Offline Witness Encryption

Hamza Abusalah, Georg Fuchsbauer, and Krzysztof Pietrzak *

IST Austria
{habusalah, gfuchsbauer, pietrzak}@ist.ac.at

Abstract. Witness encryption (WE) is an exciting new primitive introduced by Garg et al. (STOC 2013). WE is defined for some NP language L and allows to encrypt a message relative to an instance x so that one can decrypt with any w witnessing $x \in L$. Garg et al. construct WE for an NP-complete language from multilinear maps and give another construction from indistinguishability obfuscation (FOCS 2013). Due to the reliance on such heavy tools, WE can currently hardly be implemented on powerful hardware and will not be realizable on constrained devices like smart cards any time soon.
In this paper we construct a witness encryption scheme where *encryption* is a single Naor-Yung encryption (two CPA-encryptions and one NIZK proof showing the ciphertexts encrypt the same message), so encryption can even be done on a smart card. To achieve this, our scheme has a setup phase, which outputs public parameters containing an obfuscated circuit (only required for decryption), two public keys for a standard public-key encryption scheme and a common reference string for the NIZK (used for encryption). This setup phase need only be run once, and the parameters can be used for arbitrary many encryptions. Our scheme can easily be turned into a *functional* WE scheme, where a message is encrypted w.r.t. a statement and a function f, and using a witness w one learns $f(m, w)$.
Our construction and its proof are inspired by those of functional encryption by Garg et al. (FOCS 2013) and to prove (selective) security of our scheme we also assume indistinguishability obfuscation and statistically simulation-sound NIZK. We give a construction of the latter in bilinear groups and combining it with ElGamal encryption, our ciphertexts are of size 1.3 kB at a 128-bit security level.

Keywords: Witness Encryption, Indistinguishability Obfuscation, NIZK.

1 Introduction

Witness encryption. In an encryption scheme, the receiver needs to know some secret piece of information (the secret key) to decrypt. Garg, Gentry, Sahai and Waters [GGSW13] propose the intriguing new notion of *witness encryption* (WE). A WE scheme is defined for some NP language L with witness relation R: $L = \{x \mid R(x, w) = 1\}$. The encryption algorithm takes an instance x (instead of a public key) and a message m and produces a ciphertext c. In order to decrypt a ciphertext c, one needs a witness w such that $R(x, w) = 1$. Decryption is only possible if x is actually in the language and it is required that a ciphertext computed for some $x \notin L$ computationally hides the message m.

Applications. As shown in [GGSW13], from WE one can construct powerful cryptographic primitives such as identity-based encryption and even attribute-based encryption [SW05] for circuits. But WE also allows for applications that were not possible before; for example, one can encrypt a message with respect to a puzzle, such that only someone who found the solution for it can decrypt.[1] Another application is asymmetric password-based encryption [BH15], which allows hashed passwords (for any password-hashing function already in place) to be used as public encryption keys and passwords to decrypt.

Constructing WE. Garg et al. [GGSW13] construct a WE scheme for the NP-complete language "exact set cover", which implies WE for any language $L \in$ NP via polynomial-time many-one reductions (a.k.a. Levin reductions). The security of this construction is based on a strong assumption on "approximate" multilinear maps as constructed in [GGH13a]. Subsequently, a construction of WE from indistinguishability obfuscation (iO) was given in [GGH$^+$13b] and another one based

* Research supported by ERC starting grant (259668-PSPC)

[1] This puzzle can be any problem where solutions can be efficiently verified, like a crossword or Sudoku puzzle, or even the proof for some mathematical conjecture.

on multilinear maps in [GLW14]. The only candidate construction of iO is also based on the approximate multilinear maps from [GGH13a].

Implementing multilinear maps as required for iO or WE is computationally very expensive, but a first—though far from practical—implementation exists [AHKM14], and it is conceivable that algorithmic and hardware progress yield practical implementations in the not too distant future.

Offline witness encryption. Given that WE is not even practical on high-end machines, it seems foolish to hope for an implementation on low-end devices like smart cards. In this paper we show that it is possible to construct a WE scheme where *encryption* is very efficient, as the entire computationally hard work can be moved to a setup phase and—to a lesser extent—to the decryption process. This setup is either run by the sender *before* she knows the instance and the message for an encryption; or it is run by a trusted party once and for all and everyone can use the same parameters. The first case is reminiscent of online/offline encryption or signatures [EGM96], except that in our case, once generated, the parameters can be used for arbitrary many "online phases".

We call this concept *offline witness encryption* and define it as a tuple of three algorithms. The setup phase (which is not present in standard WE) takes as input only a security parameter 1^λ and outputs public parameters $(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$. To encrypt a message m for an instance x, one runs an encryption algorithm $c \leftarrow \mathsf{Enc}(x, m, \mathsf{pp_e})$. Such ciphertext c can then be decrypted given a witness w, i.e., for which $R(x, w) = 1$ holds, as $m = \mathsf{Dec}(c, w, \mathsf{pp_d})$. The goal of offline WE is to keep the parameters $\mathsf{pp_e}$ for encryption small and the Enc algorithm efficient.

Applications of offline WE. In any application of witness encryption its offline variant can be used to make encryption practically efficient, if one accepts an additional setup phase. However, for applications like IBE and attribute-based encryption, as discussed in [GGSW13], system-wide parameters must be set up by a trusted party anyway. This party could therefore simply also generate the offline-WE parameters, meaning encryption can be made efficient without requiring any additional trust.

Bellare and Tung Hoang [BH15] define and construct *asymmetric* password-based encryption (A-PBE), where a hash of a password can be used as a public key to encrypt messages, which can then be decrypted using the password. Unlike its symmetric counterpart, A-PBE remains secure even when the server storing hashed passwords is compromised. In particular, they show that if hashed passwords are already deployed using an already existing password-hashing function, witness encryption can be used to turn the hashed passwords into public keys.[2] The drawback of using WE is that both encryption and decryption are inefficient. Using offline WE where a trusted third party produces the system parameters in an offline phase, encryption can be made significantly more efficient, whereas decryption (and the one-time setup) remains inefficient.

The use of offline WE is therefore particularly appealing in scenarios where decryption is usually not done anyway, but ciphertexts are made public as a means of deterrent. Consider a scenario where a content provider lets subscribed users set up passwords and use them to access some content. The provider typically stores a hash of the password. In order to discourage subscribers from distributing their passwords and allowing others to access content, the provider could simply encrypt some sensitive user information (such as credit card details, etc.) under a user's hashed password and publish this ciphertext. As anyone who knows the password could decrypt, it is then in the user's interest to keep his password secret.

Our construction. Our construction, as well as its proof, is inspired from the *functional encryption* scheme by Garg et al. [GGH+13b].

[2] Such a key consists of a pair (sa, hpw) of a salt and a hashed password $hpw = \mathsf{PH}(sa, pw)$ for a password-hashing function PH. Given a WE for the NP-language $\{(sa, \mathsf{PH}(sa, pw)) \mid pw\}$, messages are encrypted w.r.t. statements (sa, hpw) and can be decrypted using witness pw such that $hpw = \mathsf{PH}(sa, pw)$.

The parameters required for encryption $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$ consist of two public keys of a standard public-key encryption (PKE) scheme and a common reference string for a non-interactive zero-knowledge (NIZK) proof system. The encryption $c = (x, c_1, c_2, \pi)$ of a message m for an instance x is simply a Naor-Yung [NY90, Sah99] CCA-secure encryption of the pair (x, m); that is, encryptions c_1 and c_2 of (x, m) under pk_1 and pk_2, respectively, together with a NIZK proof π showing that the two ciphertext c_1, c_2 encrypt the same message.

The setup algorithm samples two key pairs (sk_1, pk_1), (sk_2, pk_2) for the PKE scheme and a CRS for the NIZK proof system. The parameters $\mathsf{pp_d}$ required for decryption consist of the obfuscation $\tilde{D}$ of a circuit D, defined as follows. On input a ciphertext $c = (x, c_1, c_2, \pi)$ and a string w, the circuit D

- checks if $R(x, w) = 1$ (i.e., w is a witness for $x \in L$);
- checks if π is a proof that c_1 and c_2 encrypt the same message; and
- if both checks pass, decrypts $(x', m) = \mathsf{PK.Dec}(sk_1, c_1)$ and outputs m if $x' = x$.

Given an (obfuscated) circuit as above, the decryption algorithm of our WE scheme simply evaluates $\tilde{D}((x, c_1, c_2, \pi), w)$, which will output the encrypted message for any witness w with $R(x, w) = 1$.

We prove in Theorem 1 that the above is a secure offline-WE scheme (defined as ciphertexts for $x \notin L$ computationally hiding the message), assuming that the obfuscation satisfies the notion of *indistinguishability obfuscation* [BGI+01], the NIKZ is *statistically simulation-sound* [GGH+13b] and the PKE is semantically secure under chosen-plaintext attack (CPA).

Efficiency of encryption. In Section 5 we propose a concrete instantiation of our encryption algorithm. In order to avoid random oracles, we use Groth-Sahai proofs [GS08], which are perfectly sound NIZK proofs in the standard model for languages defined over bilinear groups (see [GPS08]). They let us prove that two ElGamal ciphertexts encrypt the same message. Using ideas from [GGH+13b] and translating them into the bilinear-group framework, we convert the proof system into a statistically simulation-sound (SSS) proof system. Under the so-called SXDH assumption (which states that the decisional Diffie-Hellman problem holds in the base groups), the encryption scheme is CPA-secure and the proof system we construct is zero-knowledge.

In our instantiation a proof consists of 28 elements from a bilinear group and is computed by using bilinear-group exponentiations. For a 128-bit security level, the size of the output of our encryption algorithm, comprising 2 ciphertexts and one SSS proof, is about 1.3 kB.

Handling long messages and instances. ElGamal encryption is defined over a group $\mathbb{G}$ and encrypts elements from $\mathbb{G}$; we therefore need to encode the message (x, m) into $\mathbb{G}$. Using elliptic-curve-based groups, for 128-bit security the length of an element from $\mathbb{G}$ is 256 bits, and standard encoding techniques to elliptic curves [FJT13] allow for encoding of 128 bits into one group element, which is prohibitively small for any meaningful application.

We could of course choose a larger group such that one group element fits the entire tuple (x, m), but this would become very inefficient for large values. The encryption procedure we construct in Section 5 will therefore allow to encrypt arbitrarily long messages by encrypting them block-wise. We then need to provide a proof for each 128-bit block separately; however, using some optimization, we manage to limit the growth of the ciphertext to 0.25 kB for every 128 bits of plaintext, meaning the ciphertext grows by a factor of 16 compared to the plaintext.

A more efficient way of dealing with large m's is to use key encapsulation: when encrypting, the sender first picks a key k for a symmetric encryption scheme and generates a ciphertext $c = (c_K, c_M)$, where c_K is the WE encryption of (x, k), and c_M is the (secret-key) encryption of m under key k. To decrypt (given a witness w), the receiver first decrypts c_K to learn k and then decrypts c_M to recover m.

Dealing with large x's turns out more tricky. Standard tricks like encrypting only a hash $y = H(x)$ instead of the entire x using a collision-resistant hash function[3] do work, but require the obfuscation to satisfy the notion of *extractability obfuscation* (eO) [BCP14], which seems a much stronger assumption than indistinguishability obfuscation [GGHW14].

Functional witness encryption. Functional witness encryption was proposed by Boyle et al. [BCP14]. Here one additionally encrypts a circuit f together with the instance x and message m, and a party knowing a witness w now does not learn m itself, but only the function $f(m, w)$. For example, x could be a labeled graph and a party knowing a t-clique in x can learn the labels of this clique (but no other labels). In Section 4 we explain how our WE scheme can be easily turned into a functional WE scheme by changing the definition of the (obfuscated) decryption circuit: instead of outputting m if given a witness w, we now parse m as a pair (f, m'), and output $f(m', w)$. Note that key encapsulation as discussed above does not work any more for this functional version of the WE scheme.

Boyle et al. [BCP14] consider an "extractable" notion of functional WE, where they require that an adversary who can distinguish the encryption of two messages m_0, m_1 must know of a witness w where $f(m_0, w) \neq f(m_1, w)$. They construct such a scheme assuming *extractability* obfuscation. The notion we achieve is akin to that for standard WE, namely that encryptions of m_0, m_1 are indistinguishable if x is not in the language, and thus no witness exists. However, we only require indistinguishability obfuscation, avoiding thereby the implausibility result for extractable witness encryption and eO by Garg et al. [GGHW14].

Related work. Zhandry [Zha14] proposes the related notion of *reusable witness encryption*. This notion is similar to offline WE as defined in this work, the main difference being that now there is a secret key that allows decryption of any ciphertext (even if it is encrypted relative to a no instance). Zhandry constructs reusable WE from *witness pseudorandom functions*, a primitive he introduces and constructs based on a new family of assumptions on multilinear maps.

The security notion of reusable WE differs from offline WE in two main aspects. First, in offline WE the challenge is chosen by the adversary, whereas in reusable WE the challenge may not be arbitrary, for otherwise the implausibility result of [GGHW14] applies. To overcome this implausibility, the security of reusable WE is defined with respect to a fixed instance sampler, which upon receiving the offline public parameters samples a challenge instance for the WE adversary. Second, reusable WE allows for CCA type decryption queries during the security experiment, whereas for offline WE we don't consider decryption queries as there's no secret key which would allow to implement the security experiment efficiently.

2 Preliminaries

2.1 Notations and Conventions

Families of circuits. A family of circuits $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ is of polynomial size if for some polynomial $p(.)$ the size of every $C \in \mathcal{C}_\lambda$ is at most $|C| \leq p(\lambda)$.

Probabilistic algorithms. If $\mathcal{X}$ is a finite set then $x \leftarrow \mathcal{X}$ denotes the process of sampling x uniformly at random from $\mathcal{X}$. Let $\mathcal{A}$ be a probabilistic polynomial-time (PPT) algorithm; then $\Pr[y \leftarrow \mathcal{A}(x)]$ denotes the probability that $\mathcal{A}(x)$ outputs y when run on uniformly sampled coins. We let $\Pr\left[\varphi(x_1, x_2, \ldots) = 1 :\ x_1 \leftarrow \mathcal{X}_1; x_2 \leftarrow \mathcal{X}_2; \ldots\right]$ denote the probability that the predicate φ evaluated on $(x_1, x_2, \ldots)$ is true after the ordered execution of $x_1 \leftarrow \mathcal{X}_1$, $x_2 \leftarrow \mathcal{X}_2, \ldots$

Negligible functions. A function $\nu\colon \mathbb{N} \to \mathbb{R}$ is called negligible, if for every positive polynomial $p(\cdot)$, and all sufficiently large $n \in \mathbb{N}$, it holds that $\nu(n) \leq \frac{1}{p(n)}$. With $\mathsf{negl}(\cdot)$ we denote a negligible function.

[3] The full x can then be explicitly given to the decryption algorithm, which will check if $y \stackrel{?}{=} H(x)$

2.2 Indistinguishability Obfuscation

The strongest notion of obfuscation is *virtual black-box* obfuscation, where one requires that given the obfuscation of a circuit, everything that can be done could also be done given black-box access to the functionality realized by the circuit. Barak et al. [BGI+12] show that this notion cannot be achieved in general. They propose several weaker notions which potentially can be realized, the weakest being indistinguishability obfuscation ($i\mathcal{O}$), which only requires that the obfuscations of two circuits computing the same function are indistinguishable.

Definition 1 (Indistinguishability obfuscation [BGI+12, GGH+13b]). *A uniform PPT algorithm $i\mathcal{O}$ is an indistinguishability obfuscator for a family of polynomial-size circuits $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$, if the following hold:*

- *For all $\lambda \in \mathbb{N}$, $C \in \mathcal{C}_\lambda$ and x we have*

$$\Pr\left[C(x) = \widetilde{C}(x) : \ \widetilde{C} \leftarrow i\mathcal{O}(1^\lambda, C)\right] = 1 \ .$$

- *For every non-uniform PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $C_0, C_1 \in \mathcal{C}_\lambda$ such that $C_0(x) = C_1(x)$ for all x:*

$$\left|\Pr\left[\mathcal{A}(i\mathcal{O}(1^\lambda, C_0)) = 1\right] - \Pr\left[\mathcal{A}(i\mathcal{O}(1^\lambda, C_1)) = 1\right]\right| = \mathsf{negl}(\lambda) \ . \tag{1}$$

Garg et al. [GGH+13b] construct a candidate $i\mathcal{O}$ for families of polynomial-size circuits, based on a strong assumption on "approximate" multi-linear maps from [GGH13a] and fully homomorphic encryption [Gen09].

2.3 Statistically Simulation-Sound NIZK

A non-interactive zero-knowledge (NIZK) proof system for a language $L \in \mathsf{NP}$ consists of four PPT algorithms: a *common-reference string* (CRS) generator G, a prover P, a verifier V, and a simulator S. A proof for a statement $y \in L$ on a CRS generated by G consists of a single message π sent from P to V, to which V responds by either accepting or rejecting. We require a NIZK proof system to satisfy completeness, statistical soundness, and zero-knowledge. Completeness requires the ability of an honest P to convince V of the validity of all true statements $y \in L$, by producing a proof π using a witness for $y \in L$. Statistical soundness requires that no unbounded adversary can convince an honest verifier of a proof of a false statement. Zero-knowledge means that a proof does not reveal any information (in a computational sense) about the witness used to compute it; this is formalized by requiring the existence of a simulator that can output a CRS and a proof for any statement, which are computationally indistinguishable from real ones.

A NIZK proof system is statistically simulation-sound (SSS-NIZK), as defined in [GGH+13b], if no *unbounded* adversary can produce a valid proof of an incorrect statement $y \notin L$ even when given a simulated proof for any other statement $y' \neq y$.

Definition 2 (SSS-NIZK). *A tuple of PPT algorithms $(\mathsf{G}, \mathsf{P}, \mathsf{V}, \mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2))$ is a statistically simulation-sound non-interactive zero-knowledge (SSS-NIZK) proof system for a language $L \in \mathsf{NP}$ with witness relation R if the following hold:*

- *Perfect completeness: For every (y, w) such that $R(y, w) = 1$,*

$$\Pr\left[\mathsf{V}(\mathsf{crs}, y, \pi) = 1 : \mathsf{crs} \leftarrow \mathsf{G}(1^\lambda)\,;\, \pi \leftarrow \mathsf{P}(\mathsf{crs}, y, w)\right] = 1 \ .$$

- *Statistical soundness:*

$$\Pr\left[\exists\,(y, \pi) \ s.t.\ y' \notin L \ \wedge\ \mathsf{V}(\mathsf{crs}, y, \pi) = 1 \ : \ \mathsf{crs} \leftarrow \mathsf{G}(1^\lambda)\right] = \mathsf{negl}(\lambda) \ .$$

$\underline{\mathbf{Exp}_{\mathcal{A}}^{\text{CPA-}b}(\lambda):}$
$(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$
$(m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda, pk)$
$c_b \leftarrow \mathsf{Enc}(pk, m_b)$
$b' \leftarrow \mathcal{A}_2(st, c_b)$
Return b'

Fig. 1. $\mathbf{Exp}_{\mathcal{A}}^{\text{CPA-}b}(\lambda)$: The security game of CPA-secure public-key encryption.

– *Computational zero-knowledge: For every (y, w) such that $R(y, w) = 1$, and non-uniform PPT adversary $\mathcal{A}$, it holds that*

$$\left|\Pr\left[\mathcal{A}(\mathsf{crs}, y, \pi) = 1 : \mathsf{crs} \leftarrow \mathsf{G}(1^\lambda); \pi \leftarrow \mathsf{P}(\mathsf{crs}, y, w)\right] - \Pr\left[\mathcal{A}(\mathsf{crs}, y, \pi) = 1 : (\mathsf{crs}, \tau) \leftarrow \mathsf{S}_1(1^\lambda, y); \pi \leftarrow \mathsf{S}_2(\mathsf{crs}, \tau, y)\right]\right| = \mathsf{negl}(\lambda) \ . \quad (2)$$

– *Statistical simulation soundness: For every y, it holds that*

$$\Pr\left[\begin{array}{l|l} \exists\,(y', \pi')\ s.t.\ y' \neq y \wedge y' \notin L & (\mathsf{crs}, \tau) \leftarrow \mathsf{S}_1(1^\lambda, y); \\ \wedge\ \mathsf{V}(\mathsf{crs}, y', \pi') = 1 & \pi \leftarrow \mathsf{S}_2(\mathsf{crs}, \tau, y) \end{array}\right] = \mathsf{negl}(\lambda) \ . \quad (3)$$

Garg et al. [GGH+13b] construct an SSS-NIZK scheme from any statistically sound NIZK scheme and any computationally hiding and perfectly binding non-interactive commitment scheme. In Sect. 5, we give an efficient instantiation of this, following their blueprint and using Groth-Sahai proofs [GS08], which are perfectly sound, and ElGamal encryption as perfectly binding and computationally hiding commitment scheme.

2.4 Public-Key Encryption

Our last ingredient is a standard public-key encryption scheme.

Definition 3 (PKE). *A public-key encryption scheme for a message space $\mathcal{M}$ is a tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Gen, on input a security parameter 1^λ, outputs a secret/public key pair (sk, pk). Enc, on input a public key pk and a message $m \in \mathcal{M}$, outputs a ciphertext c using randomness $r \in \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$. Finally, Dec, on input a secret key sk and a ciphertext c, outputs $m \in \mathcal{M} \cup \{\perp\}$. Furthermore we require correctness and security:*

– *Correctness: For every $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$, $c \leftarrow \mathsf{Enc}(pk, m)$, it holds that $\mathsf{Dec}(sk, c) = m$.*
– *Indistinguishability under chosen-plaintext attacks (CPA): For every non-uniform PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in $\mathbf{Exp}_{\mathcal{A}}^{\text{CPA-}b}(\lambda)$ as defined in Figure 1 (where we assume that $\mathcal{A}$'s output satisfies $|m_0| = |m_1|$), it holds that*

$$\left|\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\text{CPA-}0}(\lambda) = 1\right] - \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\text{CPA-}1}(\lambda) = 1\right]\right| = \mathsf{negl}(\lambda) \ .$$

Definition 4. *Let $(\mathsf{PK.Gen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$ be a public-key encryption scheme. Then we define the NP language L_{enc} and let R_{enc} denote its corresponding witness relation:*

$$L_{\mathsf{enc}} := \left\{(pk_1, pk_2, c_1, c_2) \;\middle|\; \begin{array}{r} \exists\,(x, m, r_1, r_2)\ s.t.\ c_1 = \mathsf{PK.Enc}(pk_1, (x, m); r_1) \\ \wedge\ c_2 = \mathsf{PK.Enc}(pk_2, (x, m); r_2) \end{array}\right\} \quad (4)$$

$\underline{\mathbf{Exp}^{\text{sel-WE-}b}_{L,\mathcal{A}}(\lambda):}$

$(x, m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda)$
$(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$
$c_b \leftarrow \mathsf{Enc}(1^\lambda, x, m_b, \mathsf{pp_e})$
$b' \leftarrow \mathcal{A}_2(st, c_b, \mathsf{pp_e}, \mathsf{pp_d})$
If $x \in L$, return 0
Return b'

Fig. 2. $\mathbf{Exp}^{\text{sel-WE-}b}_{L,\mathcal{A}}(\lambda)$: The security game of selectively-secure witness encryption.

$\underline{\mathbf{Exp}^{\text{adp-WE-}b}_{L,\mathcal{A}}(\lambda):}$

$(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(x, m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{pp_e}, \mathsf{pp_d})$
$c_b \leftarrow \mathsf{Enc}(1^\lambda, x, m_b, \mathsf{pp_e})$
$b' \leftarrow \mathcal{A}_2(st, c_b)$
If $x \in L$, return 0
Return b'

Fig. 3. $\mathbf{Exp}^{\text{adp-WE-}b}_{L,\mathcal{A}}(\lambda)$: The security game of adaptively-secure witness encryption.

3 Offline Witness Encryption

A (standard) witness encryption scheme [GGSW13, BH15] is defined by an encryption algorithm Enc that takes a security parameter 1^λ, a statement x and a message m and outputs a ciphertext c; and a decryption algorithm Dec that on input a ciphertext c and a witness w, outputs a message. Offline witness encryption allows for efficient encryption by outsourcing the resource-heavy computations to a setup phase, which is independent of the statement and message to be encrypted. There is thus a third algorithm Setup which on input a security parameter 1^λ outputs a pair of parameters: $\mathsf{pp_e}$, which is used by Enc, and $\mathsf{pp_d}$, which is used by Dec.

In our formalization we follow the strengthened definition of witness encryption put forth by Bellare and Tung Hoang [BH15], who observe that the original WE definition of [GGSW13] allows insecure schemes to be proven secure.

Definition 5 (Offline witness encryption). *An offline witness encryption scheme for a language $L \in \mathsf{NP}$ with witness relation R is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ such that*

- $(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$*: On input a security parameter 1^λ, Setup outputs parameters for encryption $\mathsf{pp_e}$ and parameters for decryption $\mathsf{pp_d}$.*
- $c \leftarrow \mathsf{Enc}(1^\lambda, x, m, \mathsf{pp_e})$*: On input a security parameter 1^λ, a string $x \in \{0,1\}^*$, a message $m \in \mathcal{M}$, and encryption parameters $\mathsf{pp_e}$, Enc outputs a ciphertext c.*
- $\mathsf{Dec}(c, w, \mathsf{pp_d}) \in \mathcal{M} \cup \{\perp\}$*: On input a ciphertext c, a string $w \in \{0,1\}^*$ and decryption parameters $\mathsf{pp_d}$, Dec outputs $m \in \mathcal{M} \cup \{\perp\}$.*

We require correctness and security:

- *Correctness: For all $\lambda \in \mathbb{N}$, (x, w) such that $R(x, w) = 1$, $m \in \mathcal{M}$, $(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$, and $c \leftarrow \mathsf{Enc}(1^\lambda, x, m, \mathsf{pp_e})$, we have $\mathsf{Dec}(c, w, \mathsf{pp_d}) = m$.*
- *Security: $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ is selectively secure if for every non-uniform PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in $\mathbf{Exp}^{\text{sel-WE-}b}_{L,\mathcal{A}}(\lambda)$ as defined in Figure 2 (where we assume that $\mathcal{A}$'s output satisfies $|m_0| = |m_1|$), it holds that*

$$\left| \Pr\left[\mathbf{Exp}^{\text{sel-WE-}0}_{L,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{sel-WE-}1}_{L,\mathcal{A}}(\lambda) = 1\right] \right| = \mathsf{negl}(\lambda) \ .$$

Furthermore, $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ is adaptively secure if the same holds for $\mathbf{Exp}^{adp\text{-}WE\text{-}b}_{L,\mathcal{A}}(\lambda)$ as defined in Figure 3.

We now give our construction of offline WE that we have outlined in the introduction and prove that it satisfies selective security.

Construction 1 (Offline WE). Let $\mathsf{PKE} = (\mathsf{PK.Gen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$ be a public-key encryption scheme, $\mathsf{SSS\text{-}NIZK} = (\mathsf{G}, \mathsf{P}, \mathsf{V}, \mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2))$ an SSS-NIZK scheme for L_{enc} (Def. 4), and let $i\mathcal{O}$ be an indistinguishability obfuscator for the family of polynomial-size circuits $\mathcal{D}_\lambda$ defined in (5). We construct an offline witness encryption scheme $\mathsf{OWE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ as follows:

$\underline{(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)}$: On input a security parameter 1^λ, do the following:

- $(sk_1, pk_1) \leftarrow \mathsf{PK.Gen}(1^\lambda)$ and $(sk_2, pk_2) \leftarrow \mathsf{PK.Gen}(1^\lambda)$.
- $\mathsf{crs} \leftarrow \mathsf{NIZK.G}(1^\lambda)$.
- Construct the circuit $D_{sk_j,\mathsf{crs}} \in \mathcal{D}_\lambda$ with $j = 1$

$$
\begin{array}{l}
D_{sk_j,\mathsf{crs}}(c, w): \\
\quad \text{Parse } c \text{ as } c = (x, c_1, c_2, \pi) \\
\quad \text{If } \mathsf{NIZK.V}(\mathsf{crs}, (pk_1, pk_2, c_1, c_2), \pi) = 1 \\
\qquad\qquad \text{// Verify that } \pi \text{ is a proof for } (pk_1, pk_2, c_1, c_2). \\
\qquad\qquad \text{// w.r.t. } L_{\mathsf{enc}}, \text{ where } (pk_1, pk_2) \text{ is hardcoded.} \\
\qquad (\hat{x}, \hat{m}) := \mathsf{PK.Dec}(sk_j, c_j) \\
\qquad \text{If } (\hat{x} = x) \wedge R(x, w) = 1 \\
\qquad\quad \text{Return } \hat{m} \\
\quad \text{Return } \perp
\end{array}
\tag{5}
$$

- $\tilde{D}_{sk_1,\mathsf{crs}} \leftarrow i\mathcal{O}(1^\lambda, D_{sk_1,\mathsf{crs}})$ after padding $D_{sk_1,\mathsf{crs}}$ appropriately.[4]
- Set $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$ and $\mathsf{pp_d} = \tilde{D}_{sk_1,\mathsf{crs}}$.
- Output $(\mathsf{pp_e}, \mathsf{pp_d})$.

$\underline{c \leftarrow \mathsf{Enc}(1^\lambda, x, m, \mathsf{pp_e})}$: On input a security parameter 1^λ, a string $x \in \{0,1\}^*$, a message $m \in \mathcal{M}$, and $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$, Enc does the following:

- $r_1, r_2 \leftarrow \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$.
- Set $c_1 = \mathsf{PK.Enc}(pk_1, (x, m); r_1)$ and $c_2 = \mathsf{PK.Enc}(pk_2, (x, m); r_2)$.
- $\pi \leftarrow \mathsf{NIZK.P}\big(\mathsf{crs}, (pk_1, pk_2, c_1, c_2), (x, m, r_1, r_2)\big)$.
- Output $c := (x, c_1, c_2, \pi)$.

$\underline{\mathsf{Dec}(c, w, \mathsf{pp_d})}$: On input a ciphertext $c = (x, c_1, c_2, \pi)$, a string $w \in \{0,1\}^*$ and parameters $\mathsf{pp_d} = \tilde{D}_{sk_1,\mathsf{crs}}$, Dec interprets $\tilde{D}_{sk_1,\mathsf{crs}}$ as a circuit and outputs $m := \tilde{D}_{sk_1,\mathsf{crs}}(c, w)$.

Theorem 1. $\mathsf{OWE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ *from Construction 1 is a selectively-secure offline witness encryption scheme if* $\mathsf{PKE} = (\mathsf{PK.Gen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$ *is a CPA-secure PKE scheme,* $\mathsf{SSS\text{-}NIZK} = (\mathsf{G}, \mathsf{P}, \mathsf{V}, \mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2))$ *is an SSS-NIZK scheme for* L_{enc}*, and* $i\mathcal{O}$ *is an indistinguishability obfuscator for* $\mathcal{D}_\lambda$*.*

Proof. Assume towards contradiction that there exists a non-uniform PPT adversary $\mathcal{A}$ that non-negligibly distinguishes $\mathbf{Exp}^{\text{sel-WE-0}}_{L,\mathcal{A}}$ from $\mathbf{Exp}^{\text{sel-WE-1}}_{L,\mathcal{A}}$, which we abbreviate as $\mathbf{Exp}^{\text{WE-}b}$. We reach a contradiction by first constructing a series of games $\mathbf{Exp}^{(i)}$ defined in Figure 4, where by construction, $\mathbf{Exp}^{\text{WE-0}} = \mathbf{Exp}^{(0)}$ and $\mathbf{Exp}^{\text{WE-1}} = \mathbf{Exp}^{(6)}$, and then proving for $i = 0, 1, \ldots, 5$ that $\mathbf{Exp}^{(i)}$ and $\mathbf{Exp}^{(i+1)}$ are computationally indistinguishable.

$\mathbf{Exp}^{(1)}$ differs from $\mathbf{Exp}^{(0)}$, which coincides with the original game $\mathbf{Exp}^{\text{WE-0}}$, in that the CRS crs for the NIZK and the proof π are simulated rather than honestly generated. The zero-knowledge property of $\mathsf{SSS\text{-}NIZK}$ guarantees that honestly generated CRSs and proofs are indistinguishable from simulated ones by PPT adversaries.

Proposition 1. $\mathbf{Exp}^{(0)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* $\mathsf{SSS\text{-}NIZK}$ *is zero-knowledge.*

$\mathbf{Exp}^{(2)}$ differs from $\mathbf{Exp}^{(1)}$ in that the second ciphertext c_2 is generated as $\mathsf{PK.Enc}(pk_2, (x, \underline{m_1}))$ rather than $\mathsf{PK.Enc}(pk_2, (x, m_0))$. ($D_{sk_1,\mathsf{crs}}$ and (π, crs) are the same as in $\mathbf{Exp}^{(1)}$.) The CPA-security of PKE for key pk_2 guarantees that whether c_2 encrypts (x, m_0) or (x, m_1) is indistinguishable by PPT adversaries.

[4] W.l.o.g. we assume that $|D_{sk_1,\mathsf{crs}}| = |D_{sk_2,\mathsf{crs}}|$; otherwise we always pad to the maximum possible length.

$\underline{\mathbf{Exp}^{(i)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda) \quad // \; i \in \{0,1,2,3,4,5,6\}}$

$(x, m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda)$
$(sk_1, pk_1) \leftarrow \mathsf{PK.Gen}(1^\lambda)$
$(sk_2, pk_2) \leftarrow \mathsf{PK.Gen}(1^\lambda)$

If $i \in \{0,6\}$ $\quad \mathsf{crs} \leftarrow \mathsf{NIZK.G}(1^\lambda)$
Elseif $i \in \{1,2,3,4,5\}$ $\quad (\mathsf{crs}, \tau) \leftarrow \mathsf{NIZK.S_1}(1^\lambda, c_1, c_2)$

If $i \in \{0,1,2,5,6\}$ $\quad D := D_{sk_j,\mathsf{crs}}$ with $j = 1$ as defined in (5)
Elseif $i \in \{3,4\}$ $\quad D := D_{sk_j,\mathsf{crs}}$ with $j = 2$ as defined in (5)

$\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D)$
Set $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$ and $\mathsf{pp_d} = \tilde{D}$
$r_1, r_2 \leftarrow \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$

If $i \in \{0,1,2,3\}$ $\quad c_1 := \mathsf{PK.Enc}(pk_1, (x, m_0); r_1)$
Elseif $i \in \{4,5,6\}$ $\quad c_1 := \mathsf{PK.Enc}(pk_1, (x, m_1); r_1)$

If $i \in \{0,1\}$ $\quad c_2 := \mathsf{PK.Enc}(pk_2, (x, m_0); r_2)$
Elseif $i \in \{2,3,4,5,6\}$ $\quad c_2 := \mathsf{PK.Enc}(pk_2, (x, m_1); r_2)$

If $i = 0$ $\quad \pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}, (pk_1, pk_2, c_1, c_2), (x, m_0, r_1, r_2))$
Elseif $i = 6$ $\quad \pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}, (pk_1, pk_2, c_1, c_2), (x, \underline{m_1}, r_1, r_2))$
Elseif $i \in \{1,2,3,4,5\}$ $\quad \pi \leftarrow \mathsf{NIZK.S_2}(\mathsf{crs}, \tau, (pk_1, pk_2, c_1, c_2))$

Set $c = (x, c_1, c_2, \pi)$
$b' \leftarrow \mathcal{A}_2(st, c, \mathsf{pp_e}, \mathsf{pp_d})$
If $x \in L$, return 0
Return b'

Fig. 4. The hybrid games used in the proof of Theorem 1.

Proposition 2. $\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* PKE *is CPA-secure.*

$\mathbf{Exp}^{(3)}$ differs from $\mathbf{Exp}^{(2)}$ in that $D_{\underline{sk_2},\mathsf{crs}}$ is obfuscated instead of $D_{sk_1,\mathsf{crs}}$. Statistical simulation-soundness of SSS-NIZK now guarantees that $D_{sk_1,\mathsf{crs}}$ and $D_{sk_2,\mathsf{crs}}$ are functionally equivalent when crs is simulated for the statement $y := (pk_1, pk_2, c_1, c_2)$. It then follows from the security of $i\mathcal{O}$ that their obfuscations are computationally indistinguishable.

Proposition 3. $\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* SSS-NIZK *is statistically simulation-sound, and* $i\mathcal{O}$ *is secure.*

$\mathbf{Exp}^{(4)}$ differs from $\mathbf{Exp}^{(3)}$ in that the first ciphertext c_1 is generated as $\mathsf{PK.Enc}(pk_1, (x, \underline{m_1}))$ rather than $\mathsf{PK.Enc}(pk_1, (x, m_0))$. ($D_{sk_2,\mathsf{crs}}$ and (π, crs) are the same as in $\mathbf{Exp}^{(3)}$.) Now CPA security of PKE w.r.t. pk_1 implies that this change is computationally indistinguishable.

Proposition 4. $\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(4)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* PKE *is CPA-secure.*

$\mathbf{Exp}^{(5)}$ differs from $\mathbf{Exp}^{(4)}$ in that $D_{\underline{sk_1},\mathsf{crs}}$ is obfuscated rather than $D_{sk_2,\mathsf{crs}}$. Statistical simulation soundness of SSS-NIZK together with security of $i\mathcal{O}$ implies that this change is computationally indistinguishable.

Proposition 5. $\mathbf{Exp}^{(4)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(5)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* SSS-NIZK *is statistically simulation-sound, and* $i\mathcal{O}$ *is secure.*

$\mathbf{Exp}^{(6)}$ coincides with the original game $\mathbf{Exp}^{\text{WE-1}}$, and differs from $\mathbf{Exp}^{(5)}$ in that the CRS and NIZK proof (crs, π) are honestly generated rather than simulated. By the zero-knowledge property of SSS-NIZK this change is computationally indistinguishable.

Proposition 6. $\mathbf{Exp}^{(5)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *and* $\mathbf{Exp}^{(6)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)$ *are computationally indistinguishable if* SSS-NIZK *is zero-knowledge.*

Theorem 1 now follows from Propositions 1–6, which we prove in Appendix A. □

4 Offline Functional Witness Encryption

We also define a "functional" version of offline WE. Here a receiver holding a witness w does not learn the message m, but now we think of the encrypted string m as a tuple (m', f), and decryption using witness w yields $f(m', w)$.

Definition 6 (Offline functional witness encryption). *Everything is defined as in Definition 5, except that we change the correctness property to*

- *Correctness: For all* $\lambda \in \mathbb{N}$, (x, w) *such that* $R(x, w) = 1$, $m \in \mathcal{M}$, $(\mathsf{pp_e}, \mathsf{pp_d}) \leftarrow \mathsf{Setup}(1^\lambda)$, *and* $c \leftarrow \mathsf{Enc}(1^\lambda, x, m, \mathsf{pp_e})$, *parsing* m *as a tuple* $m = (m', f)$, *we have* $\mathsf{Dec}(c, w, \mathsf{pp_d}) = f(m', w)$.

Construction 2 (Offline functional WE). This construction is defined exactly as Construction 1, except that in the definition of the decryption circuit in Equation (5) on page 8 we replace

$$\text{Return } \hat{m}$$

with

$$\text{Parse } \hat{m} \text{ as } (\hat{m}', f) \text{ and return } f(\hat{m}', w) \ .$$

Here, f can be any description of a function, like a circuit or a Turing machine. If we let f encode a Turing machine, we must put an upper bound on its running time, as the decryption, which must evaluate $f(\hat{m}, w)$, is done by an (obfuscated) circuit whose size is a priori fixed.

The proof of Theorem 2 below is basically identical to the proof of Theorem 1 and is therefore omitted.

Theorem 2. *Construction 2 is a selectively-secure offline* functional *witness encryption scheme under the same assumptions as in Theorem 1.*

5 Instantiating Enc in the Standard Model

We now show how to efficiently instantiate the encryption algorithm of our offline-WE scheme in a bilinear group and prove its security under the SXDH assumption without random oracles. We use ElGamal encryption [ElG84] for the public-key encryption scheme and build an SSS-NIZK proof system from Groth-Sahai proofs [GS08] following the abstract blueprint for it given in [GGH+13b].

5.1 Tools

Bilinear groups. $\mathcal{G}$ is a bilinear-group generator if on input a security parameter 1^λ it returns the description of a bilinear group $\Lambda = (p, \mathbb{G}, \mathbb{H}, \mathbb{T}, e, g, h)$ with the following properties:

- $\mathbb{G}$, $\mathbb{H}$ and $\mathbb{T}$ are groups of prime order p, where p is of bit-length λ.
- $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{T}$ is a bilinear map, that is, $e(R^a, S^b) = e(R, S)^{ab}$ for all $R \in \mathbb{G}$, $S \in \mathbb{H}$, $a, b \in \mathbb{Z}_p$.
- g and h generate $\mathbb{G}$ and $\mathbb{H}$, resp., and $e(g, h)$ generates $\mathbb{T}$.

We will use Type-3 bilinear groups [GPS08], where no efficiently computable homomorphisms are assumed to exist from $\mathbb{G}$ to $\mathbb{H}$ or vice versa. We can therefore assume that the decisional Diffie-Hellman assumption (DDH) holds in $\mathbb{G}$: for any non-uniform PPT $\mathcal{A}$,

$$\left|\Pr\left[\begin{matrix}1 \leftarrow \mathcal{A}(\Lambda, g^a, g^b, g^{ab}) : \\ \Lambda \leftarrow \mathcal{G}(1^\lambda); a, b \leftarrow \mathbb{Z}_p\end{matrix}\right] - \Pr\left[\begin{matrix}1 \leftarrow \mathcal{A}(\Lambda, g^a, g^b, g^c) : \\ \Lambda \leftarrow \mathcal{G}(1^\lambda); a, b, c \leftarrow \mathbb{Z}_p\end{matrix}\right]\right| = \mathsf{negl}(\lambda) \ . \tag{6}$$

We moreover assume DDH holds in $\mathbb{H}$, that is, (6) holds with g replaced by h. The SXDH assumption for a bilinear-group generator $\mathcal{G}$ is that DDH holds in both $\mathbb{G}$ and $\mathbb{H}$. It implies security of Groth-Sahai proofs.

ElGamal encryption. We use ElGamal encryption to encrypt message vectors in $\mathbb{G}^\ell$, for some fixed ℓ. A secret key $\boldsymbol{x} \leftarrow \mathbb{Z}_p^\ell$ defines a public key $\boldsymbol{X} \in \mathbb{G}^\ell$ as $X_i := g^{x_i}$ for $i = 1, \ldots, \ell$. A message $\boldsymbol{M} = (M_i)_{i=1}^\ell \in \mathbb{G}^\ell$ is encrypted under $\boldsymbol{X}$ by choosing $r \leftarrow \mathbb{Z}_p$ and setting $\boldsymbol{c} = (c_1, \ldots, c_\ell, c_{\ell+1}) := ((M_i \cdot {X_i}^r)_{i=1}^\ell, g^r)$. Note that we use the same randomness for every component, which decreases ciphertext length. CPA security follows from the DDH assumption in $\mathbb{G}$ via a standard hybrid argument.

Groth-Sahai proofs. Groth-Sahai (GS) proofs [GS08] are efficient non-interactive witness-indistinguishable[5] (WI) proofs for several types of equations in bilinear groups. We only require *linear pairing-product equations* over variables $W_1, \ldots, W_n \in \mathbb{H}$, which are of the form

$$\prod_{i=1}^{n} e(A_i, \underline{W_i}) = t \ , \tag{7}$$

where $(A_i)_{i=1}^n \in \mathbb{G}^n$, and $t \in \mathbb{T}$ are public constants. (As a convention, we always underline the variables in equations to ease readability.) GS proofs allow a prover to prove that she knows an assignment to variables which satisfy a given set of equations. Groth-Sahai proofs are perfectly sound (meaning there do not exist proofs for an unsatisfiable set of equations); moreover, given a trapdoor for the CRS, one can extract the satisfying values from a proof. The instantiation of GS proofs we use is WI under the SXDH assumption. The cost of a proof is 2 elements from $\mathbb{H}$ per variable and 2 elements from $\mathbb{G}$ per equation.

5.2 Instantiation

Using ElGamal encryption, we encode pairs $M = (x, m)$ (that is, statement/message pairs which we encrypt in our offline-WE instantiation) as a vector of group elements from $\mathbb{G}^\ell$. We thus assume that there exists an efficiently decodable encoding Cd of pairs (x, m) into $\mathbb{G}^\ell$ [FJT13].

We now construct an SSS-NIZK proof system which allows us to prove that 2 ElGamal ciphertexts under different keys encrypt the same message M. A CRS for this system consists of a CRS for GS proofs together with a commitment C to $\mathbf{1}$ (which we assume is not a valid statement/message pair M). An SSS-NIZK proof for the statement y: "$\boldsymbol{c}^{(1)}$ and $\boldsymbol{c}^{(2)}$ encrypt the same message" is a GS proof that either y holds OR C is a commitment to $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$. Statistical soundness now follows from perfect soundness of GS proofs: since C is not a commitment to $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$, it must be the case that y holds.

Zero-knowledge holds, since given a statement $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$, the simulator can include a commitment to it in the CRS and can now use the second clause in the disjunction to simulate a proof for that statement. Since this is (in an information-theoretic sense) the only statement that can be simulated, statistical simulation-soundness (SSS) holds as well. We now present the details.

Commitment: A statement for our language L_{enc} defined in Equation (4) is of the form $(\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$, where $\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)} \in \mathbb{G}^\ell$ are ElGamal encryption keys and $\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)} \in \mathbb{G}^{\ell+1}$ are ElGamal encryptions of the same message. Since the public keys are hard-coded in the description of $D_{sk_j,\mathsf{crs}}(c, w)$ (defined in (5)), we need not include them in the statement.[6] We define our non-interactive commitment scheme that lets us commit to a message $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}) \in \mathbb{G}^{2\ell+2}$ as follows:

- The commitment key is $ck = (K_1^{(1)}, \ldots, K_{\ell+1}^{(1)}, K_1^{(2)}, \ldots, K_{\ell+1}^{(2)}) \leftarrow \mathbb{G}^{2\ell+2}$.

[5] Witness-indistinguishability for a proof system for a language L means the following: no PPT adversary that given crs chooses y, w_0, w_1 with $R(y, w_0) = R(y, w_1) = 1$ can distinguish $\pi_0 \leftarrow \mathsf{P}(\mathsf{crs}, y, w_0)$ from $\pi_1 \leftarrow \mathsf{P}(\mathsf{crs}, y, w_1)$.

[6] We actually construct a proof system for the language $L_{\mathsf{enc},pk_1,pk_2} := \{(c_1, c_2) | \ \exists\, (M, r_1, r_2) : c_1 = \mathsf{PK.Enc}(pk_1, M; r_1) \wedge c_2 = \mathsf{PK.Enc}(pk_2, M; r_2)\}$, where M is an encoding of (x, m).

– A commitment $\mathsf{Com}(ck, (\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}))$ to a message $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}) \in \mathbb{G}^{2\ell+2}$ is computed by picking $r_c \leftarrow \mathbb{Z}_p$ and setting

$$\boldsymbol{C} = \left(\left(C_j^{(i)} := c_j^{(i)} \cdot (K_j^{(i)})^{r_c}\right)_{j=1\ldots\ell+1}^{i=1,2},\ C' := g^{r_c} \right) \ .$$

A commitment can be opened by publishing the "opening" $W = h^{r_c}$, which allows one to check $e(C', h) = e(g, W)$ and

$$e(C_j^{(i)} \cdot (c_j^{(i)})^{-1}, h) = e(K_j^{(i)}, W) \quad \text{for } i = 1, 2 \ , \ j = 1 \ldots, \ell + 1 \ .$$

This yields a perfectly binding commitment scheme for messages from $\mathbb{G}^{2\ell+2}$, and, as the commitment is an ElGamal encryption, it is computationally hiding under the DDH assumption in $\mathbb{G}$.

Proof system: The CRS of our SSS proof system consists of a $\mathsf{crs}_{\mathrm{GS}} \leftarrow \mathsf{GS.G}(\Lambda)$, a commitment key $ck \leftarrow \mathbb{G}^{2\ell+2}$ and a commitment $\boldsymbol{C} \leftarrow \mathsf{Com}(ck, (1, \ldots, 1))$, that is

$$\mathsf{crs} := (\mathsf{crs}_{\mathrm{GS}}, ck, \boldsymbol{C}) \ .$$

We use (witness-indistinguishable) GS proofs to prove that there exist values H_c, H_e, W_1, W_2, $W_c \in \mathbb{H}$ which satisfy the following equations:

$$e(g, \underline{H_c} \cdot \underline{H_e} \cdot h^{-1}) = 1 \tag{8}$$

$$e(C_j^{(i)} \cdot (c_j^{(i)})^{-1}, \underline{H_c}) = e(K_j^{(i)}, \underline{W_c}) \quad \text{for } i = 1, 2 \ , \ j = 1 \ldots, \ell + 1 \tag{9}$$

$$e(C', \underline{H_c}) = e(g, \underline{W_c}) \tag{10}$$

$$e(c_j^{(1)} \cdot (c_j^{(2)})^{-1}, \underline{H_e}) = e(X_j^{(1)}, \underline{W_1})\, e(X_j^{(2)}, \underline{W_2}) \quad \text{for } j = 1, \ldots, \ell \tag{11}$$

$$e(c_{\ell+1}^{(i)}, \underline{H_e}) = e(g, \underline{W_i}) \text{ for } i = 1, 2 \tag{12}$$

A user who computes encryptions $\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}$ as $\boldsymbol{c}^{(i)} = \left((M_j \cdot (X_j^{(i)})^{r_i})_{j=1}^{\ell}, g^{r_i}\right)$ instantiates the variables as

$$H_c := 1 \ , \qquad H_e := h \ , \qquad W_c := 1 \ , \qquad W_1 := h^{r_1} \ , \qquad W_2 := h^{r_2} \ ,$$

which satisfy equations (8)–(12) and can thus compute a GS proof.

Soundness. Below we show that a proof for equations (8)–(12) proves that

$$\begin{array}{l} - \textbf{ either } \boldsymbol{c}^{(1)} \text{ and } \boldsymbol{c}^{(2)} \text{ are encryptions of the same message} \\ - \textbf{ or } \boldsymbol{C} \text{ contained in the CRS is a commitment to } (\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}). \end{array} \tag{13}$$

Since GS proofs are statistically sound and an honestly generated CRS contains a commitment to $(1, \ldots, 1)$ (which we assume is not a valid instance/message pair), a valid proof shows that the "either" clause above is satisfied, thus $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$ is in the language. We now show (13).

– Equation (8) proves that either $H_c \neq 1$ or $H_e \neq 1$; since $e(g, 1 \cdot 1 \cdot h^{-1}) \neq 1$.
– If $H_c \neq 1$ then (9)–(10) prove that $(C_1^{(1)}, \ldots, C_{\ell+1}^{(1)}, C_1^{(2)}, \ldots, C_{\ell+1}^{(2)}, C')$ commits to $(c_1^{(1)}, \ldots, c_{\ell+1}^{(1)}$, $c_1^{(2)} \ldots, c_{\ell+1}^{(2)})$: Let $\eta, \omega \in \mathbb{Z}_p$, $\eta \neq 0$, be such that $H_c = h^{\eta}$ and $W_c = h^{\omega}$. From (10) we have $C' = g^{\omega/\eta}$, whereas the equations in (9) yield $C_j^{(i)} \cdot (c_j^{(i)})^{-1} = (K_j^{(i)})^{\omega/\eta}$, thus $C_j^{(i)} = c_j^{(i)} \cdot (K_j^{(i)})^{\omega/\eta}$, which together means that $(C_1^{(1)}, \ldots, C_{\ell+1}^{(2)}, C')$ is a commitment to $(c_1^{(1)}, \ldots, c_{\ell+1}^{(2)})$ with randomness $r_c = \omega/\eta$.
– If $H_e \neq 1$ then with $\eta \neq 0, \omega_1$ and ω_2 such that $H_e = h^{\eta}$ and $W_i = h^{\omega_i}$ the equations in (12) show that $c_{\ell+1}^{(i)} = g^{\omega_i/\eta}$, for $i = 1, 2$. Set $r_i := \omega_i/\eta$ and let $m_j^{(i)}$ be (the unique values) such that $c_j^{(i)} = g^{m_j^{(i)}} \cdot (X_j^{(i)})^{r_i}$. Then the equations in (11) yield $c_j^{(1)} \cdot (c_j^{(2)})^{-1} = (X_j^{(1)})^{r_1} \cdot (X_j^{(2)})^{-r_2}$, thus $g^{m_j^{(1)}} = g^{m_j^{(2)}}$ for all $j = 1, \ldots, \ell$, meaning $\boldsymbol{c}^{(1)}$ and $\boldsymbol{c}^{(2)}$ encrypt the same message.

Simulation. Given a statement $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$, the simulator sets up the CRS by choosing $r_c \leftarrow \mathbb{Z}_p$ and setting $\boldsymbol{C} := \mathsf{Com}(ck, (\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}); r_c)$. It then computes a proof for equations (8)–(12) by instantiating the variables as

$$H_c := h \ , \qquad H_e := 1 \ , \qquad W_c := h^{r_c} \ , \qquad W_1 := 1 \ , \qquad W_2 := 1 \ .$$

Since the commitment in the CRS is hiding under DDH in $\mathbb{G}$, and since GS proofs are witness-indistinguishable under SXDH, this simulation is also indistinguishable under SXDH (which implies DDH in $\mathbb{G}$). Statistical simulation-soundness holds, since once the CRS is set up, $(\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$ is the only statement for which a proof using the 2^{nd} clause in (13) can be computed. Any other proof must use the first clause, meaning the statement must be in the language.

5.3 Cost of an Encryption

In standard implementations of bilinear groups for 128-bit security, $\mathbb{G}$ elements are of size 256 bits and $\mathbb{H}$ elements are of size 512 bits. Let ℓ be such that pairs (x, m) are of size $< 128 \cdot \ell$ bits, that is, they can be mapped to $\mathbb{G}^\ell$.

An encryption in our WE scheme then consists of two ElGamal ciphertexts (each in $\mathbb{G}^{\ell+1}$) and a GS proof with 5 variables in $\mathbb{H}$ (requiring 10 elements from $\mathbb{H}$) and $3\ell+6$ linear equations (requiring $6\ell+12$ elements from $\mathbb{G}$). Computing an ElGamal encryption requires $\ell+1$ exponentiations and ℓ group operations in $\mathbb{G}$. The 2 elements from $\mathbb{H}$ required for each variable require 2 exponentiations and one group operation in $\mathbb{H}$. The 2 elements from $\mathbb{G}$ required for each equation are computed using together 4 exponentiations and 2 group operations in $\mathbb{G}$.

With the above instantiation the output of Enc is in $\mathbb{G}^{8\ell+14} \times \mathbb{H}^{10}$. If two group elements suffice to encode pairs (x, m) then one encryption has ≈ 1.6 kB. For every 128-bit increase of the message length, the encryption only grows by 8 elements from $\mathbb{G}$, that is 0.25 kB.

References

[AHKM14] Daniel Apon, Yan Huang, Jonathan Katz, and Alex J. Malozemoff. Implementing cryptographic program obfuscation. Cryptology ePrint Archive, Report 2014/779, 2014. `http://eprint.iacr.org/`.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 52–73. Springer, February 2014.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, August 2001.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, May 2012.

[BH15] Mihir Bellare and Viet Tung Hoang. Adaptive witness encryption and asymmetric password-based cryptography. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 308–331, 2015.

[EGM96] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, 1996.

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, August 1984.

[FJT13] Pierre-Alain Fouque, Antoine Joux, and Mehdi Tibouchi. Injective encodings to elliptic curves. In Colin Boyd and Leonie Simpson, editors, *ACISP 13*, volume 7959 of *LNCS*, pages 203–218. Springer, July 2013.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, May 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GGHW14] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 518–535. Springer, August 2014.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013.

[GLW14] Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 426–443. Springer, August 2014.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, April 2008.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Computer Society Press, October 1999.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, May 2005.

[Zha14] Mark Zhandry. How to avoid obfuscation using witness PRFs. Cryptology ePrint Archive, Report 2014/301, 2014. `http://eprint.iacr.org/2014/301`.

A Proofs

Below, we assume that the adversary $\mathcal{A}$ is deterministic, which is without loss of generality as we always can fix $\mathcal{A}$'s random coins to some value maximizing its advantage. As $\mathcal{A}$ has zero advantage if the x it initially outputs is in L, we can further assume wlog. that the x initially output by $\mathcal{A}$ is never in L.

A.1 Proof of Proposition 1

Proof. Assume towards contradiction that there exists a non-uniform PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a polynomial $p(\cdot)$ such that for infinitely many λ,

$$\left| \Pr[\mathbf{Exp}^{(0)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda) = 1] \right| \geq \frac{1}{p(\lambda)} \ .$$

Then we use $\mathcal{A}$ to construct a non-uniform PPT adversary $\mathcal{B}$ against the zero-knowledge security of $\mathsf{SSS\text{-}NIZK}$ (cf.(2)) as follows:

$\underline{\mathcal{B}(1^\lambda)}$:

- $(x, m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda)$.
- $(sk_1, pk_1) \leftarrow \mathsf{PK.Gen}(1^\lambda)$ and $(sk_2, pk_2) \leftarrow \mathsf{PK.Gen}(1^\lambda)$.
- $r_1, r_2 \leftarrow \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$.
- $c_1 := \mathsf{PK.Enc}(pk_1, (x, m_0); r_1)$ and $c_2 := \mathsf{PK.Enc}(pk_2, (x, m_0); r_2)$.
- Set $y = (pk_1, pk_2, c_1, c_2)$ and $w = (x, m_0, r_1, r_2)$ and note that $R_{\mathsf{enc}}(y, w) = 1$.
- Submit (y, w) to the zero-knowledge game of (2) to obtain either
  - An honest (crs^*, π^*): $\mathsf{crs}^* \leftarrow \mathsf{NIZK.G}(1^\lambda)$ and $\pi^* \leftarrow \mathsf{NIZK.P}(\mathsf{crs}^*, y, w)$, or
  - A simulated (crs^*, π^*): $(\mathsf{crs}^*, \tau) \leftarrow \mathsf{NIZK.S}_1(1^\lambda, y)$, and $\pi^* \leftarrow \mathsf{NIZK.S}_2(\mathsf{crs}^*, \tau, y)$
- Set $\pi = \pi^*$ and $\mathsf{crs} = \mathsf{crs}^*$.
- Construct $D := D_{sk_j, \mathsf{crs}}$ with $j = 1$ as defined in (5).
- $\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D)$.
- Set $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$, $\mathsf{pp_d} = \tilde{D}$, and $c = (x, c_1, c_2, \pi)$.
- Output $b' \leftarrow \mathcal{A}_2(st, c, \mathsf{pp_e}, \mathsf{pp_d})$.

By construction, if (crs^*, π^*) is generated honestly, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(0)}_{L_{\mathsf{enc}},\mathcal{A}}$, and if (crs^*, π^*) is simulated, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}$. Therefore, for all (y, w) such that $R_{\mathsf{enc}}(y, w) = 1$ and

infinitely many λ, it holds that

$$\tfrac{1}{p(\lambda)} \leq \left|\Pr[\mathbf{Exp}^{(0)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| = \left|\Pr\left[\begin{matrix}\mathcal{B}(\mathsf{crs},y,\pi) \\ =1\end{matrix} : \begin{matrix}\mathsf{crs}\leftarrow\mathsf{G}(1^\lambda); \\ \pi\leftarrow\mathsf{P}(\mathsf{crs},y,w)\end{matrix}\right] - \Pr\left[\begin{matrix}\mathcal{B}(\mathsf{crs},y,\pi) \\ =1\end{matrix} : \begin{matrix}(\mathsf{crs},\tau)\leftarrow\mathsf{S}_1(1^\lambda,y); \\ \pi\leftarrow\mathsf{S}_2(\mathsf{crs},\tau,y)\end{matrix}\right]\right|$$

We therefore reach a contradiction to the zero-knowledge security of $\mathsf{SSS\text{-}NIZK}$, and conclude that

$$\left|\Pr[\mathbf{Exp}^{(0)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| = \mathsf{negl}(\lambda) \ .$$

A.2 Proof of Proposition 2

Proof. Assume towards contradiction that there exists a non-uniform PPT adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ and a polynomial $p(\cdot)$ such that for infinitely many λ,

$$\left|\Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| \geq \tfrac{1}{p(\lambda)} \ .$$

Then we use $\mathcal{A}$ to construct a non-uniform PPT adversary $\mathcal{B}=(\mathcal{B}_1,\mathcal{B}_2)$ which runs in the CPA security game $\mathbf{Exp}^{\text{CPA-}b}_{\mathcal{B}}(\lambda)$ of PKE (cf. Figure 1) as follows:

$\underline{\mathcal{B}_1(1^\lambda, pk)}$:
- $(x, m_0, m_1, st_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda)$.
- $(sk_1, pk_1) \leftarrow \mathsf{PK.Gen}(1^\lambda)$ and set $pk_2 = pk$.
- $r_1 \leftarrow \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$.
- $c_1 := \mathsf{PK.Enc}(pk_1, (x, m_0); r_1)$.
- Set $m'_0 = (x, m_0)$, $m'_1 = (x, m_1)$, and $st = (sk_1, pk_1, c_1, r_1, x, m_0, m_1, st_{\mathcal{A}})$.
- Output (m'_0, m'_1, st).

$\underline{\mathcal{B}_2(st, c'_b)}$:
- Set $c_2 = c'_b$ and $y = (pk_1, pk_2, c_1, c_2)$.
- $(\mathsf{crs}, \tau) \leftarrow \mathsf{NIZK.S}_1(1^\lambda, y)$.
- $\pi \leftarrow \mathsf{NIZK.S}_2(\mathsf{crs}, \tau, y)$.
- Construct $D := D_{sk_j, \mathsf{crs}}$ with $j = 1$ as defined in (5).
- $\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D)$.
- Set $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$, $\mathsf{pp_d} = \tilde{D}$, and $c = (x, c_1, c_2, \pi)$.
- Output $b' \leftarrow \mathcal{A}_2(st_{\mathcal{A}}, c, \mathsf{pp_e}, \mathsf{pp_d})$.

By construction, if $c'_b \leftarrow \mathsf{PK.Enc}(pk, m'_0)$, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}$, and if $c'_b \leftarrow \mathsf{PK.Enc}(pk, m'_1)$, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}$. Therefore, for infinitely many λ, it holds that

$$\tfrac{1}{p(\lambda)} \leq \left|\Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| = \left|\Pr[\mathbf{Exp}^{\text{CPA-0}}_{\mathcal{B}}(\lambda)=1] - \Pr[\mathbf{Exp}^{\text{CPA-1}}_{\mathcal{B}}(\lambda)=1]\right| \ .$$

We therefore reach a contradiction to the CPA security of PKE, and conclude that

$$\left|\Pr[\mathbf{Exp}^{(1)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| = \mathsf{negl}(\lambda) \ .$$

A.3 Proof of Proposition 3

Proof. Assume towards contradiction that there exists a non-uniform PPT adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ and a polynomial $p(\cdot)$ such that for infinitely many λ,

$$\left|\Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1] - \Pr[\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}},\mathcal{A}}(\lambda)=1]\right| \geq \tfrac{1}{p(\lambda)} \ .$$

Then we use $\mathcal{A}$ to construct a non-uniform PPT adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the indistinguishability security of $i\mathcal{O}$ (cf. (1)) as follows:

$\underline{\mathcal{B}(1^\lambda)}$:
- $(x, m_0, m_1, st) \leftarrow \mathcal{A}_1(1^\lambda)$.
- $(sk_1, pk_1) \leftarrow \mathsf{PK.Gen}(1^\lambda)$ and $(sk_2, pk_2) \leftarrow \mathsf{PK.Gen}(1^\lambda)$.
- $r_1, r_2 \leftarrow \{0,1\}^{\ell_{\mathsf{PK}}(\lambda)}$.
- Set $c_1 = \mathsf{PK.Enc}(pk_1, (x, m_0); r_1)$ and $c_2 = \mathsf{PK.Enc}(pk_2, (x, m_1); r_2)$.
- Set $y = (pk_1, pk_2, c_1, c_2)$.
- $(\mathsf{crs}, \tau) \leftarrow \mathsf{NIZK.S}_1(1^\lambda, y)$.
- $\pi \leftarrow \mathsf{NIZK.S}_2(\mathsf{crs}, \tau, y)$.
- Construct $D_j := D_{sk_j, \mathsf{crs}}$ for $j = 1, 2$ as defined in (5).
- Submit (D_1, D_2) to the $i\mathcal{O}$ challenger to obtain $\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D_j)$.
- Set $\mathsf{pp_e} = (\mathsf{crs}, pk_1, pk_2)$, $\mathsf{pp_d} = \tilde{D}$, and $c = (x, c_1, c_2, \pi)$.
- Output $b' \leftarrow \mathcal{A}_2(st_{\mathcal{A}}, c, \mathsf{pp_e}, \mathsf{pp_d})$.

By construction, if $\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D_1)$, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}}, \mathcal{A}}$, and if $\tilde{D} \leftarrow i\mathcal{O}(1^\lambda, D_2)$, then $\mathcal{B}$ simulates $\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}}, \mathcal{A}}$. Therefore, for infinitely many λ, it holds that

$$\frac{1}{p(\lambda)} \leq \left| \Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}}, \mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}}, \mathcal{A}}(\lambda) = 1] \right| = \left| \Pr[\mathcal{B}(i\mathcal{O}(1^\lambda, D_1)) = 1] - \Pr[\mathcal{B}(i\mathcal{O}(1^\lambda, D_2)) = 1] \right| . \quad (14)$$

Relying on the statistical simulation soundness of $\mathsf{SSS\text{-}NIZK}$, we claim that $D_1 \equiv D_2$. Let $((x', c'_1, c'_2, \pi'), w')$ be an arbitrary input, we show that $D_1 \equiv D_2$ as follows:

- If $x' \notin L$, we have $D_1 \equiv D_2 \equiv \perp$ as for any $w' \in \{0,1\}^*$, it holds that $R(x', w') = 0$.
- If $x' \in L$, observe that $D_1 \not\equiv D_2$ if and only if $\exists m'_0 \neq m'_1$, pk'_1, pk'_2, r'_1, r'_2, and

$$y' := (pk'_1, pk'_2, c'_1 = \mathsf{PK.Enc}(pk'_1, (x', m'_0); r'_1), c'_2 = \mathsf{PK.Enc}(pk'_1, (x', m'_1); r'_2))$$

such that $\mathsf{NIZK.V}(\mathsf{crs}, y', \pi') = 1$. Note that $y' \notin L_{\mathsf{enc}}$, $x \notin L$ and therefore $y \neq y'$. It follows from the statistical simulation soundness of $\mathsf{SSS\text{-}NIZK}$ (cf. (3)) that there exists no such (y', π') such that $\mathsf{NIZK.V}(\mathsf{crs}, y', \pi') = 1$ for a simulated crs.

Together, $D_1 \equiv D_2$ and (14) contradict the security of $i\mathcal{O}$, and we conclude that

$$\left| \Pr[\mathbf{Exp}^{(2)}_{L_{\mathsf{enc}}, \mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Exp}^{(3)}_{L_{\mathsf{enc}}, \mathcal{A}}(\lambda) = 1] \right| = \mathsf{negl}(\lambda) .$$

A.4 Proofs of Propositions 4, 5 and 6

The proofs of Propositions 4, 5 and 6 are analogous to those of Propositions 2, 3 and 1, respectively.