf{Setup}(1^\lambda, \text{mode})$
  $\mathsf{PRG} \leftarrow_{\mathrm{R}} \mathsf{PRG.Setup}(1^\lambda)$
  $\mathsf{H} \leftarrow_{\mathrm{R}} \mathsf{LF.Setup}(1^\lambda, \text{dense})$
  $(\mathsf{lpk}, \mathsf{lsk}) \leftarrow_{\mathrm{R}} \mathsf{LE.Setup}(1^\lambda, \text{inj})$
  $K_1, K_2, K_3 \leftarrow_{\mathrm{R}} \mathsf{PRF.KeyGen}(1^\lambda)$
  $(\mathsf{fmpk}, \mathsf{fmsk}) \leftarrow_{\mathrm{R}} \mathsf{FE.Setup}(1^\lambda)$
  $\mathsf{sk_f} \leftarrow_{\mathrm{R}} \mathsf{FE.KeyGen}(\mathsf{fmsk}, \mathsf{f})$
  $\mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|, 2\lambda+|x|)}$
  $z \leftarrow_{\mathrm{R}} \{0,1\}^\lambda$
  $Z \leftarrow \mathsf{PRG}(z)$
  $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{1,a^*})$
  $\mathsf{CRS} := (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})$
  Return $\mathsf{CRS}$

$\mathsf{ProgProv}_{1,a^*}(x, w, r)$
  if $(x, w) \notin R$
    Return $\perp$
  Hardcoded: Keys $K_1, K_2, K_3, z$
  $a \leftarrow_{\mathrm{R}} \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$
  if a<$a^*$ then
    $(C, \pi) = \mathsf{HidingProof}_{\mathsf{crs}}(x, w, a)$
  if $a \geq a^*$
    $(C, \pi) = \mathsf{BindingProof}_{\mathsf{crs}}(x, a)$
  Return $\Pi := (C, \pi)$
```

Fig. 7. Hybrid $\mathsf{H}_{(1,a^*)}$ for the proofs of Theorems 19 and 20. Note that the Prover, $\mathsf{Verifier}$, $\mathsf{BindingProof}$, $\mathsf{HidingProof}$ and function f are the same as defined in Figure 5 and are not represented again for succinctness. Changes between hybrids $\mathsf{H}_(1, a^*)$ and game G_1 are highlighted in light gray.

Hybrid H_{1,a^*} is defined in the same way as game G_1, except that:

1. $\mathsf{DM\text{-}NIWI.Setup}$ is changed such that the computation of the public parameter $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{\text{binding},\mathsf{crs}})$ is replaced by $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{1,a^*})$.
2. Program $\mathsf{ProgProv}_{1,a^*}$ on inputs x, w, r is the program which first computes $a = \mathsf{LE.Enc}(\mathsf{lpk}, (x, w), r)$. Then it compares a with hardcoded value a^* and for $a < a^*$, it computes $(C, \pi) = \mathsf{HidingProof}_{\mathsf{crs}}(x, a)$, while for $a \geq a^*$ it computes $(C, \pi) = \mathsf{BindingProof}_{\mathsf{crs}}(x, w, a)$. It then returns proof (C, π).

Note that hybrid $\mathsf{H}_{1,0^{p(|x|+\lambda)}}$ is the same as game G_1, while hybrid $\mathsf{H}_{1,1^{p(|x|+\lambda)}}$ is the same as game $\mathsf{G}_2 = \mathsf{Exp\text{-}CRS\text{-}IND}_1(1^\lambda, \mathcal{A})$. Just as before, for every hybrid H_i, we denote by $\mathsf{Adv}_i(\mathcal{A})$ the advantage of $\mathcal{A}$ in H_i, that is, $\Pr[\mathsf{G}_i(1^\lambda, \mathcal{A}) = 1]$. In Theorem 20, we formally prove that every two consecutive hybrids $\mathsf{H}(1, a^*)$ and $\mathsf{H}(1, a^* + 1)$ are computationally indistinguishable, i.e. $\mathsf{H}_{(1,a^*-1)} \approx_{\mathrm{C}} \mathsf{H}_{(1,a^*)}$, for every $a^* \in [2^{p(|x|+\lambda)}]$.

Theorem 20 (From $\mathsf{H}_{(1,a^*)}$ to $\mathsf{H}_{(1,(a^*+1))}$). *For every* PPT *adversary $\mathcal{A}$, there exist* PPT *adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$, such that:*

$$|\mathsf{Adv}_{(1,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(1,(a^*+1))}(\mathcal{A})| \leq$$
$$8 \cdot \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}_1) + 4 \cdot \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B}_2) + \mathsf{Adv}^{\mathsf{FE}}_{\mathsf{Exp\text{-}}s\mathsf{\text{-}IND\text{-}FE\text{-}CPA}}(\kappa, \mathcal{B}_3) + \varDelta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda) + \frac{1}{2^{p(|x|+\lambda)+\lambda}}.$$

Proof. We prove this through a sequence of hybrids $\mathsf{H}_{(1,a^*)}$ up to $\mathsf{H}_{(15,a^*)}$, where hybrid $\mathsf{H}_{(15,a^*)}$ is identical to hybrid $\mathsf{H}_{(1,(a^*+1))}$. In terms of notation, hybrid $\mathsf{H}_{(i,a^*)}$ will have $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{i,a^*,\mathsf{crs}})$. The proof strategy is to leverage the properties of $\mathsf{iO}, \mathsf{FE}, \mathsf{PRF}s, \mathsf{LE}$ and H in order to replace actual proofs computed by the hidden-bits prover P_H to simulated proofs computed by S_H. Notice that in $\mathsf{H}_{(1,a^*)}$, proofs corresponding to a are computed by subprogram $\mathsf{BindingProof}_{\mathsf{crs}}(x, w, a)$, while in $\mathsf{H}_{(15,a^*)}$ they are computed by subprogram $\mathsf{HidingProof}_{\mathsf{crs}}(x, w)$. This is the only difference between the two hybrids. In order to replace subprogram $\mathsf{BindingProof}_{\mathsf{crs}}()$ by $\mathsf{HidingProof}_{\mathsf{crs}}()$ we define a series of subprograms $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, for $i \in [15]$. As expected, every hybrid $\mathsf{H}_{(i,a^*)}$ will be defined to be identical to H_{1,a^*}, except that for $a = a^*$, $(C, \pi) = \mathsf{HybridProof}_{i,a^*,\mathsf{crs}}(x, w, a)$. The hybrids are described in Figure 7. For a detailed decription of subprograms $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, see Figures 9 to 12.

Hybrid $\mathsf{H}_{(2,a^*)}$ In this hybrid, the subprogram $\mathsf{HybridProof}_{2,a^*,\mathsf{crs}}$ is changed so that key K_1 is punctured at point a^*. This is a standard punctured programming technique. Once we puncture the key, only $K_1\{a^*\}$ is hardcoded in the program, along with the evaluation of $r_1^* \leftarrow \mathsf{PRF}(K_1, a^*)$, but not K_1 itself. Observe that key K_1 is punctured in $\mathsf{ProgProv}_{2,a^*,\mathsf{crs}}$ and all its subprograms as well. In $\mathsf{H}_{(i,a^*)}, i \in [15]$ subprograms $\mathsf{BindingProof}_{\mathsf{crs}}(x, w, a)$ and $\mathsf{HidingProof}_{\mathsf{crs}}(x, w, a)$ are never called on inputs $a \neq a^*$, so they never need the evaluation of $\mathsf{PRF}(K_1, a^*)$.

```
Hybrid $\mathsf{H}_{1,a^*}, \ldots, \mathsf{H}_{15,a^*}$

$\mathsf{Setup}(1^\lambda, \text{mode})$
  $\mathsf{PRG} \leftarrow_{\mathrm{R}} \mathsf{PRG.Setup}(1^\lambda)$
  $\mathsf{H} \leftarrow_{\mathrm{R}} \mathsf{LF.Setup}(1^\lambda, \text{dense})$
  $(\mathsf{lpk}, \mathsf{lsk}) \leftarrow_{\mathrm{R}} \mathsf{LE.Setup}(1^\lambda, \text{inj})$
  $K_1, K_2, K_3 \leftarrow_{\mathrm{R}} \mathsf{PRF.KeyGen}(1^\kappa)$
  $(\mathsf{fmpk}, \mathsf{fmsk}) \leftarrow_{\mathrm{R}} \mathsf{FE.Setup}(1^\kappa)$
  $\mathsf{sk_f} \leftarrow_{\mathrm{R}} \mathsf{FE.KeyGen}(\mathsf{fmsk}, \mathsf{f})$
  $\mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|, 2\lambda+|x|)}$
  $z \leftarrow_{\mathrm{R}} \{0,1\}^\lambda$
  $Z \leftarrow \mathsf{PRG}(z)$
  $\mathsf{PC} = \mathsf{iO}(\mathsf{ProgProv}_{i,a^*,\mathsf{crs}})$
  $\mathsf{CRS} := (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})$
  Return $\mathsf{CRS}$

$\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}(x, w, r)$
  if $(x, w) \notin R$
    Return $\perp$
  $a \leftarrow_{\mathrm{R}} \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$
  if $a < a^*$ then
    $(C, \pi) = \mathsf{HidingProof}_{\mathsf{crs}}(x, w, a)$
  if $a = a^*$
    $(C, \pi) = \mathsf{HybridProof}_{i,a^*,\mathsf{crs}}(x, w, a)$
  if $a > a^*$
    $(C, \pi) = \mathsf{BindingProof}_{\mathsf{crs}}(x, a)$
  Return $\Pi := (C, \pi)$
```

Fig. 8. Hybrids $\mathsf{H}_{(i,a^*)}$ for the proofs of Theorems 19 and 20. Note that the Prover, Verifier, BindingProof, HidingProof and function f are the same as defined in Figure 5 and are not represented again for succinctness. For $i = 1$, subprogram $\mathsf{HybridProof}_{1,a^*,\mathsf{crs}} = \mathsf{BindingProof}_{\mathsf{crs}}$ and for $i = 15$, $\mathsf{HybridProof}_{15,a^*,\mathsf{crs}} = \mathsf{HidingProof}_{\mathsf{crs}}$. All $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}(x, w, r)$ are padded so that they have equal sizes.

This puncturing can be done since a^* is a parameter of the hybrid (we are enumerating over all values of a). Since the programs are functionally equivalent, this change is computationally indistinguishable by the security of iO. Observe that when we hardcode a value in a subprogram $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, it is understood that this value is also hardcoded in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$. A full description of $\mathsf{HybridProof}_{2,a^*,\mathsf{crs}}$ can be found in Figure 9. This shows the following lemma:

Lemma 21 (From $\mathsf{H}_{(1,a^*)}$ to $\mathsf{H}_{(2,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:* $|\mathsf{Adv}_{(1,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(2,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B})$.

Hybrid $\mathsf{H}_{(3,a^*)}$ Here subprogram $\mathsf{HybridProof}_{3,a^*,\mathsf{crs}}$ is changed so that r_1^* is now a uniformly random value hardcoded inside our program. This change is computationally indistinguishable by the pseudorandomness at punctured points of PRF (we are replacing the evaluation at $K_1\{a^*\}$ by a uniformly random). A full description of subprogram $\mathsf{HybridProof}_{3,a^*,\mathsf{crs}}$ can be found in Figure 9. This shows the following lemma:

Lemma 22 (From $\mathsf{H}_{(2,a^*)}$ to $\mathsf{H}_{(3,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:* $|\mathsf{Adv}_{(2,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(3,a^*)}(\mathcal{A})| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B})$.

Hybrid $\mathsf{H}_{(4,a^*)}$ Subprogram $\mathsf{HybridProof}_{2,a^*,\mathsf{crs}}$ is changed so that key K_2 is punctured at point a^*. This is by the same argument as in Lemma 21 and uses the security of iO. Once we puncture the key, only $K_2\{a^*\}$ is hardcoded in all subroutines of $\mathsf{ProgProv}_{4,a^*,\mathsf{crs}}$, along with the evaluation of $r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$, but not K_2 itself. This shows the following lemma:

Lemma 23 (From $\mathsf{H}_{(3,a^*)}$ to $\mathsf{H}_{(4,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:* $|\mathsf{Adv}_{(3,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(4,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B})$.

Hybrid $\mathsf{H}_{(5,a^*)}$ Here subprogram $\mathsf{HybridProof}_{5,a^*,\mathsf{crs}}$ is changed so that r_2^* is now a uniformly random value hardcoded inside our program. This change is computationally indistinguishable by the pseudorandomness at punctured points of PRF (we are replacing the evaluation at $K_2\{a^*\}$ by a uniformly random). The full description of $\mathsf{HybridProof}_{5,a^*,\mathsf{crs}}$ can be found in Figure 10. This shows the following lemma:

Lemma 24 (From $\mathsf{H}_{(4,a^*)}$ to $\mathsf{H}_{(5,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:* $|\mathsf{Adv}_{(4,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(5,a^*)}(\mathcal{A})| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B})$.

```
$\mathsf{HybridProof}_{1,a^*,\mathsf{crs}}(x,w,a)$
    Hardcoded: Keys $K_1, K_2, K_3, z$
    $X \leftarrow \mathsf{PRF}(K_1, a)$
    $\mathsf{hrs} \leftarrow \mathsf{H}(X) \oplus \mathsf{crs}$
    $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
    $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
    $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (X, \mathcal{I}, 0, 0); r_2)$
    Return $\mathit{\Pi} := (C, \pi)$
```

```
$\mathsf{HybridProof}_{2,a^*,\mathsf{crs}}(x,w,a)$
    Hardcoded: Keys $K_1\{a^*\}$, $K_2, K_3, z$
                $r_1^* \leftarrow \mathsf{PRF}(K_1, a^*)$
    $\mathsf{hrs} \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$
    $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
    $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
    $C = \mathsf{FE.Enc}(\mathsf{fmpk}, ( r_1^* , \mathcal{I}, 0, 0); r_2)$
    Return $\mathit{\Pi} := (C, \pi)$
```

```
$\mathsf{HybridProof}_{3,a^*,\mathsf{crs}}(x,w,a)$
    Hardcoded: Keys $K_1\{a^*\}, K_2, K_3, z$
                $r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$
    $\mathsf{hrs} \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$
    $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
    $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
    $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2)$
    Return $\mathit{\Pi} := (C, \pi)$
```

```
$\mathsf{HybridProof}_{4,a^*,\mathsf{crs}}(x,w,a)$
    Hardcoded: Keys $K_1\{a^*\}$, $K_2\{a^*\}$ $, K_3, z$
                $r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$
                $r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$
    $\mathsf{hrs} \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$
    $(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
    $r_2 \leftarrow \mathsf{PRF}(K_2, a)$
    $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2)$
    Return $\mathit{\Pi} := (C, \pi)$
```

Fig. 9. Descriptions of $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, for $i = 1 \ldots 4$. In each subprogram, the changes relative to the previous subprogram are highlighted in gray. When we hardcode a value in a subprogram $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, it is understood that this value is also hardcoded in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$. If a key K is punctured in $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, we understand that it is punctured in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$ and all its subprograms as well. Note that $\mathsf{HybridProof}_{1,a^*,\mathsf{crs}}$ is the same as $\mathsf{BindingProof}_{\mathsf{crs}}$.

Hybrid $\mathsf{H}_{(6,a^*)}$ Subprogram $\mathsf{HybridProof}_{6,a^*,\mathsf{crs}}$ precomputes and hardcodes the (C^*, π^*) corresponding to a^*. For this we make the crucial observation that for every a, there exists only one corresponding (x, w). This follows from the perfect correctness of the lossy encryption scheme LE, because LE is in injective mode and because $a = \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$. To compute this hybrid, we use lsk to decrypt a^* and obtain the corresponding (x^*, w^*). Thus, if a^* is known in advance this means (x^*, w^*) is also known in advance. Since crs is a parameter of the circuit and also known in advance, we can compute $\mathsf{hrs}^* \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$, $(\pi^*, \mathcal{I}^*) \leftarrow \mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$ and $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}^*, 0, 0); r_2^*)$. We hardcode (C^*, π^*) and these are also the returned values when $\mathsf{HybridProof}_{6,a^*,\mathsf{crs}}$ is invoked on (x^*, w^*, a^*). Since $\mathsf{ProgProv}_{6,a^*,\mathsf{crs}}$ is functionally equivalent to $\mathsf{ProgProv}_{5,a^*,\mathsf{crs}}$, this step is justified by iO security. The full description of $\mathsf{HybridProof}_{6,a^*,\mathsf{crs}}$ can be found in Figure 10. From all the above, we have the following lemma:

Lemma 25 (From $\mathsf{H}_{(5,a^*)}$ to $\mathsf{H}_{(6,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:* $|\mathsf{Adv}_{(5,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(6,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B})$.

Hybrid $\mathsf{H}_{(7,a^*)}$ To obtain subprogram $\mathsf{HybridProof}_{7,a^*,\mathsf{crs}}$, we use the selective security of the functional encryption scheme FE to switch ciphertext $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}^*, 0, 0); r_2^*)$ to ciphertext:

$$C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2^*)$$

We argue that these two ciphertexts are indistinguishable. Consider decryption key $\mathsf{sk_f}$ used by the verifier, this key is associated to function f. But from the definition of f, it holds that:

$$\mathsf{f}(r_1^*, \mathcal{I}^*, 0, 0) = \mathsf{f}(0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}).$$

Since r_2^* used for encryption has been previously switched to a uniformly random, we can therefore reduce the gap between these two games to the $\mathsf{SEL\text{-}IND\text{-}FE\text{-}CPA}$ game. Also note that we are only able to use the selective security of the FE scheme because all the values above are known in advance and are derived from a. The full description of $\mathsf{HybridProof}_{7,a^*,\mathsf{crs}}$ can be found in Figure 10. We have therefore proven the following lemma:

```
$\mathsf{HybridProof}_{5,a^*,\mathsf{crs}}(x,w,a)$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
                $r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$
                $r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
$\mathsf{hrs} \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$
$(\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2)$
Return $\Pi := (C, \pi)$
```

```
$\mathsf{HybridProof}_{6,a^*,\mathsf{crs}}(x,w,a)$
Precompute: $r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$
    $r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
    $\mathsf{hrs}^* \leftarrow \mathsf{H}(r_1^*) \oplus \mathsf{crs}$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
    $(\pi^*, \mathcal{I}^*) \leftarrow \mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$
    $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}^*, 0, 0); r_2^*)$
Return $\Pi := ( C^* , \pi^* )$
```

```
$\mathsf{HybridProof}_{7,a^*,\mathsf{crs}}(x,w,a)$
Precompute: $r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$
    $r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
    $T^* \leftarrow \mathsf{H}(r_1^*)$
    $\mathsf{hrs}^* \leftarrow T^* \oplus \mathsf{crs}$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
    $(\pi^*, \mathcal{I}^*) \leftarrow \mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$
    $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z , T^*_{\mathcal{I}^*} ); r_2^*)$
Return $\Pi := (C^*, \pi^*)$
```

```
$\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}(x,w,a)$
Precompute: $r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
    $T^* \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|,2\lambda+|x|)}$
    $\mathsf{hrs}^* \leftarrow T^* \oplus \mathsf{crs}$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
    $(\pi^*, \mathcal{I}^*) \leftarrow \mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$
    $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2^*)$
Return $\Pi := (C^*, \pi^*)$
```

Fig. 10. Descriptions of $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, for $i = 5 \ldots 8$. In each subprogram, the changes relative to the previous subprogram are highlighted in gray. A hardwired value is understood to also be hardwired in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$. If key K is punctured, we understand that it is punctured in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$ and all its subprograms.

Lemma 26 (From $\mathsf{H}_{(6,a^*)}$ to $\mathsf{H}_{(7,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(6,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(7,a^*)}(\mathcal{A})| \le \mathsf{Adv}^{\mathsf{FE}}_{\mathsf{Exp}\text{-}s\text{-}\mathsf{IND}\text{-}\mathsf{FE}\text{-}\mathsf{CPA}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(8,a^*)}$ Subprogram $\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}$ is defined like $\mathsf{HybridProof}_{7,a^*,\mathsf{crs}}$, except that the computation of hrs^* changes. Instead of computing $\mathsf{hrs}^* \leftarrow T^* \oplus \mathsf{crs}$, where $T^* \leftarrow \mathsf{H}(r_1^*)$, we compute $T^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(|x|,\lambda)}$ and let $\mathsf{hrs}^* \leftarrow T^* \oplus \mathsf{crs}$. This step is justified by the dense mode of H. From Definition 2, we know that for uniformly random r_1^*, we have $\mathsf{H}(r_1^*)$ statistically indistinguishable from a uniformly random. Moreover, by choosing the security parameter in $\mathsf{LF.Setup}$ (1^λ,dense) to be large enough, we can offset the $2^{p(|x|+\lambda)}$ factor coming from enumerating over all values of a. The full description of $\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}$ can be found in Figure 10. We have therefore proven the following lemma:

Lemma 27 (From $\mathsf{H}_{(7,a^*)}$ to $\mathsf{H}_{(8,a^*)}$). *For every (potentially unbounded) adversary $\mathcal{A}$, it holds that:*

$$|\mathsf{Adv}_{(7,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(8,a^*)}(\mathcal{A})| \le \tfrac{1}{2^{p(|x|+\lambda)+\lambda}}.$$

Hybrid $\mathsf{H}_{(9,a^*)}$ In this hybrid, we use the zero-knowledge property of the hidden-bits NIZK system to replace real proofs by simulated ones. Subprogram $\mathsf{HybridProof}_{9,a^*,\mathsf{crs}}$ is defined like $\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}$, but now the precomputation of the program involves choosing a uniformly random $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(|x|,\lambda)}$. Polynomial $p_3(|x|,\lambda)$ represents the size of the random tape needed by the hidden-bits simulator S_H. Proofs are now simulated, i.e. $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$

We now argue that this hybrid is statistically indistinguishable from the previous one. The reason this works is that we already used FE security to ensure that only the revealed bits of the $\mathsf{hrs}^*_{I^*}$ are encoded in ciphertext C^* and also that hrs^* is uniformly random. This, coupled with the fact that in $\mathsf{H}_{(9,a^*)}$ only the value of the real proof (C^*, π^*) is hardcoded means we can use the ZK property of NIZK_H. In $\mathsf{HybridProof}_{9,a^*,\mathsf{crs}}$ we can hardcode only the simulated proof, and there is no need to include the simulator code in $\mathsf{ProgProv}_{9,a^*,\mathsf{crs}}$.

The full description of $\mathsf{HybridProof}_{9,a^*,\mathsf{crs}}$ can be found in Figure 11. We have the following lemma, which we prove in detail in Appendix B:

```
$\mathsf{HybridProof}_{9,a^*,\mathsf{crs}}(x, w, a)$
Precompute: $r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
        $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(|x|,\lambda)}$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
        $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$
        $T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
        $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2^*)$
Return $\varPi := (C^*, \pi^*)$
```

```
$\mathsf{HybridProof}_{10,a^*,\mathsf{crs}}(x, w, a)$
Precompute: $r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$
        $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(|x|,\lambda)}$
Hardcoded: Keys $K_1\{a^*\}, K_2\{a^*\}, K_3, z$
        $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$
        $T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
        $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2^*)$
Return $\varPi := (C^*, \pi^*)$
```

```
$\mathsf{HybridProof}_{11,a^*,\mathsf{crs}}(x, w, a)$
Precompute: $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(|x|,\lambda)}$
Hardcoded: Keys $K_1\{a^*\}, K_2, K_3, z$
        $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$
        $T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
$r_2 \leftarrow \mathsf{PRF}(K_2, a)$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2)$
Return $\varPi := (C, \pi^*)$
```

```
$\mathsf{HybridProof}_{12,a^*,\mathsf{crs}}(x, w, a)$
Precompute: $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(|x|,\lambda)}$
Hardcoded: Keys $K_1\{a^*\}, K_2, K_3\{a^*\}, z$
        $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$
        $T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
$r_2 \leftarrow \mathsf{PRF}(K_2, a)$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2)$
Return $\varPi := (C, \pi^*)$
```

Fig. 11. Descriptions of $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, for $i = 9 \ldots 12$. In each subprogram, the changes relative to the previous subprogram are highlighted in gray. A hardwired value is also hardwired in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$. If key K is punctured, we understand that it is punctured in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$ and all its subprograms.

Lemma 28 (From $\mathsf{H}_{(8,a^*)}$ to $\mathsf{H}_{(9,a^*)}$). *Let $a^* = \mathsf{LE.Enc}(\mathsf{lpk}, (x^*, w^*); r)$. Then it holds that either:*

1. *if $(x^*, w^*) \in R$, then $\mathsf{H}_{(8,a^*)}$ and $\mathsf{H}_{(9,a^*)}$ are statistically close. Namely, for every (potentially unbounded) adversary $\mathcal{A}$,*

$$|\mathsf{Adv}_{(8,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(9,a^*)}(\mathcal{A})| \leq \varDelta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda).$$

2. *if $(x^*, w^*) \notin R$, then $\mathsf{H}_{(8,a^*)}$ and $\mathsf{H}_{(9,a^*)}$ are computationally indistinguishable. Namely, for every* PPT *adversary $\mathcal{A}$, there exists* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(8,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(9,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(10,a^*)}$ In subprogram $\mathsf{HybridProof}_{10,a^*,\mathsf{crs}}$, the only change made is that r_2^* is changed from a uniformly random value (as in hybrid $\mathsf{H}_{(9,a^*)}$) to $r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$. This change is justified by the pseudorandomness of $\mathsf{PRF}(K_2, \cdot)$ at punctured point a^*. The full description of $\mathsf{HybridProof}_{10,a^*,\mathsf{crs}}$ can be found in Figure 11. This shows the following lemma:

Lemma 29 (From $\mathsf{H}_{(9,a^*)}$ to $\mathsf{H}_{(10,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(9,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(10,a^*)}(\mathcal{A})| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(11,a^*)}$ In subprogram $\mathsf{HybridProof}_{11,a^*,\mathsf{crs}}$, the only change made is that r_2 is not precomputed anymore (as in hybrid $\mathsf{H}_{(10,a^*)}$).

Value $r_2 \leftarrow \mathsf{PRF}(K_2, a^*)$ is now compted on the fly. This means C must also be computed on the fly in this hybrid. These changes are justified by the fact that the two programs are functionally equivalent and thus their obfuscations computationally indistinguishable. The full description of $\mathsf{HybridProof}_{11,a^*,\mathsf{crs}}$ can be found in Figure 11. This shows the following lemma:

Lemma 30 (From $\mathsf{H}_{(10,a^*)}$ to $\mathsf{H}_{(11,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(10,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(11,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

```
$\mathsf{HybridProof}_{13,a^*,\mathsf{crs}}(x, w, a)$
$\mathsf{Precompute}$: $r_3^* \leftarrow \mathsf{PRF}(K_3, a^*)$
$\mathsf{Hardcoded}$: Keys $K_1\{a^*\}, K_2, K_3\{a^*\}, z$
    $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$
    $T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
$r_2 \leftarrow \mathsf{PRF}(K_2, a^*)$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r_2)$
Return $\Pi := (C, \pi^*)$
```

```
$\mathsf{HybridProof}_{14,a^*,\mathsf{crs}}(x, w, a)$
$\mathsf{Hardcoded}$: Keys $K_1\{a^*\}, K_2, K_3, z$
$r_3 \leftarrow \mathsf{PRF}(K_3, a)$
$(\mathsf{hrs}_I, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$
$T \leftarrow \mathsf{hrs}_I \oplus \mathsf{crs}_{\mathcal{I}}$
$r_2 \leftarrow \mathsf{PRF}(K_2, a)$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}, z, T_{\mathcal{I}}); r_2)$
Return $\Pi := (C, \pi)$
```

```
$\mathsf{HybridProof}_{15,a^*,\mathsf{crs}}(x, w, a)$
$\mathsf{Hardcoded}$: Keys $K_1, K_2, K_3, z$
$r_3 \leftarrow \mathsf{PRF}(K_3, a)$
$(\mathsf{hrs}_I, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$
$T \leftarrow \mathsf{hrs}_I \oplus \mathsf{crs}_{\mathcal{I}}$
$r_2 \leftarrow \mathsf{PRF}(K_2, a)$
$C = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}, z, T_{\mathcal{I}}); r_2)$
Return $\Pi := (C, \pi)$
```

Fig. 12. Descriptions of $\mathsf{HybridProof}_{i,a^*,\mathsf{crs}}$, for $i = 13 \dots 16$. In each subprogram, the changes relative to the previous subprogram are highlighted in gray. A hardwired value is also hardwired in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$. If key K is punctured, we understand that it is punctured in $\mathsf{ProgProv}_{i,a^*,\mathsf{crs}}$ and all its subprograms.

Hybrid $\mathsf{H}_{(12,a^*)}$ In subprogram $\mathsf{HybridProof}_{12,a^*,\mathsf{crs}}$, we puncture key K_3 at $K_3\{a^*\}$ and only hardcode this punctured key in our programs. This change is justified by the fact that the two programs are functionally equivalent and thus their obfuscations computationally indistinguishable. The full description of $\mathsf{HybridProof}_{12,a^*,\mathsf{crs}}$ is given in Figure 11. This shows the following lemma:

Lemma 31 (From $\mathsf{H}_{(11,a^*)}$ to $\mathsf{H}_{(12,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(10,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(11,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(13,a^*)}$ Subprogram $\mathsf{HybridProof}_{13,a^*,\mathsf{crs}}$ is changed so that r_3^* is not a hard-wired uniformly random value anymore, but is chosen as $r_3^* \leftarrow \mathsf{PRF}(K_3, a^*)$. This change is justified by the pseudo-randomness of $\mathsf{PRF}(K_3, \cdot)$ at punctured point a^*. The full description of $\mathsf{HybridProof}_{13,a^*,\mathsf{crs}}$ is given in Figure 12. From the above, we have shown the following lemma:

Lemma 32 (From $\mathsf{H}_{(12,a^*)}$ to $\mathsf{H}_{(13,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(12,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(13,a^*)}(\mathcal{A})| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(14,a^*)}$ In subprogram $\mathsf{HybridProof}_{14,a^*,\mathsf{crs}}$ the key K_3 is not punctured anymore at a^*. This means that $r_3 \leftarrow \mathsf{PRF}(K_3, a)$ is not hardwired anymore. As a consequence, the simulated proofs are also not hardcoded. Since this program is functionally equivalent to $\mathsf{HybridProof}_{14,a^*,\mathsf{crs}}$, we justify this change by the security of iO. The full description of $\mathsf{HybridProof}_{14,a^*,\mathsf{crs}}$ is given in Figure 12. From all the above, we have shown the following lemma:

Lemma 33 (From $\mathsf{H}_{(13,a^*)}$ to $\mathsf{H}_{(14,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(13,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(14,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

Hybrid $\mathsf{H}_{(15,a^*)}$ In subprogram $\mathsf{HybridProof}_{15,a^*,\mathsf{crs}}$ the key K_1 is not punctured anymore at a^*. Key K_1 is not even used anymore in this subprogram, therefore this program is functionally equivalent to $\mathsf{HybridProof}_{14,a^*,\mathsf{crs}}$. We thus justify this change by the security of iO. The full description of $\mathsf{HybridProof}_{15,a^*,\mathsf{crs}}$ is given in Figure 12. Remark that $\mathsf{HybridProof}_{15,a^*,\mathsf{crs}}$ is the same as $\mathsf{HidingProof}_{\mathsf{crs}}$, which means $\mathsf{H}_{(15,a^*)} = \mathsf{H}_{(1,(a^*+1))}$. From all the above, we have:

Lemma 34 (From $\mathsf{H}_{(14,a^*)}$ to $\mathsf{H}_{(15,a^*)}$). *For every* PPT *adversary $\mathcal{A}$, there exists a* PPT *adversary $\mathcal{B}$, such that:*

$$|\mathsf{Adv}_{(14,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(15,a^*)}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

5 Acknowledgements

We would like to acknowledge the anonymous reviewer that pointed out a flaw in the proof of theorem 14 of a previous version of the paper. Both authors were supported by ERC PREP-CRYPTO Grant Agreement ID 724307.

References

1. Agrikola, T., Hofheinz, D.: Interactively secure groups from obfuscation. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 341–370. Springer, Heidelberg (Mar 2018)
2. Albrecht, M.R., Farshim, P., Hofheinz, D., Larraia, E., Paterson, K.G.: Multilinear maps from obfuscation. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 446–473. Springer, Heidelberg (Jan 2016)
3. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (Aug 2001)
4. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (Aug 2009)
5. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and impossibility results for encryption and commitment secure under selective opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (Apr 2009)
6. Bitansky, N., Paneth, O.: ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 401–427. Springer, Heidelberg (Mar 2015)
7. Bitansky, N., Paneth, O., Wichs, D.: Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 474–502. Springer, Heidelberg (Jan 2016)
8. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: Guruswami, V. (ed.) 56th FOCS. pp. 171–190. IEEE Computer Society Press (Oct 2015)
9. Blazy, O., Fuchsbauer, G., Izabachène, M., Jambert, A., Sibert, H., Vergnaud, D.: Batch Groth-Sahai. In: Zhou, J., Yung, M. (eds.) ACNS 10. LNCS, vol. 6123, pp. 218–235. Springer, Heidelberg (Jun 2010)
10. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th ACM STOC. pp. 103–112. ACM Press (May 1988)
11. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (Aug 2008)
12. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (Mar 2011)
13. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (Dec 2013)
14. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (Mar 2014)

15. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.: Fiat-shamir from simpler assumptions. IACR Cryptology ePrint Archive 2018, 1004 (2018)
16. Canetti, R., Chen, Y., Reyzin, L.: On the correlation intractability of obfuscated pseudorandom functions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 389–415. Springer, Heidelberg (Jan 2016)
17. Canetti, R., Lichtenberg, A.: Certifying trapdoor permutations, revisited. In: TCC 2018 (2018), appears. http://eprint.iacr.org/2017/631
18. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (Mar 2015)
19. Canetti, R., Lombardi, A., Wichs, D.: Non-interactive zero knowledge and correlation intractability from circular-secure FHE. Cryptology ePrint Archive, Report 2018/1248 (2018), http://eprint.iacr.org/2018/1248, http://eprint.iacr.org/
20. Escala, A., Groth, J.: Fine-tuning Groth-Sahai proofs. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 630–649. Springer, Heidelberg (Mar 2014)
21. Farshim, P., Hesse, J., Hofheinz, D., Larraia, E.: Graded encoding schemes from obfuscation. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 371–400. Springer, Heidelberg (Mar 2018)
22. Feige, U., Lapidot, D., Shamir, A.: Multiple noninteractive zero knowledge proofs under general assumptions. SIAM Journal on Computing 29(1), 1–28 (1999), https://doi.org/10.1137/S0097539792230010
23. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: 22nd ACM STOC. pp. 416–426. ACM Press (May 1990)
24. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987)
25. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More constructions of lossy and correlation-secure trapdoor functions. Journal of Cryptology 26(1), 39–74 (Jan 2013)
26. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS. pp. 40–49. IEEE Computer Society Press (Oct 2013)
27. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th FOCS. pp. 464–479. IEEE Computer Society Press (Oct 1984)
28. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the ACM 38(3), 691–729 (1991)
29. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: 17th ACM STOC. pp. 291–304. ACM Press (May 1985)
30. Goldwasser, S., Ostrovsky, R.: Invariant signatures and non-interactive zero-knowledge proofs are equivalent (extended abstract). In: Brickell, E.F. (ed.) CRYPTO'92. LNCS, vol. 740, pp. 228–245. Springer, Heidelberg (Aug 1993)
31. Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive zaps and new techniques for NIZK. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 97–111. Springer, Heidelberg (Aug 2006)
32. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (Apr 2008)
33. Hartung, G., Hoffmann, M., Nagel, M., Rupp, A.: BBA+: Improving the security and applicability of privacy-preserving point collection. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 17. pp. 1925–1942. ACM Press (Oct / Nov 2017)
34. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM Journal on Computing 28(4), 1364–1396 (1999)
35. Herold, G., Hesse, J., Hofheinz, D., Ràfols, C., Rupp, A.: Polynomial spaces: A new framework for composite-to-prime-order transformations. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 261–279. Springer, Heidelberg (Aug 2014)
36. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 13. pp. 669–684. ACM Press (Nov 2013)
37. Lindell, Y.: An efficient transform from sigma protocols to NIZK with a CRS and non-programmable random oracle. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 93–109. Springer, Heidelberg (Mar 2015)
38. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010), http://eprint.iacr.org/2010/556
39. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. IACR Cryptology ePrint Archive 2019, 158 (2019), https://eprint.iacr.org/2019/158

40. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (Aug 2008)
41. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 187–196. ACM Press (May 2008)
42. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th ACM STOC. pp. 475–484. ACM Press (May / Jun 2014)
43. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (May 2005)
44. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (Aug 1990)

A Summary of Theorem 19

Game	hrs	$(\pi, \mathcal{I})$	C	remark	justification
$\mathsf{H}_{(a^*,1)}$	$\mathsf{H}(\mathsf{PRF}(K_1, a)) \oplus \mathsf{crs}$	$\mathsf{P}_H(x, w, \mathsf{hrs})$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1, \mathcal{I}, 0, 0); r_2)$	$r_1 \leftarrow \mathsf{PRF}(K_1, a)$	$\mathsf{BindingProof}_{\mathsf{crs}}$
$\mathsf{H}_{(a^*,2)}$	$\mathsf{H}(r_1^*) \oplus \mathsf{crs}$	$\mathsf{P}_H(x, w, \mathsf{hrs})$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2)$	$K_1\{a^*\}$ punctured $r_1^* \leftarrow \mathsf{PRF}(K_1, a^*)$	iO security
$\mathsf{H}_{(a^*,3)}$	$\mathsf{H}(r_1^*) \oplus \mathsf{crs}$	$\mathsf{P}_H(x, w, \mathsf{hrs})$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2)$	$r_1^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_1(\lvert x\rvert, \lambda)}$	PRF security
$\mathsf{H}_{(a^*,4)}$	$\mathsf{H}(r_1^*) \oplus \mathsf{crs}$	$\mathsf{P}_H(x, w, \mathsf{hrs})$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2^*)$	$K_2\{a^*\}$ punctured $r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$	iO security
$\mathsf{H}_{(a^*,5)}$	$\mathsf{H}(r_1^*) \oplus \mathsf{crs}$	$\mathsf{P}_H(x, w, \mathsf{hrs})$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}, 0, 0); r_2^*)$	$r_2^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(\lvert x\rvert, \lambda)}$	PRF security
$\mathsf{H}_{(a^*,6)}$	$\mathsf{H}(r_1^*) \oplus \mathsf{crs}$	$\mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (r_1^*, \mathcal{I}^*, 0, 0); r_2^*)$		iO security
$\mathsf{H}_{(a^*,7)}$	$T^* \oplus \mathsf{crs}$	$\mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2^*)$	$T^* \leftarrow \mathsf{H}(r_1^*)$	SEL-IND-FE-CPA
$\mathsf{H}_{(a^*,8)}$	$T^* \oplus \mathsf{crs}$	$\mathsf{P}_H(x^*, w^*, \mathsf{hrs}^*)$	$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2^*)$	$T^* \leftarrow_{\mathrm{R}} \{0,1\}^{t(\lvert x\rvert, 2\lambda+\lvert x\rvert)}$	Statistical step
$\mathsf{H}_{(a^*,9)}$	$(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2^*)$	$T^* \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$ $r_3^* \leftarrow_{\mathrm{R}} \{0,1\}^{p_3(\lvert x\rvert, \lambda)}$	ZK of NIZK_H
$\mathsf{H}_{(a^*,10)}$	$(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2^*)$	$r_2^* \leftarrow \mathsf{PRF}(K_2, a^*)$	PRF security
$\mathsf{H}_{(a^*,11)}$	$(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2)$	K_2 unpunctured $r_2 \leftarrow \mathsf{PRF}(K_2, a^*)$	iO security
$\mathsf{H}_{(a^*,12)}$	$(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2)$	$K_3\{a^*\}$ punctured	iO security
$\mathsf{H}_{(a^*,13)}$	$(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*) \leftarrow \mathsf{S}_H(x^*; r_3^*)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I^*, z, T^*_{\mathcal{I}^*}); r_2)$	$r_3^* \leftarrow \mathsf{PRF}(K_3, a^*)$	PRF security
$\mathsf{H}_{(a^*,14)}$	$(\mathsf{hrs}_I, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I, z, T_{\mathcal{I}}); r_2)$	K_3 unpunctured $r_3 \leftarrow \mathsf{PRF}(K_3, a)$	iO security
$\mathsf{H}_{(a^*,15)}$	$(\mathsf{hrs}_I, \pi, \mathcal{I}) \leftarrow \mathsf{S}_H(x; r_3)$		$\mathsf{FE.Enc}(\mathsf{fmpk}, (0, I, z, T_{\mathcal{I}}); r_2)$	K_1 unpunctured	iO security $\mathsf{HidingProof}_{\mathsf{crs}}$

Fig. 13. An overview of the games used in the proof of Theorem 20. Changes between consecutive hybrids are highlighted in light gray. Starred terms represent values hardwired in our programs.

B Proof of Lemma 28

Lemma 27 (From $\mathsf{H}_{(8,a^*)}$ to $\mathsf{H}_{(9,a^*)}$) Let $a^* = \mathsf{LE.Enc}(\mathsf{lpk}, (x^*, w^*); r)$. Then it holds that either:

1. if $(x^*, w^*) \in R$, then $\mathsf{H}_{(8,a^*)}$ and $\mathsf{H}_{(9,a^*)}$ are statistically close. Namely, for every (potentially unbounded) adversary $\mathcal{A}$,
$$|\mathsf{Adv}_{(8,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(9,a^*)}(\mathcal{A})| \le \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda).$$
2. if $(x^*, w^*) \notin R$, then $\mathsf{H}_{(8,a^*)}$ and $\mathsf{H}_{(9,a^*)}$ are computationally indistinguishable. Namely that for every PPT adversary $\mathcal{A}$, there exists PPT adversary $\mathcal{B}$, such that:
$$|\mathsf{Adv}_{(8,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(9,a^*)}(\mathcal{A})| \le \mathsf{Adv}^{\mathsf{iO}}(\kappa, \mathcal{B}).$$

Proof. Recall that because LE is in injective mode, a^* corresponds to a single (x^*, w^*). The lossy encryption keys $(\mathsf{lpk}, \mathsf{lsk})$ are chosen on the fly by hybrid games $\mathsf{H}_{(8,a^*)}$ and $\mathsf{H}_{(9,a^*)}$. Depending on the keys chosen,

a^* can decrypt to either $(x^*, w^*) \in R$ or $(x^*, w^*) \notin R$, therefore we need to show that the hybrids are indistinguishable in both cases:

Case 1: $(\mathbf{x}^*, \mathbf{w}^*) \notin \mathbf{R}$. Making any modifications to $\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}$ does not change the functionality of $\mathsf{ProgProv}_{8,a^*,\mathsf{crs}}$, as this program outputs $\perp$ without ever executing $\mathsf{HybridProof}_{8,a^*,\mathsf{crs}}$. Therefore, the hybrids are computationally indistinguishable.

Case 2: $(\mathbf{x}^*, \mathbf{w}^*) \in \mathbf{R}$. The zero-knowledge property of NIZK_H (see Definition 11) says that for all $(x, w) \in R$, we have that $\Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda) \coloneqq \Delta(E_0(x, w), E_1(x, w))$ is negligible, where the E_0 and E_1 are defined as:

$$E_0(x, w) \coloneqq \{(\mathsf{hrs}_{\mathcal{I}}, \pi, \mathcal{I}) : \mathsf{hrs} \leftarrow \{0,1\}^{t(|x|,\lambda)}, (\pi, \mathcal{I}) \leftarrow \mathsf{P}_H(x, w, \mathsf{hrs})\}$$

$$E_1(x, w) \coloneqq \{\mathsf{S}_H(x)\}$$

Now let E'_0 be the distribution of the CRS in hybrid game $\mathsf{H}_{(8,a^*)}$ and E'_1 be the distribution of the CRS in hybrid game $\mathsf{H}_{(9,a^*)}$. We show that there exists a probabilistic polynomial-time function F, such that F^{E_b} outputs the distribution E'_b without knowing bit b (Notation F^{E_b} means that F has oracle access to $E_b(x, w)$). Then since $E_0(x, w)$ and $E_1(x, w)$ are close for all values of $(x, w) \in R$, it will necessarily follow that $\Delta(F^{E_0}, F^{E_1}) = \Delta(E'_0, E'_1) \le \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda)$.

Firstly, function F generates dense H, then PRG, $(\mathsf{fmpk}, \mathsf{fmsk})$, $\mathsf{sk_f}$, crs, z and Z. It continues by generating and puncturing $K_1\{a^*\}, K_2\{a^*\}, K_3$. Then F chooses $(\mathsf{lpk}, \mathsf{lsk}) \leftarrow_{\mathrm{R}} \mathsf{LE.Setup}(1^\lambda, \mathrm{inj})$. Remark that this choice of $(\mathsf{lpk}, \mathsf{lsk})$ determines what (x^*, w^*) is encoded in a^*, if any.

Secondly, now that (x^*, w^*) is known to F, it will make an oracle call to $E_b(x^*, w^*)$ and obtain a sample $\Pi^* = (\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*)$. It then draws $r_2 \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$ and computes $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r^*_2)$. Finally, it uses (Π^*, C^*) to construct and obfuscate a program we call $\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}(x, w, r)$ to obtain $\mathsf{PC} \coloneqq \mathsf{iO}(\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}})$. Finally, F returns $\mathsf{CRS} = (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})$. Function F^{E_b} and $\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}$ are described in Figure 14.

```
Function $F$
  $\mathsf{PRG} \leftarrow_{\mathrm{R}} \mathsf{PRG.Setup}(1^\lambda)$
  $\mathsf{H} \leftarrow_{\mathrm{R}} \mathsf{LF.Setup}(1^\lambda, \mathrm{dense})$
  $(\mathsf{lpk}, \mathsf{lsk}) \leftarrow_{\mathrm{R}} \mathsf{LE.Setup}(1^\lambda, \mathrm{inj})$
  $K_1, K_2, K_3 \leftarrow_{\mathrm{R}} \mathsf{PRF.KeyGen}(1^\kappa)$
  $(\mathsf{fmpk}, \mathsf{fmsk}) \leftarrow_{\mathrm{R}} \mathsf{FE.Setup}(1^\kappa)$
  $\mathsf{sk_f} \leftarrow_{\mathrm{R}} \mathsf{FE.KeyGen}(\mathsf{fmsk}, \mathsf{f})$
  $\mathsf{crs} \leftarrow_{\mathrm{R}} \{0,1\}^{t(|x|, 2\lambda + |x|)}$
  $z \leftarrow_{\mathrm{R}} \{0,1\}^\lambda, Z \leftarrow \mathsf{PRG}(z)$
  $(x^*, w^*) \leftarrow \mathsf{LE.Dec}(a^*)$
  Oracle call to $E_b(x^*, w^*)$
      Obtain $(\mathsf{hrs}^*_{I^*}, \pi^*, \mathcal{I}^*)$
  $T^*_{\mathcal{I}^*} \leftarrow \mathsf{hrs}^*_{I^*} \oplus \mathsf{crs}_{\mathcal{I}^*}$
  $r_2 \leftarrow_{\mathrm{R}} \{0,1\}^{p_2(|x|,\lambda)}$
  $C^* = \mathsf{FE.Enc}(\mathsf{fmpk}, (0, \mathcal{I}^*, z, T^*_{\mathcal{I}^*}); r^*_2)$
  $\mathsf{PC} = \mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}$
  $\mathsf{CRS} \coloneqq (\mathsf{H}, \mathsf{fmpk}, \mathsf{lpk}, \mathsf{sk_f}, \mathsf{crs}, Z, \mathsf{PC})$
```

```
$\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}(x, w, r)$
  if $(x, w) \notin R$
      Return $\perp$
  $a \leftarrow_{\mathrm{R}} \mathsf{LE.Enc}(\mathsf{lpk}, (x, w); r)$
  if $a < a^*$ then
      $(C, \pi) = \mathsf{HidingProof}_{\mathsf{crs}}(x, w, a)$
  if $a = a^*$
      Return $(C^*, \pi^*)$
  if $a > a^*$
      $(C, \pi) = \mathsf{BindingProof}_{\mathsf{crs}}(x, a)$
  Return $\Pi \coloneqq (C, \pi)$
```

Fig. 14. Function F and program $\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}$ used in the proof of Lemma 28. K_1 and K_2 are punctured at a^* in the subprograms of $\mathsf{ProgProv}^{89}$. Also, $\mathsf{ProgProv}^{89}_{a^*,\mathsf{crs}}(x, w, r)$, $\mathsf{ProgProv}_{8,a^*,\mathsf{crs}}(x, w, r)$ and $\mathsf{ProgProv}_{9,a^*,\mathsf{crs}}(x, w, r)$ are padded so that they have equal sizes.

To conclude, we have exhibited probabilistic polynomial-time function F, such that $F^{E_b} = E'_b$. Then since $\Delta(E_0, E_1) = \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda)$, it will necessarily follow that $\Delta(F^{E_0}, F^{E_1}) \le \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda)$. This means that the distributions of the CRS in $\mathsf{H}_{(8,a^*)}$ and in $\mathsf{H}_{(9,a^*)}$ are statistically closer than $\Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda)$. Therefore, we can conclude that:

$$|\mathsf{Adv}_{(8,a^*)}(\mathcal{A}) - \mathsf{Adv}_{(9,a^*)}(\mathcal{A})| \le \Delta^{\mathsf{NIZK}_H}_{\text{Zero Knowledge}}(\lambda).$$

```
Setup($1^\lambda$)
hrs $\leftarrow_{\mathrm{R}} \{0,1\}^{n^3 \times n^3 \times 5 \log_2 n}$
Return hrs

Prover(Graph $G$, Hamiltonian cycle Cycle, hrs)
$M$ = Parse(hrs)
if $M$ is "good" (contains Hamiltonian matrix $A$ and all its other entries are 0) then
  Reveal entries of $M$ which are not in $A$
  Using Cycle, compute $\pi$ such that $A \subseteq \pi(G)$
  Reveal $M_{i,j}$ that correspond to non-edges of $\pi(G)$
Return $\pi$

Verifier($\pi$, Revealed(hrs))
if $M$ is not completely revealed then
  For every non-edge $G_{i,j} = 0$, check if $M_{i,j}$ is revealed and equal to 0.
if $M$ is completely revealed then
  check that $M$ is indeed not "good". Reject if $M$ good.
Accept if all checks are succesful.

Parse(hrs)
"Parse hrs as a $n^3 \times n^3$ Boolean matrix $M$, with $\Pr[M_{i,j} = 1] = \frac{1}{n^5}, \forall i,j \in [n^3]$"
Partition hrs into blocks $B_{i,j}$ of size $5 \log_2 n$
if $B_{i,j}$ is all 1s, then $M_{i,j} = 1$, else $M_{i,j} = 0$
Return $M$
```

Fig. 15. Hidden-Bits Non-Interactive ZK scheme NIZK_H, due to [22]. By n we denote the size of G.

C Hidden Bits NIZK

In this appendix, we briefly go over the hidden-bits NIZK construction from [22]. The scheme is described in Figure 15. This NIZK system computes proofs that graphs G contain a Hamiltonian cycle, without revealing any information about the Hamiltonian cycle itself. Since Hamiltonian Cycle is an NP-complete language, this proof system can be used to prove any statement in NP. Briefly, the way the scheme works is the following: the hidden string hrs is interpreted as a matrix M which will contain a Hamiltonian matrix A with high probability. M will usually be larger than A and when M contains only A and its other entries are set to 0, M is called a "good" matrix. A proof π is a permutation of the graph G such that $A \subseteq \pi(G)$. Such a proof can be computed in polynomial time if the prover knows a Hamiltonian cycle of the graph G. A prover sends π to the verifier and reveals all entries of A which do not correspond to edges in $\pi(G)$.

The verifier knows that with high probability, M is good, i.e. there exists Hamiltonian matrix A embedded in hrs. And if the verifier trusts that this is the case, then the fact that $A \subseteq \pi(G)$ means that indeed G contains a Hamiltonian cycle. Intuitively, this leaks no information about the actual cycle of G, as only the zeros of matrix A corresponding to non-edges of G are revealed.

This hidden-bits NIZK scheme satisfies statistical soundness and perfect zero-knowledge (note that it is impossible to obtain both statistical soundness and perfect ZK in the standard model, but it is possible to do this in the hidden-bits model).

Lemma 35 ([22]: Probability that M is "good" (contains only a Hamiltonian matrix and nothing else)). *Let* hrs $\leftarrow_{\mathrm{R}} \{0,1\}^{n^3 \times n^3 \times 5 \log_2 n}$ *and* $M =$ Parse(hrs) *(see Figure 15). For sufficiently large n, the probability that M is good is greater than* $\frac{1}{dn\sqrt{n}}$ *for some constant d.*

Proof. Consider the probability that a certain row of M contains more than a single 1 (i.e. at least two ones). This happens with probability $\binom{n^3}{2} \times \frac{1}{(n^5)^2} < \frac{1}{n^4}$. Then with probability greater than $\frac{1}{n}$, every row has at most one entry set to 1.

Now, the bits of the hrs are unbiased and independent, and the the probability that $M_{i,j} = 1$ is $\frac{1}{n^5}$. Therefore, the probability that M has exactly n ones is $\binom{n^6}{n} \times \left(\frac{1}{n^5}\right)^n \times \left(1 - \frac{1}{n^5}\right)^{n^6-n} \approx \frac{n^{6n}}{n!} \times \frac{1}{n^{5n}} \times \left(1 - \frac{1}{n^5}\right)^{n^6-n} \approx \frac{n^n}{\sqrt{2\pi n}(\frac{n}{e})^n} \times \left(1 - \frac{1}{n^5}\right)^{n^6-n}$, which is approximately $\frac{e^n}{\sqrt{2\pi n}} \times \left(1 - \frac{1}{n^5}\right)^{n^6-n}$. By a series expansion of the second term, this is approximately $\frac{e^n}{\sqrt{2\pi n}} \times \left(1 - \frac{1}{n^5}\right)^{n^6-n} > \frac{1}{\sqrt{n}}$ for sufficiently large n. By the birthday paradox, with constant probability, each row and each column will have exactly one entry set to 1, so the probability that M contains a permutation matrix and has every other entry set to 0 is greater than $\frac{1}{d\sqrt{n}}$, for some constant d. Now, since there are $n!$ permutation matrices with n rows and n columns, and $(n-1)!$ of them are Hamiltonian, this means the probability that M contains a Hamiltonian matrix and is "good" is greater than $\frac{1}{dn\sqrt{n}}$.

Theorem 36. [22]*: The* NIZK *in the hidden-bits model from Figure 15 is perfectly zero-knowledge.*

Proof. Perfect Correctness: In this case, the prover knows a Hamiltonian cycle Cycle of G. When M is not "good", it will be completely revealed to the verifier, which trivially accepts. Otherwise, if M contains a Hamiltonian matrix A and its other entries are 0, the verifier's first check passes. Then the verifier checks if every non-edge of $\pi(G)$ corresponds to a revealed 0 in matrix A.

Statistical Soundness Now we suppose that G is not Hamiltonian. By Lemma 35, with probability at least $\frac{1}{dn\sqrt{n}}$, the matrix M is "good" (contains a $n \times n$ Hamiltonian submatrix A and has all its other entries set to 0). Then, the prover must reveal all entries not in the submatrix A since mapping $V(G) \times V(G)$ to any other $n \times n$ submatrix of M will reveal values of 1 in the rest of M. Therefore, the prover must output π such that the entries of $\pi(V(G)) \times \pi(V(G))$ correspond to entries of A. Moreover, each non-edge of G must be mapped to a 0 of A. This means that the 1s of A have preimages that correspond to edges in G, and since A is a Hamiltonian matrix, this induces a Hamiltonian cycle in G. We do not have perfect soundness, as when M is not "good", all entries are revealed and the proof is accepted for every x (verifier trivially accepts). Soundness can be amplified by increasing the size of the hrs to encode more than one matrix M. Namely, we amplify by parsing a larger hrs as ℓ matrices $M_1 \dots M_\ell$ and the prover outputs proof $\pi_i, \dots, \pi_\ell$. It is known that $\lim_{\alpha \to \infty} \left(1 - \frac{1}{\alpha}\right)^\alpha = \frac{1}{e}$. Consider some desired security parameter λ. Then if $\ell = n^2\sqrt{n}\lambda$ then with probability $(1 - \frac{1}{e^{n\lambda}})$ at least one M_i is a good matrix.

The verifier accepts if all checks pass for each (M_i, π_i), $i = 1 \dots \ell$. Then the probability that a non-Hamiltonian graph is accepted will be $\frac{1}{e^{n\lambda}}$. It is this protocol with amplified soundness the one we use for our construction, and which we denote as $(\mathsf{P}_H, \mathsf{V}_H)$ in our candidate from Figure 5.

Perfect Zero-Knowledge Now we are left to prove perfect zero-knowledge. In Figure 16, we exhibit an efficient simulator that briefly works as follows: on input graph G, it first chooses a permutation π, reveals the non-edges of $\pi(G)$ as zeroes and draws the positions corresponding to edges such that each entry is 1 with probability $\frac{1}{n^5}$, just as in the honest execution of the protocol. If M is "good", then the simulator outputs the corresponding proof.

We show that the distribution of revealed real hrs and real proofs is identical to the distribution of revealed simulated hrs and simulated proofs. In the real case, matrix A (which defines the 0/1 values of M) is randomly chosen with uniform distribution (among the $(n-1)!$ possibilities). Moreover, note that any two different Hamiltonian cycles A and A' determine two disjoint sets S_A and $S_{A'}$ of n permutations, where each permutation in S_A $(S_{A'})$ maps the Hamiltonian cycle of G onto $A(A')$. Therefore, for any permutation in $Sym(n)$, the probability that V receives it is $\frac{1}{n!}$. So the real proofs are distributed identically as the simulated proofs.

Now, consider the probability that the hidden matrix M contains only a Hamiltonian cycle, this is the same in both cases and the revealed portions of the hrs are distributed identically. Note that the distributions are identical only if G indeed contains a Hamiltonian cycle, but this is exactly where zero-knowledge must hold.

```
$\mathsf{Sim}(\text{Graph } G)$
Choose uniform permutation $\pi$
For every non-edge of $\pi(G)$
  Choose $r \leftarrow_{\mathrm{R}} \{0,1\}^{5\log_2 n} \setminus \{1\dots1\}$
  Reveal $r$ as a part of $\mathsf{hrs}$ and a 0 entry of $M$.
For unrevealed positions of $M$
  choose entry of $M$ by setting $\mathsf{hrs} \leftarrow_{\mathrm{R}} \{0,1\}^{5\log_2 n}$
if $M$ contains Hamiltonian cycle in unrevealed positions
  Return $\pi$
else reveal everything
```

Fig. 16. Simulator for the Hidden-Bits NIZK [22]. An alternative way to simulate would be to have matrix M chosen before π and then completely disregarded by revealing a zero for every non-edge of $\pi(G)$.

D Statistical extractability

Theorem 37. *When in binding mode, the* DM-NIWI *system in Figure 5 satisfies statistical extractability.*

Proof. This is a non-generic proof, we need to use fine-grained properties of the underlying hidden bits NIZK from [22], which is described in Figure 15, Appendix C. This proof could be made generic if we formalized some notion of extractability of the hidden-bits NIZK_H system. Nevertheless, this would be a non-standard notion and we prefer to give a non-generic proof instead.

Consider any $(x, \varPi = (C, \pi))$. Our goal is to show that for every $(\mathsf{CRS}, \mathsf{td}_{\mathsf{ext}}) \leftarrow_{\mathrm{R}} \mathsf{DM\text{-}NIWI.Setup}(1^\lambda, \text{binding})$, for all $w \leftarrow_{\mathrm{R}} \mathsf{DM\text{-}NIWI.Extract}(\mathsf{td}_{\mathsf{ext}}, x, \varPi)$, it holds that if $\mathsf{DM\text{-}NIWI.Verify}(\mathsf{CRS}, x, \varPi) = 1$, then w is a witness for x with overwhelming probability.

```
$\mathsf{DM\text{-}NIWI.Extract}(\mathsf{td}_{\mathsf{ext}} = \mathsf{fmsk}, \varPi = (C, \pi))$
  Decrypt $C = \mathsf{FE.Enc}(\mathsf{fmpk}, (X, \mathcal{I}, 0, 0); r_2)$ using $\mathsf{fmsk}$ to recover $X$.
  Compute $\mathsf{hrs} \leftarrow \mathsf{H}(X) \oplus \mathsf{crs}$
  Divide $\mathsf{hrs}$ into $\ell$ blocks $\mathsf{hrs}^1 \dots \mathsf{hrs}^\ell$
  $M_i \leftarrow \mathsf{Parse}(\mathsf{hrs}^i)$, for every $i \in [\ell]$
  Find block $i$ where $M_i$ is a good
  recover Hamiltonian matrix $A_i$ from $M_i$
  Use $\pi_i$ and $A_i$ to recover Hamiltonian Cycle $w$ in graph $x$
  Return $w$
```

Fig. 17. Algorithm DM-NIWI.Extract for the proof of Theorem 37. Algorithm Parse is from the hidden-bits NIZK and is decribed in Figure 15, Appendix C.

Recall that we are in binding mode. Just as in the soundness proof Theorem 14, we condition on the event E that Z does not have a PRG preimage, which happens with probability $1 - \frac{1}{2^{\lambda+|x|}}$. If Z has no preimage, then from the functionality of iO and the special correctness of the FE scheme, the adversary produces a ciphertext which decrypts in the same way as ciphertexts of the form $C = \mathsf{FE.Enc}(X, \mathcal{I}, \cdot, \cdot)$. Note that both the functional equivalence of iO and the correctness of the functional encryption scheme are statistical properties.

Then we deduce that decrypting C will indeed yield the entire X and we know that the verifier computes $\mathsf{hrs}_{\mathcal{I}} = \mathsf{H}(X) \oplus \mathsf{crs}_{\mathcal{I}}$. We want to prove that $\mathsf{V}_H(x, \mathsf{hrs}_{\mathcal{I}}, \mathcal{I}, \pi) = 1$ implies that the witness extracted by Extract is valid. Recall that H is a lossy function in lossy mode and it has only 2^k images. This means that $\mathsf{H}(x)$ can only influence k bits of the hidden-bits string hrs.

Let $|x| = n$. In Appendix C, we have shown that for an hrs of size $q(n, \lambda) = n^8\sqrt{n}\lambda$, the probability that hrs contains no good matrix is $\frac{1}{e^{n\lambda}}$. Then by choosing an hrs of size $(k+1) \cdot q(n, \lambda)$[7], we ensure that regardless of the choice of X, the probability that hrs contains no good matrix is $\frac{1}{e^{n\lambda}}$. Intuitively, this is because $\mathsf{H}(X)$ can influence only k bits of the hidden-bits string.

[7] Actually, the value of this polynomial can be greatly reduced, but we choose it as such for simplicity.

Let M_j be the good matrix in hrs and A_j the Hamiltonian matrix embedded inside it. Then this matrix cannot be revealed as a bad matrix, since then V_H will be able to detect that and reject. This means that every accepting proof (C, π) must contain a permutation π_j of the graph x such that $A \subseteq \pi_j(x)$. But then, since we know hrs entirely, we can invert π_j and compute a Hamiltonian cycle w of graph x.

E Comparison with other dual-mode NIZK constructions

Discussion: relation to Groth-Sahai. The only other known dual-mode proof system is due to Groth and Sahai [32]. Their scheme can be used to prove the satisfiability of (systems of) multivariate quadratic equations over cyclic groups.[8] In their scheme, a proof consists of a commitment com_w to a witness (i.e., a satisfying assignment) w, and helper information open that helps to recognize com_w as such. Specifically, their commitment scheme allows to homomorphically compute a commitment $\mathsf{com}_{f(w)}$ from com_w for any quadratic function f. Here, open simply contains an opening of the so-computed $\mathsf{com}_{f_i(w)}$ for any f_i for which $f_i(w) = 0$ shall be proved. A verifier can then compute $\mathsf{com}_{f_i(w)}$ from com_w and check that open indeed opens $\mathsf{com}_{f_i(w)}$ to 0.

If the used commitment is statistically binding, then the corresponding NIZK proof is statistically sound. Conversely, if the commitment is statistically hiding, then the NIZK system is statistically zero-knowledge.[9] Interestingly, the commitment scheme can be switched between binding and hiding in a computationally indistinguishable way, by tweaking its public parameters.

Our idea to switch the lossy encryption a of (x, w) between injective and lossy is superficially similar to this step. However, in our system, this switch helps to prove that openings of $\mathsf{hrs}_{\mathcal{I}}$ do not reveal anything beyond $\mathsf{hrs}_{\mathcal{I}}$. The actual switch between soundness and zero-knowledge happens when switching the function H, as described above. Hence, we do not view our system as an abstraction of Groth-Sahai proofs, but instead as a fundamentally different way to obtain dual-mode features.

Discussion: relation to Canetti, Lombardi and Wichs. Independently and concurrently to this work, [19] introduced a dual-mode NIZK for NP based on circular-secure FHE. We already know from [18] that FHE can be obtained from sub-exponentially secure iO, subexponentially secure one-way functions and lossy encryption (or rerandomizable encryption). We note, though, that the FHE scheme from [18] is not known to be circular-secure, therefore using it in the construction of [19] requires a somewhat nonstandard additional assumption to construct dual-mode NIZKs.

Discussion: relation to [16] and [15]. Another approach to obtaining a dual-mode NIZK is by combining the concurrent and independent work of [15] with the one of [16]. This yields a scheme from subexponentially-secure IO, sub-exponentially-secure one-way functions, lossy encryption and (polynomially-secure) virtual grey-box obfuscation (VGB) for evasive circuits.

[8] One interesting special case are multivariate quadratic equations over $\mathbb{Z}_p$ for prime p. The language of satisfiable (systems of) equations of this type is NP-complete.

[9] Technically, Groth and Sahai prove only statistical witness-indistinguishability, which can however be converted to (statistical) zero-knowledge in many cases.