# Offline Witness Encryption with Semi-Adaptive Security

Peter Chvojka[1] Tibor Jager[1] Saqib A. Kakvi[1]

November 20, 2019

[1]Department of Computer Science, Bergische Universität Wuppertal
(chvojka,jager,kakvi)@uni-wuppertal.de

**Abstract**

The first construction of Witness Encryption (WE) by Garg et al. (STOC 2013) has led to many exciting avenues of research in the past years. A particularly interesting variant is Offline WE (OWE) by Abusalah et al. (ACNS 2016), as the encryption algorithm uses neither obfuscation nor multilinear maps.

Current OWE schemes provide only *selective* security. That is, the adversary must commit to their challenge messages $m_0$ and $m_1$ *before* seeing the public parameters.We provide a new, generic framework to construct OWE, which achieves adaptive security in the sense that the adversary may choose their challenge messages adaptively. We call this *semi-adaptive* security, because – as in prior work – the instance of the considered NP language that is used to create the challenge ciphertext must be fixed before the parameters are generated in the security proof. We show that our framework gives the first OWE scheme with *constant* ciphertext overhead even for messages of polynomially-bounded size. We achieve this by introducing a new variant of puncturable encryption defined by Green and Miers (S&P 2015) and combining it with the iO-based approach of Abusalah et al. Finally, we show that our framework can be easily extended to construct the first *Extractable* Offline Witness Encryption (EOWE), by using *extractability* obfuscation of Boyle et al. (TCC 2014) in place of iO, opening up even more possible applications.

The obfuscation is needed only for our public parameters, but its functionality can be realised with a Trusted Execution Environment (TEE), which means we have a very efficient scheme with ciphertexts consisting of only 5 group elements.

**Keywords:** Witness encryption, functional encryption, obfuscation, provable security

An extended abstract of this work appeared in the proceedings of ACNS 2020. This is the full version.

# 1 Introduction

Witness Encryption. Since the seminal paper of Garg, Gentry, Sahai and Waters [GGSW13], witness encryption has enjoyed great attention as a versatile building block for cryptography. Whereas previous public key encryption, itself a special case of witness encryption, required a full key pair to be generated, witness encryption gives more freedom. The encryption key is now a statement $x$ that is (ostensibly) in some NP language, even without knowing a corresponding witness $w$. Any ciphertext produced using $x$ is decryptable by parties in possession of *any* witness $w$ that $x$ is indeed in the language. For example, a quite often mentioned direct application of witness encryption is a prize for solving some NP-hard puzzle. Here the puzzle would be some statement $x$ and an encryption of a password to access the prize. Thus anybody who solves the puzzle, i.e. finds any valid witness $w$, will be able to recover the password and gain their prize.

Applications of Witness Encryption. There are a plethora of further applications for witness encryptions in cryptography, starting with those stated by Garg et al. themselves; but there are several follow-up applications of witness encryption and its variants appear in secure computation [BL18, CDG$^+$17, GLS15, GK18, KSY18], for new primitives [BMSZ16, BH15, BGI14, GKP$^+$13, LJKW18] or indeed novel constructions of known primitives [AJN$^+$16, BGI$^+$17, FNV17].

Variants and Extensions of Witness Encryption. Several interesting variants or extension of witness encryption have been proposed.

- Boyle et al. [BCP14] introduced *Functional* Witness Encryption (FWE), where decryption with witness $w$ reveals not directly the message $m$, but $F(m, w)$ for some function $F$ that may be specified during encryption.
- Abusalah et al. [AFP16] introduced *Offline* Witness Encryption (OWE), where encryption can be performed with classical public key cryptography and only decryption requires obfuscation. This is of particular interest when there is a disparity in the computing powers of the encryptor and decryptor, such as in the case of Asymmetric Password Based Encryption [BH15]. The Offline Witness Encryption scheme of [AFP16] can also be turned into an *Offline Functional* Witness Encryption (OFWE) scheme.
- *Extractable* Witness Encryption additionally guarantees that any party who successfully decrypts a ciphertext encrypted under NP statement $x$ must "know" a corresponding witness $w$. This variant has had immediate applications such as running Turing machines on encrypted data [GKP$^+$13], asymmetric password-based cryptography [BH15], functional signatures [BGI14], secret-sharing for NP [KNY14], and time-lock encryption [LJKW18].

## 1.1 Our Contributions

We introduce a new generic technique to construct Offline Witness Encryption schemes with stronger security and reduced ciphertext overhead. Concretely, we present the first Offline Witness Encryption scheme that achieves *adaptive chosen-message* security. That is, prior work [AFP16] only provided security against adversaries that commit to the "challenge messages" $m_0, m_1$ even before seeing the public parameters of the scheme. This significantly limits the potential use of this interesting primitive in cryptographic applications where messages may be chosen adaptively

by an adversary and thus can depend on the parameters. In contrast, our construction achieves adaptive security, in the sense that the adversary may choose the "challenge messages" adaptively during the security experiment.

On the technical level, we show that puncturable encryption can be used in place of the Naor-Yung [NY90] style double encryption used by Abusalah et al. [AFP16]. The latter requires two ciphertexts and a zero-knowledge proof which can be realised quite efficiently, using a Groth-Sahai-like pairing-based proof system [GS08], but still consists of a rather large number of group elements that grows linearly with the size of the message. In contrast, we show that puncturable encryption yields a significantly more efficient instantiation that requires only one puncturable encryption ciphertext, no zero-knowledge proofs, and has a ciphertext overhead that is constant and independent of the size of the messages.

In summary, our approach significantly extends the security and efficiency, and thereby the applicability, of Witness Encryption schemes in cryptography.

### 1.2 Our Approach

Since its inception, witness encryption has been inexorably linked with multi-linear maps [BS02, BS03, FHHL18, GGH12, GGH13a] and obfuscation [BGI$^+$01, BGI$^+$12, GGH$^+$13b, Had00], which are known to be equivalent [AFH$^+$15, AFH$^+$16, PS15]. In light of that and recent advances [Agr19], we will simply speak of obfuscation. Obfuscation aims to present the adversary with a fully functional program, but have them learn none of the specifics of said program. At first glance, this is very reminiscent of a black box, which was indeed the initial goal of obfuscation. This type of obfuscation, called Virtual Black Box (VBB) obfuscation, was shown to be impossible in general, under reasonable assumptions about the polynomial time hierarchy [BGI$^+$01]. In response to this, weaker forms of obfuscation were suggested, as they could potentially be achievable.

The first of these was indistinguishability obfuscation (iO), which aims to hide the internal workings of a circuit. In essence, here one would seek to hide the exact method of computing a function and not the function itself. Consider two circuits that double the input; one that adds the input to itself and one that multiplies the input by two. iO guarantees that an adversary could not distinguish between an obfuscation of either circuit. Essentially, the adversary could only observe the input/output behaviour but no internal computations. Despite being seemingly benign, iO and its existence have several consequences. The most direct application is to give the adversary access to computations using secret keys without having to reveal the keys themselves. It is exactly this property that we will use to build our witness encryption.

The second ingredient we need is puncturable encryption, which was formalised by Green and Miers [GM15], but was first informally mentioned by Anderson [And97]. The core concept of puncturable encryption is to have a way to make certain ciphertexts "undecryptable", by modifying the decryption key such that the information needed to decrypt *exactly* these ciphertexts is removed. It is imperative to note that the ciphertexts are not changed in any way and should be decryptable up until the decrpytion key is punctured. This primitive has a very direct application in forward-secure instant-messaging [GM15] and so-called 0-RTT protocols with full forward security [DGJ$^+$18, DJSS18, GHJL17]. However, we do not require the full security of puncturable encryption, thus we introduce a new variant of puncturable encryption. We relax the original security definition from *many-time* puncturability to *one-time* puncturability. This essentially means that we can create *exactly one* special secret key that is able to decrypt all ciphertexts encrypted under all but one tag. Note that similar all-but-one techniques have already been used for a long time to construct

CCA-secure encryption schemes. Indeed, we can leverage some of these techniques to give a simple and very efficient instantiation of our primitive and show one specific tag-based encryption due to Kiltz [Kil06] scheme does indeed meet our definition.

We show how to combine these ingredients to get an OWE that is semi-adaptively secure. Technically, we show how to leverage puncturable encryption in the security proof to get adaptivity in the challenge messages. Since we compute our decryption in an obfuscation, we can follow the approach of [AFP16] and we get a generic framework for constructing semi-adaptively secure OWE from the primitives listed above.

Finally, we turn to a variant of iO, namely *extractability obfuscation* (eO) [BCP14]. eO is a variant of iO, where the adversary should not be able to distinguish obfuscations of circuits that differ on only a "sufficiently small" number of inputs. Furthermore, if any adversary is able to do so, then they must "know" an input where these circuits differ, which we can then extract from them. Having this new tool at our disposal, we can convert our OWE into an EOWE.

### 1.3 Application of Semi-Adaptive Offline Witness Encryption

While it is always desirable to have the strongest possible security properties of a scheme, it is not always necessary. In fact, when we are able to use weaker security notation, we often have some performance gains. In fact, we do exactly this later in this paper with puncturable encryption. We will now discuss two of the most interesting applications of semi-adaptive EOWE know to date. As OWE is a relatively new primitive, one can expect more application to follow in the near future.

The first application is the classic puzzle example. In this some party, who we will call the encryptor, offers a sum of money to the first person who solves their puzzle. Classically, the first to solve would contact the encryptor, who would then verify the solution and arrange for the funds to be released to the solver. Of course this requires the encryptor to be available at all times. With an OWE, the encryptor could simply encrypt credentials to a bank account containing the money. It is clear to see that semi-adaptivity is sufficient as the puzzle, the instance, is fixed once and for all.

The second application is the more recent primitive of Time-Lock Encryption (TLE) introduced by Liu et al. [LJKW18]. TLE allows a sender to encrypt a plaintext such that after a certain amount of time has lapsed, the decryption key will be available to all. TLE has several interesting applications including, but not limited to: responsible disclosure, pre-distribution of digital media, sealing of auctions tenders and publication of grades. Here we see that semi-adaptive security is sufficient as the instance is the release time, which is fixed at the time of encryption. In the construction of Liu et al. [LJKW18] this is a specific, as yet unpublished block in some public blockchain.

### 1.4 Open Problems

In addition to messages, encryption also depends on an instance $x$ that may or may not lie in a given NP language. The schemes from [AFP16] and ours both require that the challenge *instance* $x$ is fixed at the beginning of the game, and thus we say they are both *instance selective*. Therefore, we say that our framework provides *semi-adaptive security*, as compared to the known *selectively secure* scheme [AFP16]. While it seems that semi-adaptive security is sufficient for many applications, we leave it as an interesting open problem to construct offline witness encryption schemes with *fully-adaptive* security.

### 1.5 Related Work

Functional Witness Encryption was introduced by Boyle, Chung and Pass [BCP14], but they did not provide a concrete construction. They instead showed how to construct Functional Witness Encryption, which is not Offline, from an extractability obfuscator (eO). The only other known construction is the one due to Abusalah, Fuchsbauer and Pietrzak, who presented a concrete construction of an Offline Witness Encryption [AFP16]. However, their scheme only achieves selective security, whereas our scheme has semi-adaptive security. Furthermore, their construction requires more complex and involved primitives such as simulation secure non-interactive zero knowledge proofs. We are able to build our scheme with simpler components, thus giving us a smaller ciphertexts, while still reach a higher level of security. We believe the construction of [AFP16] can be adapted to yield an Extractable Offline Witness Encryption scheme, by replacing the indisiguishabilty obfuscator (iO) with an extractability obfuscator (eO), in a similar manner to our scheme. However, this transformation still results in a selectively secure scheme, while ours is semi-adaptively secure.

In a similar line of work, Zhandry presents a primitive called *Reusuable* Witness Encryption [Zha16]. This construction is close to Offline Witness Encryption, but does not immediately achieve the definition of Offline Witness Encryption. In addition, the construction is built in a KEM/DEM manner, it cannot be extended to a *Functional* Witness Encryption, thus we will not compare our work to this construction.

## 2 Preliminaries

### 2.1 Notations and conventions

We denote our security parameter as $\lambda$. For all $n \in \mathbb{N}$, we denote by $1^n$ the $n$-bit string of all ones. For any element $x$ in a set $S$, we use $x \in_R S$ to indicate that we choose $x$ uniformly random in $S$. For any other distribution $\mathfrak{D}$ on $S$, we use $x \in_{\mathfrak{D}} S$ to indicate $x$ is sampled from $S$ according to the distribution $\mathfrak{D}$. All algorithms may be randomized. For any algorithm $A$, we define $x \xleftarrow{\$} A(a_1, \ldots, a_n)$ as the execution of $A$ with inputs $a_1, \ldots, a_n$ and fresh randomness and then assigning the output to $x$.

### 2.2 Offline Witness Encryption

To discuss Offline Witness Encryption, we start from the ground up. Witness encryption, which is generalisation of public key encryption where anybody in possession of a valid witness $w$ that some statement $x$ is in a specified language can decrypt all ciphertexts encrypted under $x$. Witness encryption was extended to *Offline* Witness Encryption (OWE) by Abusalah, Fuchsbauer and Pietrzak [AFP16]. The main idea is to move most of the heavy computations away from the encryption into a setup algorithm (and possibly some into the decryption). This allows us to leverage WE into scenarios where there is a discrepancy in computing power between encryptor and decryptor, as is quite often the case. We now recall the definition of OWE.

**Definition 1.** An offline witness encryption scheme $\mathtt{OWE}$ for some language $\mathfrak{L}$ is defined as a triple of probabilistic polynomial time (PPT) algorithms $\mathtt{OWE} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$:

- $\mathsf{Setup}$ takes as input the unary representation of our security parameter $1^\lambda$ and outputs parameters for encryption $\mathsf{pp}_e$ and for decryption $\mathsf{pp}_d$.

- $\mathsf{Encrypt}$ takes as an input the encryption parameters $\mathsf{pp}_e$, an instance $x$ and a message $m$ and outputs a ciphertext $c$.
- $\mathsf{Decrypt}$ takes as input the decryption parameters $\mathsf{pp}_d$, a ciphertext $c$ and a witness $w$ and outputs $m$ if $(x, w) \in \mathfrak{L}$ and $\perp$ otherwise.

We say $\mathtt{OWE}$ is correct if for all messages $m$ and for all pairs $(x, w) \in \mathfrak{L}$, we have:

$$\Pr[\mathsf{Decrypt}(\mathsf{pp}_d, \mathsf{Encrypt}(\mathsf{pp}_e, x, m), w) = m] = 1$$

The security of OWE is given by the following experiment. This experiment defines security in the semi-adaptive setting, where the adversary must commit to the instance, but not the messages or functions, before seeing parameters.

$\mathsf{ExpOWE}_{\mathcal{A}}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}(1^\lambda)$:

$x \xleftarrow{\$} \mathcal{A}(1^\lambda)$
$(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathtt{OWE.Setup}(1^\lambda)$
$(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathsf{pp}_e, \mathsf{pp}_d)$
$b \in_R \{0,1\}; c^* \xleftarrow{\$} \mathtt{OWE.Encrypt}(\mathsf{pp}_e, x, m_b)$
$b' \xleftarrow{\$} \mathcal{A}(c^*)$
return $(b' = b \wedge x \notin \mathfrak{L})$

**Definition 2.** An offline witness encryption scheme $\mathtt{OWE}$ for some language $\mathfrak{L}$ with corresponding relation $R$ is said to be $(t, \varepsilon)$-semi-adaptively secure, if for any adversary $\mathcal{A}$ running in time at most $t$ holds:

$$\mathbf{Adv}_{\mathcal{A}}^{\mathtt{OWE}} = \left| \Pr[\mathsf{ExpOWE}_{\mathcal{A}}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}(1^\lambda) = 1] - \frac{1}{2} \right| \leq \varepsilon.$$

We now discuss *Extractable* Offline Witness Encryption (EOWE). We have defined OWE, which gives us our functionality, we now further require our scheme to also provide us with security, more specifically *extractable* security as introduced by Goldwasser et al [GKP+13]. Broadly speaking, if an adversary can distinguish between the encryptions of two messages under some instance $x$, then it must "know" a witness $w$ to the instance $x$, which we can then extract. We now present the definition of extractable security for OWE. We begin by defining the following experiment:

$\mathsf{ExpEOWE}_{\mathcal{A}}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}(1^\lambda, x)$:

$(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathtt{OWE.Setup}(1^\lambda)$
$(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$
$b \in_R \{0,1\}; c^* \xleftarrow{\$} \mathtt{OWE.Encrypt}(\mathsf{pp}_e, x, m_b)$
$b' \xleftarrow{\$} \mathcal{A}(c^*)$
return $b' = b$

**Definition 3.** An offline witness encryption scheme $\mathtt{OWE}$ for some language $\mathfrak{L}$ with corresponding relation $R$ and a class of functions computable by $\mathfrak{C}_\lambda$ is said to be $(\varepsilon, \alpha, \mathfrak{p})$-extractable secure, if for any adversary $\mathcal{A}$, there exists a PPT extractor $\mathcal{E}$ and negligible function $\alpha$ such that for all $x \in \{0,1\}^*$ holds: If

$$\Pr[\mathsf{ExpEOWE}_{\mathcal{A}}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}(1^\lambda, x) = 1] = \frac{1}{2} + \varepsilon > \frac{1}{2} + \alpha(\lambda),$$

then $\mathcal{E}(1^\lambda, x)$ output a witness $w$ such that $R(x, w) = 1$ with non-negligible probability in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$, where $\mathfrak{p}$ is a polynomial.

Now that we have the definitions of our goals, we can work towards building them. To this end, we now recall the definitions of the building blocks we need to achieve our goal. In the following subsections, we will build up the definitions of primitives that we will need to achieve OWE and EOWE.

### 2.3 Obfuscation

A key ingredient of our construction is obfuscation. In recent years, there has been a large amount of research in the field of obfuscation, in particular the various flavours of obfuscation. The idea behind obfuscation is to take an arbitrary program circuit and turn into a black box, which we can give to the adversary. While highly desirable, this proved somewhat difficult, if not impossible. To deal with this, several alternative weaker definitions were proposed. In all our definitions, we will consider a class of circuits $\mathfrak{C}_\lambda$, which consists of circuits with $n$-bit inputs and single bit outputs of size bounded by $poly(\lambda)$.

We first recall the definitions of an *indistinguishability obfuscator* ($i\mathcal{O}$) due to Barak et al. [BGI+01]. The main idea behind $i\mathcal{O}$ is that we hide the exact steps taken to compute our circuit, but not the input/output behaviour. This flavour of obfuscation is somewhat reminiscent of indistinguishablity games employed in encryption.

**Definition 4.** A uniform PPT machine $i\mathcal{O}$ is called an *$\varepsilon$-indistinguishability obfuscator* for a circuit class $\mathfrak{C}_\lambda$ if the following conditions are met:

- **Preserving Functionality:** Let $\widetilde{C} = i\mathcal{O}(C)$, then we have $\forall C \in \mathfrak{C}_\lambda, \forall x \in \{0,1\}^n$, $\widetilde{C}(x) = C(x)$
- **Indistinguishability:** $\forall C_0, C_1 \in \mathfrak{C}_\lambda$ such that for all inputs $x \in \{0,1\}^n$, we have $C_0(x) = C_1(x)$, the following holds for all PPT distinguishers $\mathcal{D}$:

$$\Pr[\mathcal{D}(i\mathcal{O}(C_0)) = 1] - \Pr[\mathcal{D}(i\mathcal{O}(C_1)) = 1] \leq \varepsilon$$

We further need to consider a relaxtion of indistinguishabilty obfuscation, specifically *differing input obfuscator* ($di\mathcal{O}$). Here instead of requiring that the circuits have identical behaviour, we require that they have *near identical* behaviour. That is to say, the output may differ for a "small" number of inputs, which is determined by the application. The idea being, that it should be hard for an adversary to find some $x' \in \{0,1\}^n$ such that $C_0(x') \neq C_1(x')$. Given such an $x'$, distinguishing between $di\mathcal{O}(C_0)$ and $di\mathcal{O}(C_1)$ is easy. Given $\widetilde{C} = di\mathcal{O}(C_b)$, we output 0 if $\widetilde{C}(x') = C_0(x')$ and 1 otherwise. In this case, it would be useful to extract such a value from this adversary. To this end, Boyle, Chung and Pass [BCP14] define *extractabity obfuscation* ($e\mathcal{O}$), which they also show to be equivalent to $di\mathcal{O}$. We will consider a small variant, specifically we consider $e\mathcal{O}$ with distributional auxiliary input, whose definition we now recall.

**Definition 5.** A uniform PPT machine $e\mathcal{O}$ is called an *extractability obfuscator w.r.t. distributional auxiliary input* for a circuit class $\mathfrak{C}_\lambda$ if the following conditions are met:

- **Preserving Functionality:** Let $\widetilde{C} = e\mathcal{O}(1^\lambda, C)$, then we have $\forall C \in \mathfrak{C}_\lambda, \forall x \in \{0,1\}^\lambda$, $\widetilde{C}(x) = C(x)$.

- **Extractability:** for all PPT distinguishers $\mathcal{D}$ and for every efficiently samplable distribution $\mathfrak{D}$ over $\mathfrak{C}_\lambda \times \mathfrak{C}_\lambda \times \{0,1\}^*$ there is a PPT extractor $\mathcal{E}$ and negligible function $\alpha$ such that for every $\lambda \in \mathbb{Z}$, it holds with overwhelming probability over $(C_0, C_1, z) \in_{\mathfrak{D}} \mathfrak{C}_\lambda \times \mathfrak{C}_\lambda \times \{0,1\}^*$: If

$$\varepsilon = \left|\Pr[\mathcal{D}(\mathit{e}\mathcal{O}(1^\lambda, C_0), C_0, C_1, z) = 1] - \Pr[\mathcal{D}(\mathit{e}\mathcal{O}(1^\lambda, C_1), C_0, C_1, z) = 1]\right| > \alpha(\lambda),$$

then $\mathcal{E}(C_0, C_1, z)$ outputs an input on which $C_0, C_1$ differ with non-negligible probability in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$, where $\mathfrak{p}$ is a polynomial. We call $\mathit{e}\mathcal{O}$ a $(\varepsilon, \alpha, \mathfrak{p})$-extractability obfuscator.

Our definition mildy deviates from that of Boyle, Chung and Pass [BCP14], who require that $\alpha(\lambda)$ is a polynomial. We do this to come closer in line with the original definition of Barak et al. [BGI+12]. We again stress that our auxiliary input distribution bypasses the negative results of [GGHW14, BP15]. Furthermore, due to the fact that our auxiliary input is partially determined by the adversary, namely the instance $x$, we can not fully relax our requirement to Public-Coin Differing-Input Obfuscation [IPS15].

### 2.4 Puncturable Tag-Based Encryption

One of the key components of our construction is our new variant of puncturable encryption [GM15]. Puncturable encryption is an extension of standard public key encryption, where we are able to make certain ciphertexts "undecryptable". This is achieved by modifying the secret key such that it is able to decrypt all ciphertexts, except the ones we have "punctured". In particular, we consider the tag-based variant of puncturable encryption. We have that both encryption and decryption require an additional value, called the tag. In particular, a ciphertext encrypted with some tag $t$ can only be decrypted with the same tag $t$. The secret key can be punctured at specific tags, thus making ciphertexts under those tags undecryptable.

However, we do not need the full security of puncturable encryption, thus we introduce a restricted variant of the original definition. In the original definition, a key could be repeatedly punctured making ciphertexts under several tags "undecryptable". We will only require that our original key can be punctured exactly once at exactly one tag, i.e. one-time puncturablity. Additionally, we allow for an alternative decryption algorithm for punctured keys, which allows to have keys of a different form. We now define puncturable encryption.

**Definition 6.** A tag-based encryption scheme $\mathtt{PE}$ for message space $\mathfrak{M}$ and a tag space $\mathfrak{T}$ is defined as a quintuple of probabilistic polynomial time (PPT) algorithms
$\mathtt{PE} = (\mathsf{KeyGen}, \mathsf{Puncture}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{PunctDec})$:

- $\mathsf{KeyGen}$ takes as input the unary representation of our security parameter $1^\lambda$ and outputs public key $\mathsf{pk}$ and an unpunctured secret key $\mathsf{sk}$.
- $\mathsf{Puncture}$ takes as input an unpunctured secret key $\mathsf{sk}$ and a single tag $t^* \in \mathfrak{T}$ and outputs a key punctured at exactly $t^*$, denoted by $\mathsf{sk}_{t^*}$.
- $\mathsf{Encrypt}$ takes as an input the public key $\mathsf{pk}$, a message $m \in \mathfrak{M}$, a tag $t \in \mathfrak{T}$ and outputs a ciphertext $c$.
- $\mathsf{Decrypt}$ takes as input the unpunctured secret key $\mathsf{sk}$, a ciphertext $c$ and a tag $t \in \mathfrak{T}$ and outputs $m \in \mathfrak{M}$ or $\perp$.

- $\mathsf{PunctDec}$ takes as input a punctured secret key $\mathsf{sk}_{t^*}$, a ciphertext $c$ and a tag $t \neq t^*$ and outputs $m \in \mathfrak{M}$ or $\perp$.

We say $\mathtt{PE}$ is correct if we have that for all key pairs $(\mathsf{pk}, \mathsf{sk}) \leftarrow_{\$} \mathsf{KeyGen}(1^\lambda)$, all messages $m \in \mathfrak{M}$ and for all tags $t \in \mathfrak{T}$, we have:

$$\Pr[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, m, t), t) = m] = 1$$

and further for any tag $t^* \in \mathfrak{T}$, all punctured keys $\mathsf{sk}_{t^*} \xleftarrow{\$} \mathsf{Puncture}(\mathsf{sk}, t^*)$ and all tags $t \neq t^*$, we have

$$\Pr[\mathsf{PunctDec}(\mathsf{sk}_{t^*}, \mathsf{Encrypt}(\mathsf{pk}, m, t), t) = m] = 1$$

For our construction, we require a relatively weak security, namely *selective* indistinguishabilty from random. We define the following experiment:

$\mathsf{ExpPE}^{\mathsf{b}}_{\mathcal{A}}(1^\lambda)$:
$t^* \xleftarrow{\$} \mathcal{A}(1^\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda); \mathsf{sk}_{t^*} \xleftarrow{\$} \mathsf{Puncture}(\mathsf{sk}, t^*)$
$(m^*) \xleftarrow{\$} \mathcal{A}(\mathsf{pk}, \mathsf{sk}_{t^*})$
$r \in_R \mathfrak{M}; c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pk}, (m^* + br))$
$b' \xleftarrow{\$} \mathcal{A}(c^*)$
return $b'$

We say a one-time puncturable encryption scheme $\mathtt{PE}$ is $(t, \varepsilon)$-*selective indistinguishable*, if for all PPT adversaries $\mathcal{A}$ running in time at most $t$ we have

$$\mathbf{Adv}^{\mathtt{PE}}_{\mathcal{A}} \quad = \quad \left|\Pr[\mathsf{ExpPE}^{\mathsf{0}}_{\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{ExpPE}^{\mathsf{1}}_{\mathcal{A}}(1^\lambda) = 1]\right| \leq \varepsilon.$$

Note here that the adversary does not have access to a decryption oracle, as opposed to the security definitions of Green and Miers. This is due to the fact that we only consider puncturing at a single tag. As the adversary is already given the punctured decryption key, which allows them to decrypt arbitrary ciphertexts that are not encrypted under the target tag $t^*$. In particular, this means that the adversary cannot decrypt the challenge ciphertext $c^*$ and trivially win.

# 3 Offline Witness Encryption Construction

In this section we provide a construction and security proof of our offline witness encryption. Let $\mathtt{PE} = (\mathtt{PE.KeyGen}, \mathtt{PE.Encrypt}, \mathtt{PE.Puncture}, \mathtt{PE.Decrypt})$ be a one-time puncturable encryption and $i\mathcal{O}$ an indistinguishablity obfuscator for a circuit class $\mathfrak{C}_\lambda$. Our construction of an Offline Witness Encryption $(\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ for a language $\mathfrak{L}$ is given in Figure 1. We assume that the decryption circuit is padded to maximal length of sizes of all circuits appearing in the security proof, hence, all circuits have the same size. For completeness we state the construction below.

*Remark* 1. Normally to encrypt large messages, one must break the message into appropriate sized blocks and encrypt each block as a separate message. This means for a message of $N$ blocks, we must produce $N$ cipertexts. However, we can bypass this by using our OWE as a Key Encapsulation

| $\underline{\mathsf{Setup}(1^\lambda)}$ | $\underline{C_{\mathsf{sk}}(c, w)}$ |
|---|---|
| $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathtt{PE.KeyGen}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $\widetilde{C}_{\mathsf{sk}} \xleftarrow{\$} i\mathcal{O}(1^\lambda, C_{\mathsf{sk}})$ | if $R(t, w) = 1$ |
| $\mathsf{pp}_e := \mathsf{pk},\ \mathsf{pp}_d := \widetilde{C}_{\mathsf{sk}}$ | $\quad m \leftarrow \mathtt{PE.Decrypt}(\mathsf{sk}, c_{pe}, t)$ |
| return $(\mathsf{pp}_e, \mathsf{pp}_d)$ | $\quad$ return $m$ |
| | else |
| | $\quad$ return $\perp$ |
| $\underline{\mathsf{Encrypt}(\mathsf{pp}_e, x, m)}$ | $\underline{\mathsf{Decrypt}(\mathsf{pp}_d, c, w)}$ |
| $c_{pe} \xleftarrow{\$} \mathtt{PE.Encrypt}(\mathsf{pp}_e, m, x)$ | return $m \leftarrow \widetilde{C}_{\mathsf{sk}}(c, w)$ |
| return $c \leftarrow (c_{pe}, x)$ | |

Figure 1: Construction of OWE

Mechanism (KEM) and encrypt a random key $\kappa$ for a symmetric block cipher. We can then encrypt our message using the symmetric block cipher with key $\kappa$, which is our Data Encapsulation Mechanism (DEM). This gives us a final ciphertext size of one OWE ciphertext and $N$ DEM ciphertext blocks.

*Theorem* 1. *Assume* $\mathtt{PE} = (\mathtt{PE.KeyGen}, \mathtt{PE.Encrypt}, \mathtt{PE.Puncture}, \mathtt{PE.Decrypt})$ *is* $(t, \varepsilon_{\mathtt{PE}})$*-selective indistinguishable from random one-time puncturable encryption and* $i\mathcal{O}$ *is a* $\varepsilon_{i\mathcal{O}}$*-indistinguishability obfuscator. Then* $(\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ *defined in Figure 1 is a* $(t, \varepsilon)$*-semi-adaptively secure offline witness encryption for* $\varepsilon \leq \varepsilon_{i\mathcal{O}} + \varepsilon_{\mathtt{PE}}$.

Proof. Correctness of the scheme is implied by correctness of the puncturable encryption scheme and the indistinguishability obfuscator. To prove security we define a series of games $\mathsf{G}_0 - \mathsf{G}_2$ which are computationally indistinguishable. Individual games differ in how we realize our setup and decryption circuit.

**Game 0.** Game $\mathsf{G}_0$ (Figure 2) corresponds to original security experiment, where we use the $\mathsf{Setup}$, $\mathsf{Encrypt}$ and $C_{\mathsf{sk}}$ directly from our construction.

| $\underline{\mathsf{G}_0(1^\lambda)}$ | $\underline{C_{\mathsf{sk}}(c, w)}$ |
|---|---|
| $x \xleftarrow{\$} \mathcal{A}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ | if $R(t, w) = 1$ |
| $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathsf{pp}_e, \mathsf{pp}_d)$ | $\quad m \leftarrow \mathtt{PE.Decrypt}(\mathsf{sk}, c_{pe}, t)$ |
| $b \in_R \{0, 1\}$ | $\quad$ return $m$ |
| $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pp}_e, x, m_b)$ | return $\perp$ |
| $b' \xleftarrow{\$} \mathcal{A}(c^*)$ | |
| return $(b' = b \wedge x \notin \mathfrak{L})$ | |

Figure 2: Game $\mathsf{G}_0$

We will now define an alternative setup algorithm $\mathsf{Setup}'$. This algorithm differs from $\mathsf{Setup}$ in that we additionally puncture the key $\mathsf{sk}$ on the challenge tag $x$.

$\underline{\mathsf{Setup}'(1^\lambda, x)}$
$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathtt{PE.KeyGen}(1^\lambda)$
$\mathsf{sk}^* \xleftarrow{\$} \mathtt{PE.Puncture}(\mathsf{sk}, x)$
$\widetilde{C}_{\mathsf{sk}^*,x} \xleftarrow{\$} i\mathcal{O}(1^\lambda, C_{\mathsf{sk}^*,x})$
$\mathsf{pp}_e := \mathsf{pk},\ \mathsf{pp}_d := \widetilde{C}_{(\mathsf{sk}^*,x)}$
return $(\mathsf{pp}_e, \mathsf{pp}_d)$

Figure 3: Alternative setup

**Game 1.** In $\mathsf{G}_1$ (Figure 4) we now run our alternative setup algorithm $\mathsf{Setup}'$, which punctures the key $\mathsf{sk}$ on the tag $x$. The decryption circuit now uses the punctured key $\mathsf{sk}^*$ and returns $\perp$ if target tag $x$ is equal to tag $t$ of the ciphertext.

| $\underline{\mathsf{G}_1(1^\lambda)}$ | $\underline{C_{\mathsf{sk}^*,x}(c, w)}$ |
|---|---|
| $x \xleftarrow{\$} \mathcal{A}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathsf{Setup}'(1^\lambda, x)$ | if $R(t, w) = 1$ |
| $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathsf{pp}_e, \mathsf{pp}_d)$ | if $x = t$ |
| $b \in_R \{0, 1\}$ | return $\perp$ |
| $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pp}_e, x, m_b)$ | $m \leftarrow \mathtt{PE.PunctDec}(\mathsf{sk}^*, c_{pe}, t)$ |
| $b' \xleftarrow{\$} \mathcal{A}(c^*)$ | return $m$ |
| return $(b' = b \wedge x \notin \mathfrak{L})$ | return $\perp$ |

Figure 4: Game $\mathsf{G}_1$

*Lemma* 1. $|\Pr[\mathsf{G}_0 = 1] - \Pr[\mathsf{G}_1 = 1]| = \mathbf{Adv}^{i\mathcal{O}}_{\mathcal{B}}$

Proof. For purpose of contradiction assume that there is an attacker $\mathcal{A}$ against our OWE that plays either game $\mathsf{G}_0$ or game $\mathsf{G}_1$. We construct an adversary $\mathcal{B}$ which breaks indistinguishability security of $i\mathcal{O}$.

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

1. Run the adversary $x \xleftarrow{\$} \mathcal{A}(1^\lambda)$.
2. Generate $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathtt{PE.KeyGen}(1^\lambda)$.
3. Puncture the key $\mathsf{sk}^* \xleftarrow{\$} \mathtt{PE.Puncture}(\mathsf{sk}, x)$.
4. Construct $C_0 := C_{\mathsf{sk}}, C_1 := C_{\mathsf{sk}^*,x}$.
5. Submit $C_0, C_1$ to indistinguishability obfuscator $\widetilde{C} \xleftarrow{\$} i\mathcal{O}(C_i)$.
6. Run the adversary $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d)$ where $\mathsf{pp}_e := \mathsf{pk},\ \mathsf{pp}_d := \widetilde{C}$.
7. Randomly pick $b \in_R \{0, 1\}$ and produce $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pk}, x, m_b)$.
8. Run $b' \xleftarrow{\$} \mathcal{A}(c^*)$.

9. Return $b' = b$.

If $\widetilde{C} \xleftarrow{\$} i\mathcal{O}(C_0)$, then $\mathcal{B}$ simulates $\mathsf{G}_0$, otherwise it simulates $\mathsf{G}_1$. Moreover, both circuits have the same input/output behaviour. Circuit $C_{\mathsf{sk}^*,x}$ can potentially differ from $C_{\mathsf{sk}}$ only on inputs where $t = x$ and in this case $C_{\mathsf{sk}^*,x}$ outputs $\perp$. However, $\mathcal{A}$ has to provide $x \notin L$ and that means for $t = x$: $R(t, w) = 0$ and hence $C_{\mathsf{sk}}$ outputs $\perp$ too. Thus it must hold

$$|\Pr[\mathsf{G}_0 = 1] - \Pr[\mathsf{G}_1 = 1]| = |\Pr[\mathcal{B}(i\mathcal{O}(C_0)) = 1] - \Pr[\mathcal{B}(i\mathcal{O}(C_1)) = 1]|\,.$$

Thus, we can conclude that $|\Pr[\mathsf{G}_0 = 1] - \Pr[\mathsf{G}_1 = 1]| = \mathbf{Adv}^{i\mathcal{O}}_{\mathcal{B}}$ as required.

□

**Game 2.** Finally, in $\mathsf{G}_2$ (Figure 5) we encrypt a random message $m$.

| $\underline{\mathsf{G}_2(1^\lambda)}$ | $\underline{C_{\mathsf{sk}^*,x}(c, w)}$ |
|---|---|
| $x \xleftarrow{\$} \mathcal{A}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathsf{Setup}'(1^\lambda, x)$ | if $R(t, w) = 1$ |
| $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$ | if $x = t$ |
| $b \in_R \{0, 1\}$ | return $\perp$ |
| $\boxed{m \in_R \mathfrak{M}, \text{ s.t. } \lvert m\rvert = \lvert m_0\rvert}$ | $m \xleftarrow{\$} \mathtt{PE}.\mathsf{PunctDec}(\mathsf{sk}^*, c_{pe}, t)$ |
| $\boxed{c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pp}_e, x, m)}$ | return $m$ |
| $b' \xleftarrow{\$} \mathcal{A}(c^*)$ | return $\perp$ |
| return $(b' = b \wedge x \notin \mathfrak{L})$ | |

Figure 5: Game $\mathsf{G}_2$

*Lemma* 2. $|\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]| = \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}}$.

Proof. Assume that there is an adversary $\mathcal{A}$ which can distinguish games $\mathsf{G}_1$ and $\mathsf{G}_2$. We construct an adversary $\mathcal{B}$ which breaks security of the puncturable encryption scheme.

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

1. Send $x$ to the challenger.
2. Receive public key $\mathsf{pk}$ and punctured secret key $\mathsf{sk}^*$ from the challenger.
3. Construct obfuscation of the circuit $C_{\mathsf{sk}^*,x}$ and run adversary $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$ where $\mathsf{pp}_e := \mathsf{pk}$, $\mathsf{pp}_d := \widetilde{C}_{\mathsf{sk}^*,x}$.
4. Randomly pick $\hat{b} \in_R \{0, 1\}$, sends $m_{\hat{b}}$ to challenger and obtains ciphertext $c^*$ as a response.
5. Run $b' \xleftarrow{\$} \mathcal{A}(c^*)$.
6. Return $b' = \hat{b}$.

It is clear to see that if challenger in $\mathsf{ExpPE}^{\mathsf{b}}_{\mathcal{A}}(1^\lambda)$ picks $b = 0$, then $c^*$ is an encryption of $m_{\hat{b}}$ and $\mathcal{B}$ perfectly simulates game $\mathsf{G}_1$, otherwise it perfectly simulates $\mathsf{G}_2$. Hence, we have

$$\mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} = |\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]|\,.$$

□

*Lemma* 3. $\Pr[\mathsf{G}_2 = 1] = 1/2$.

Proof. Now that we encrypt a random message, the challenge ciphertext $c^*$ is independent of our choice of $b$. Thus, the adversary gets no information about $b$ and can do no better than guessing and has a success probability of exactly $\frac{1}{2}$. □

Combining Lemmas 5-7 we obtain following:

$$\begin{aligned}\Pr[\mathsf{ExpOWE}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}_{\mathcal{A}}(1^\lambda) = 1] &= \Pr[\mathsf{G}_0 = 1] \\ &\leq |\Pr[\mathsf{G}_0 = 1] - \Pr[\mathsf{G}_1 = 1]| \\ &\quad + |\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]| \\ &\quad + \Pr[\mathsf{G}_2 = 1] \\ &= \mathbf{Adv}^{i\mathcal{O}}_{\mathcal{B}} + \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} + \frac{1}{2}\end{aligned}$$

Hence, it holds

$$\mathbf{Adv}^{\mathtt{OWE}}_{\mathcal{A}} \leq \mathbf{Adv}^{i\mathcal{O}}_{\mathcal{B}} + \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}},$$

which concludes our proof. □

# 4 Extractable Offline Witness Encryption

Now that we have constructed OWE in Section 3, we now require our scheme to also provide us with *extractable* security as introduced by Goldwasser et al [GKP+13]. Broadly speaking, if an adversary can distinguish between the encryptions of two messages under some instance $x$, then it must "know" a witness $w$ to the instance $x$, which we can then extract. We now present the definition of extractable security for OWE. We begin by defining the following experiment:

$\mathsf{ExpEOWE}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}_{\mathcal{A}}(1^\lambda, x)$:

$(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathtt{OWE.Setup}(1^\lambda)$
$(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$
$b \in_R \{0,1\}; c^* \xleftarrow{\$} \mathtt{OWE.Encrypt}(\mathsf{pp}_e, x, m_b)$
$b' \xleftarrow{\$} \mathcal{A}(c^*)$
return $b' = b$

*Definition* 7. An offline witness encryption scheme OWE for some language $\mathfrak{L}$ with corresponding relation $R$ and a class of functions computable by $\mathfrak{C}_\lambda$ is said to be $(\varepsilon, \alpha, \mathfrak{p})$-extractable secure, if

for any adversary $\mathcal{A}$, there exists a PPT extractor $\mathcal{E}$ and negligible function $\alpha$ such that for all $x \in \{0,1\}^*$ holds: If

$$\Pr[\mathsf{ExpEOWE}_{\mathcal{A}}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}(1^\lambda, x) = 1] = \frac{1}{2} + \varepsilon > \frac{1}{2} + \alpha(\lambda),$$

then $\mathcal{E}(1^\lambda, x)$ output a witness $w$ such that $R(x,w) = 1$ with non-negligible probability in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$, where $\mathfrak{p}$ is a polynomial.

### 4.1 Construction

In this section we provide a construction of our Extractable Offline Witness Encryption. The scheme is actually similar to the construction of OWE. The principal difference is that in EOWE we use an extractability obfuscator $e\mathcal{O}$ instead of an indistinguishability obfuscator $i\mathcal{O}$. Let $\mathtt{PE} = (\mathtt{PE.KeyGen}, \mathtt{PE.Encrypt}, \mathtt{PE.Puncture}, \mathtt{PE.Decrypt})$ be a puncturable encryption and $e\mathcal{O}$ an extractability obfuscator for a circuit class $\mathfrak{C}_\lambda$. Our construction of an extractable offline witness encryption $(\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ for a language $\mathfrak{L}$ is given in Figure 6. We assume that the decryption circuit is padded to maximal length of sizes of all circuits appearing in the security proof, hence, all circuits have the same size.

| $\underline{\mathsf{Setup}(1^\lambda)}$ | $\underline{C_{\mathsf{sk}}(c,w)}$ |
|---|---|
| $(\mathsf{sk},\mathsf{pk}) \xleftarrow{\$} \mathtt{PE.KeyGen}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $\widetilde{C}_{\mathsf{sk}} \xleftarrow{\$} e\mathcal{O}(1^\lambda, C_{\mathsf{sk}})$ | if $R(t,w) = 1$ |
| $\mathsf{pp}_e := \mathsf{pk}$, $\mathsf{pp}_d := \widetilde{C}_{\mathsf{sk}}$ | $\quad m \leftarrow \mathtt{PE.Decrypt}(\mathsf{sk}, c_{pe}, t)$ |
| return $(\mathsf{pp}_e, \mathsf{pp}_d)$ | $\quad$ return $m$ |
| | return $\perp$ |
| $\underline{\mathsf{Encrypt}(\mathsf{pp}_e, x, m)}$ | $\underline{\mathsf{Decrypt}(\mathsf{pp}_d, c, w)}$ |
| $c_{pe} \xleftarrow{\$} \mathtt{PE.Encrypt}(\mathsf{pp}_e, m, x)$ | return $m \leftarrow \widetilde{C}_{\mathsf{sk}}(c,w)$ |
| return $c \leftarrow (c_{pe}, x)$ | |

Figure 6: Construction of EOFWE

*Theorem* 2. *Assume* $\mathtt{PE} = (\mathtt{PE.KeyGen}, \mathtt{PE.Encrypt}, \mathtt{PE.Puncture}, \mathtt{PE.Decrypt})$ *is* $(t, \varepsilon_{\mathtt{PE}})$*-selective indistinguishable from random puncturable encryption and* $e\mathcal{O}$ *is a* $(\varepsilon_{e\mathcal{O}}, \alpha_{e\mathcal{O}}, \mathfrak{p})$*-extractability obfuscator. Then* $(\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ *defined in Figure 6 is* $(\varepsilon, \alpha, \mathfrak{p})$*-extractable secure offline witness encryption for* $\varepsilon \geq \varepsilon_{e\mathcal{O}} + \varepsilon_{\mathtt{PE}} + \alpha_{e\mathcal{O}}(\lambda)$ *and* $\alpha = \alpha_{e\mathcal{O}}$.

Proof. Correctness of the scheme is implied by correctness of the puncturable encryption scheme and the extractability obfuscator. To prove security we define a series of games $\mathsf{G}_0 - \mathsf{G}_3$ which are computationally indistinguishable or allow us to extract a witness. Individual games differ in how we realize our setup algorithm and decryption circuit.

**Game 0.** Game $\mathsf{G}_0$ (Figure 7) corresponds to original security experiment, where we use the $\mathsf{Setup}$, $\mathsf{Encrypt}$ and $C_{\mathsf{sk}}$ directly from our construction.

| $\underline{\mathsf{G}_0(1^\lambda, x)}$ | $\underline{C_{\mathsf{sk}}(c, w)}$ |
|---|---|
| $(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$ | if $R(t, w) = 1$ |
| $b \in_R \{0, 1\}$ | $m \leftarrow \mathsf{PE.Decrypt}(\mathsf{sk}, c_{pe}, t)$ |
| $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pp}_e, x, m_b)$ | return $m$ |
| $b' \xleftarrow{\$} \mathcal{A}(c^*)$ | return $\perp$ |
| return $b' = b$ | |

Figure 7: Game $\mathsf{G}_0$

**Game 1.** Game $\mathsf{G}_1$ (Figure 8) proceeds exactly like $\mathsf{G}_0$, except that decryption circuit returns $\perp$ if target tag $x$ is equal to tag $t$ of the ciphertext.

| $\underline{\mathsf{G}_1(1^\lambda, x)}$ | $\underline{C_{\mathsf{sk},x}(c, w)}$ |
|---|---|
| $(\mathsf{pp}_e, \mathsf{pp}_d) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ | Parse $c$ as $(c_{pe}, t)$ |
| $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$ | if $R(t, w) = 1$ |
| $b \in_R \{0, 1\}$ | $\boxed{\text{if } x = t}$ |
| $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pp}_e, x, m_b)$ | $\boxed{\text{return } \perp}$ |
| $b' \xleftarrow{\$} \mathcal{A}(c^*)$ | $m \leftarrow \mathsf{PE.Decrypt}(\mathsf{sk}, c_{pe}, t)$ |
| return $b' = b$ | return $m$ |
| | return $\perp$ |

Figure 8: Game $\mathsf{G}_1$

*Lemma* 4. *For every PPT adversary $\mathcal{A}$, there exist an extractor $\mathcal{E}$ and negligible function $\alpha$ such that the following holds. If*

$$\varepsilon = \left|\Pr[\mathsf{G}_0(1^\lambda, x) = 1] - \Pr[\mathsf{G}_1(1^\lambda, x) = 1]\right| > \alpha(\lambda),$$

*then $\mathcal{E}(1^\lambda, x)$ outputs a witness $w$ such that $R(x, w) = 1$ with non-negligible probability in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$, where $\mathfrak{p}$ is a polynomial.*

Proof. Assume there is an attacker $\mathcal{A}$ and polynomial $\alpha(\lambda)$ for which holds

$$\varepsilon = \left|\Pr[\mathsf{G}_0(1^\lambda, x) = 1] - \Pr[\mathsf{G}_1(1^\lambda, x) = 1]\right| > \alpha(\lambda).$$

The games differ only in the circuit that is obfuscated. Thus, we can use these to construct a distinguisher $\mathcal{D}$ to distinguish the obfuscations of these circuits. First, we show how to construct an efficient sampler for our circuits.

Circuit sampler $S(1^\lambda, x)$:

1. Generate $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{PE.KeyGen}(1^\lambda)$.
2. Construct $C_{\mathsf{sk}}, C_{\mathsf{sk},x}$.
3. Set $aux := (\mathsf{pk}, x)$.

4. Return $(C_{\mathsf{sk}}, C_{\mathsf{sk},x}, aux)$.

The distinguisher $\mathcal{D}(1^\lambda, \widetilde{C}, C_{\mathsf{sk}}, C_{\mathsf{sk},x}, aux)$ works as follows:

1. Parse $aux$ as $(\mathsf{pk}, x)$.
2. Run $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pk}, \widetilde{C}, x)$.
3. Randomly pick $b \in_R \{0,1\}$ and produce $c^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{pk}, x, m_b)$.
4. Run $b' \xleftarrow{\$} \mathcal{A}(c^*)$.
5. Return $b' = b$.

In the instance where $\widetilde{C}$ is an obfuscation of $C_{\mathsf{sk}}$, $\mathcal{D}$ perfectly simulates $\mathsf{G}_0$, whereas when $\widetilde{C}$ is an obfuscation of $C_{\mathsf{sk},x}$, it perfectly simulates $\mathsf{G}_1$. Hence, $\mathcal{D}$ can distinguish the obfuscated circuits with probability at least $\varepsilon > \alpha(\lambda)$. The security of $eO$ guarantees that there is an extractor $\mathcal{E}'$ which extracts an input on which given circuits differ with non-negligible probability in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$ plus the time required to simulate the security experiment. We can use $\mathcal{E}'$ to construct $\mathcal{E}$ from Lemma 4.

Extractor $\mathcal{E}(1^\lambda, x)$

1. $(C_{\mathsf{sk}}, C_{\mathsf{sk},x}, aux) \leftarrow S(1^\lambda, x)$
2. $(c, w) \xleftarrow{\$} \mathcal{E}'(1^\lambda, C_{\mathsf{sk}}, C_{\mathsf{sk},x}, aux)$
3. return $w$

Notice that the decryption circuits produce different outputs only in the case where tag $t$ is equal to our challenge $x$ and $R(t, w) = 1$. Hence, the extractor $\mathcal{E}'$ returns $(c, w) = ((c_{pe}, x), w)$ such that $R(x, w) = 1$ with non-negligible probability, which means we obtain a valid witness for our instance $x$ in time $\mathfrak{p}(\lambda, 1/(\varepsilon - \alpha(\lambda)))$. □

We will now define an alternative setup algorithm $\mathsf{Setup}'$. This algorithm differs from $\mathsf{Setup}$ in that we additionally puncture the key $\mathsf{sk}$ on the challenge tag $x$.

$\underline{\mathsf{Setup}'(1^\lambda, x)}$
$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathtt{PE}.\mathsf{KeyGen}(1^\lambda)$
$\mathsf{sk}^* \xleftarrow{\$} \mathtt{PE}.\mathsf{Puncture}(\mathsf{sk}, x)$
$\widetilde{C}_{\mathsf{sk}^*,x} \xleftarrow{\$} e\mathcal{O}(1^\lambda, C_{\mathsf{sk}^*,x})$
$\mathsf{pp}_e := \mathsf{pk}$, $\mathsf{pp}_d := \widetilde{C}_{(\mathsf{sk}^*,x)}$
return $(\mathsf{pp}_e, \mathsf{pp}_d)$

Figure 9: Alternative setup

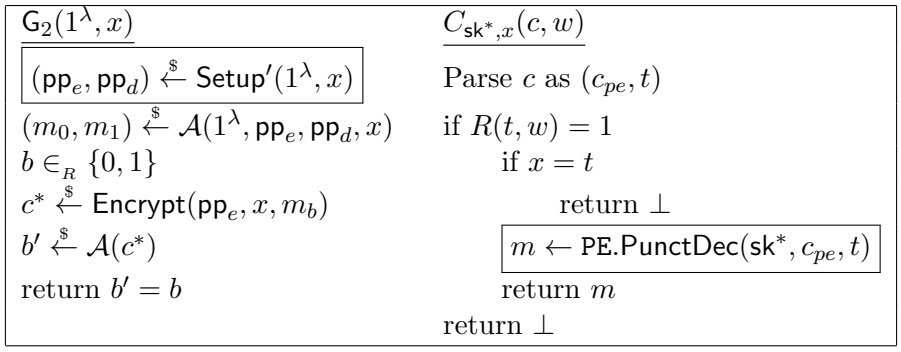


Figure 10: Game $\mathsf{G}_2$

**Game 2.** In $\mathsf{G}_2$ (Figure 10) we now run our alternative setup algorithm $\mathsf{Setup}'$, which punctures the key $\mathsf{sk}$ on the tag $x$. The decryption circuit now uses the punctured key $\mathsf{sk}^*$.

*Lemma* 5. $|\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]| \leq \alpha_{\mathit{eO}}(1^\lambda)$, *where* $\alpha_{\mathit{eO}}(1^\lambda)$ *is from Definition 5.*

Proof. For purpose of contradiction assume that there is an attacker such that we have that $|\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]| > \alpha_{\mathit{eO}}(1^\lambda)$. As in the previous lemma, this would then imply the existence of distinguisher for our circuits and hence the existence of an efficient extractor. By correctness of $\mathtt{PE}$ the circuits in both games have the exact same input/output behaviour and there is no input where the circuits differ, therefore there can not exist such an extractor, which yields a contradiction. Hence we must have $|\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_2 = 1]| \leq \alpha_{\mathit{eO}}(1^\lambda)$, as required. □

**Game 3.** Finally, in $\mathsf{G}_3$ (Figure 11) we encrypt a random message $m$ and a random circuit $F$ of correct length instead of $(m_b, F_b)$.

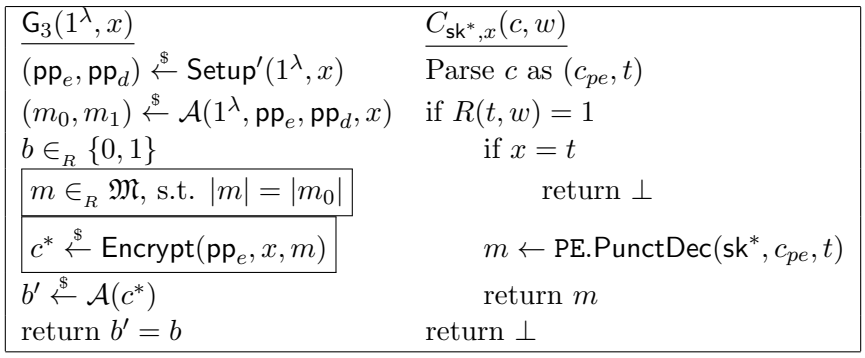


Figure 11: Game $\mathsf{G}_3$

*Lemma* 6. $|\Pr[\mathsf{G}_2 = 1] - \Pr[\mathsf{G}_3 = 1]| = \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}}$.

Proof. Assume that there is an adversary $\mathcal{A}$ which can distinguish games $\mathsf{G}_2$ and $\mathsf{G}_3$. We construct an adversary $\mathcal{B}$ which breaks security of the puncturable encryption scheme.

The adversary $\mathcal{B}(1^\lambda, x)$:

1. Send $x$ to the challenger.
2. Receive public key $\mathsf{pk}$ and punctured secret key $\mathsf{sk}^*$ from the challenger.
3. Construct obfuscation of the circuit $C_{\mathsf{sk}^*,x}$ and run the adversary $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}_e, \mathsf{pp}_d, x)$ where $\mathsf{pp}_e := \mathsf{pk}$, $\mathsf{pp}_d := \widetilde{C}_{\mathsf{sk}^*,x}$.
4. Randomly pick $\hat{b} \in_R \{0,1\}$, sends $m_{\hat{b}}$ to challenger and obtains ciphertext $c^*$ as a response.
5. Run $b' \xleftarrow{\$} \mathcal{A}(c^*)$.
6. Return $b' = \hat{b}$.

It is not hard to see that if challenger in $\mathsf{ExpPE}^{\mathsf{b}}_{\mathcal{A}}(1^\lambda)$ picks $b = 0$, then $c^*$ is an encryption of $m_{\hat{b}}$ and $\mathcal{B}$ perfectly simulates game $\mathsf{G}_2$, otherwise it perfectly simulates $\mathsf{G}_3$. Hence, we have

$$\mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} = \left|\Pr[\mathsf{G}_2 = 1] - \Pr[\mathsf{G}_3 = 1]\right|.$$

□

*Lemma* 7. $\Pr[\mathsf{G}_3 = 1] = 1/2$.

Proof. Now that we encrypt a random message and function pair, the challenge ciphertext $c^*$ is independent of our choice of $b$. Thus, the adversary gets no information about $b$ and can do no better than guessing and has a success probability of exactly $\frac{1}{2}$. □

Combining Lemmas 1-4 we obtain following:

$$\begin{aligned}\Pr[\mathsf{ExpEOWE}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}_{\mathcal{A}}(1^\lambda, x) = 1] &= \Pr[\mathsf{G}_0 = 1] \\ &\leq \sum_{i=0}^{2} \left|\Pr[\mathsf{G}_i = 1] - \Pr[\mathsf{G}_{i+1} = 1]\right| + \Pr[\mathsf{G}_3 = 1] \\ &= \frac{1}{2} + \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} + \alpha_{e\mathcal{O}}(\lambda) \\ &\quad + \left|\Pr[\mathsf{G}_0(1^\lambda, x) = 1] - \Pr[\mathsf{G}_1(1^\lambda, x) = 1]\right|\end{aligned}$$

Assuming that

$$\Pr[\mathsf{ExpEOWE}^{\mathtt{OWE.Encrypt},\mathtt{OWE.Decrypt}}_{\mathcal{A}}(1^\lambda, x) = 1] = \frac{1}{2} + \varepsilon,$$

we obtain:

$$\frac{1}{2} + \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} + \alpha_{e\mathcal{O}}(\lambda) + \left|\Pr[\mathsf{G}_0(1^\lambda, x) = 1] - \Pr[\mathsf{G}_1(1^\lambda, x) = 1]\right| \geq \frac{1}{2} + \varepsilon$$

$$\iff$$

$$\left|\Pr[\mathsf{G}_0(1^\lambda, x) = 1] - \Pr[\mathsf{G}_1(1^\lambda, x) = 1]\right| \geq \varepsilon - \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} - \alpha_{e\mathcal{O}}(\lambda)$$

By Lemma 4, this implies the existence of an extractor $\mathcal{E}$ and negligible function $\alpha_{e\mathcal{O}}$ which outputs a witness $w$ such that $R(x, w) = 1$ with non-negligible probability in time $\mathfrak{p}(1/(\varepsilon - \mathbf{Adv}^{\mathtt{PE}}_{\mathcal{B}} - 2\alpha_{e\mathcal{O}}(\lambda))$, which concludes our proof.

□

# 5 Realising Our Scheme

As our main result is a general framework, the efficiency of the final scheme is directly tied to the efficiency of the underlying components. As obfuscation is a relatively new primitive, there have not been as much progress in the efficiency of the constructions, as compared to that of encryption schemes. Obfuscation is quite a strong assumption, and one we are not sure exists in the general case. However we can realise some functionalities, including what we require, by relying on Trusted Execution Environments (TEEs). TEEs come in various flavours, such as Intel's Software Guard eXtensions (SGX), AMD's Secure Encrypted Virtualisation (SEV) and ARM's TrustZone, among others. Indeed, similar functionalities to Offline Extractable Functional Witness Encryption have already been efficiently realised in SGX [FVBG17]. This leaves our ciphertext size as the primary measure of efficiency.

Of course, this is highly dependant on our puncturable encryption scheme. We could simply take an extant puncturable scheme [GM15, GHJL17, DGJ$^+$18] and plug it into our framework, but that would be quite inefficient. The reason for this is that all known schemes aim for much stronger security than what we require, which naturally leads to more complex constructions. Intuitively, any selectively secure tag-based encryption should yield a selectively secure one-time puncturable encryption scheme. This follows from the fact that the security reduction must simulate decryption queries for all tags but the target tag, but without being able to use the standard decryption algorithm, which fits almost exactly to our definition. It must be noted that this does not work immediately with any tag based scheme. In particular, we do not see how to build such a scheme from the Twin Diffie-Hellman tag-based scheme of Haralambiev et al. [HJKS10].

Along with this counter-example, we do also have a positive result, that is we show how to achieve this defintion with one specific scheme. Specifically, the scheme we consider is the tag based encryption scheme due to Kiltz [Kil06]. It must be noted that our proof is an adaptation of Kiltz' original proof, we simply show that it is a one-time puncturable encryption scheme. Note that our proof is an adaptation of the original proof of Kiltz and our contribution is to show that it satisfies our definition.

We see that if we use the Kiltz' TBE as a building block, we get ciphertexts that consist of 5 group elements. In comparison, Abusalah, Fuchsbauer and Pietrzak [AFP16] have ciphertexts that consists of at least 32 group elements. These figures assume a "small" message that can be efficiently encoded into a single group element. Larger messages must be split into blocks and each block encrypted separately. In our case, this means that we require only 5 elements extra per block. In constrast, [AFP16] requires an additional 8 elements of $G_1$ per block. For a message consisting of $N$ blocks, we have a ciphertext of $5N$ group elements. On the other hand [AFP16] has a ciphertext of size $24 + 8N$ group elements. In the case where we can switch to the KEM/DEM hybrid encryption paradigm, we get a total KEM size of 5 group elements. Comparatively, Abusalah, Fuchsbauer and Pietrzak [AFP16] have a ciphertext of size 32 group elements.

## 5.1 Kiltz' Tag Based Encryption Scheme

The TBE scheme due to Kiltz [Kil06] is based on the Gap Decision Linear assumption ($\mathsf{GapDLin}$), which is the the Decision Linear Assumption with an additional Decisional Diffie Hellman Oracle ($\mathsf{DDHVf}$), which can be realised with a bilinear map [Kil06]. We will work relative to a group generation algorithm $\mathsf{GroupGen}(1^\lambda)$. We now recall a variant of the the Decisional Linear assumption [BBS04], specifically when it is in a so-called *gap group* [OP01], which we realise with a bilinear

pairing.

*Definition* 8 (Gap Decision Linear Assumption.). The gap decision linear assumption, denoted by $\mathsf{GapDLin}$ states an adversary which has access to a Diffie-Hellman oracle $\mathsf{DDHVf}$, given $\mathsf{gk} = (\mathbb{G}, g, p)$, two additional random generators $h, k$ of $\mathbb{G}$ and a tuple $(g^u, h^v, k^w)$ where $u, v, r \in_R \mathbb{Z}_p$, $c = u + v + \beta r$, with $\beta \in_R \{0, 1\}$, it is hard to decide if $\beta = 0$ or $\beta = 1$. $\mathsf{GapDLin}$ is said to be $(t, \varepsilon)$-hard if for all adversaries $\mathcal{A}$ running in time at most $t$, we have

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{GapDLin}} = \left| \Pr \left[ \begin{array}{c} \beta' = \beta : \\ (x, y, c, d, r \in_R \mathbb{Z}_p, \beta \in_R \{0,1\}; \\ \beta' \xleftarrow{\$} \mathcal{A}^{\mathsf{DDHVf}(\cdot)}(g^a, g^b, g^{ac}, g^{cd}, g^{c+d+\beta r}) \end{array} \right] - \frac{1}{2} \right| \leq \varepsilon.$$

A natural question to ask is how one could possibly (efficiently) provide such an oracle to and adversary without having to solve a hard problem. The solution to this are the so called Gap Diffie Hellman groups provided by bilinear pairings. In these groups the Decisional Diffie-Hellman problem is easy, but the Computational Diffie-Hellman problem is still hard. In this case, the $\mathsf{DDHVf}$ functionality is computable by all and is no longer an oracle.

$\underline{\mathsf{KeyGen}(1^\lambda)}$
$g_1 \in_R \mathbb{G}, x_q, x_2, y_1, y_2 \in_R \mathbb{Z}_p$
Pick $g_2 \in \mathbb{G}$ s.t. $g_1^{x_1} = g_2^{x_2} = z$
$u_1 = g_1^{y_1}, u_2 = g_2^{y_2}$
$\mathsf{pk} = (g_1, g_2, u_1, u_2, z), \mathsf{sk} = (x_1, x_2, y_1, y_2)$
return $(\mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Encrypt}(\mathsf{pk}, m, t)}$
$r_1, r_2 \in_R \mathbb{Z}_p$
$C_1 = g_1^{r_1}, C_2 = g_2^{r_2}$
$D_1 = z^{t \cdot r_1} u_1, D_2 = z^{t \cdot r_2} u_2$
$K = z^{r_1 + r_2}$
$\psi = K \cdot m$
return $c = (C_1, C_2, D_1, D_2, \psi)$

$\underline{\mathsf{Decrypt}(\mathsf{sk}, c, t)}$
Parse $c = (C_1, C_2, D_1, D_2, \psi)$
if $(C_1^{t \cdot x_1 + y_1} \neq D_1) \vee (C_2^{t \cdot x_2 + y_2} \neq D_2)$
    return $\perp$
else
    $K = C_1^{x_1} \cdot C_2^{x_2}$
    return $m = \psi \cdot K^{-1}$

$\underline{\mathsf{Puncture}(\mathsf{sk}, t^*)}$
Parse $\mathsf{sk} = (x_1, x_2, y_1, y_2)$
$c_1 = y_1 + t^* \cdot x_1$
$c_2 = y_2 + t^* \cdot x_2$
return $\mathsf{sk}_{t^*} = (c_1, c_2)$

$\underline{\mathsf{PunctDec}(\mathsf{sk}_{t^*}, c, t)}$
Parse $c = (C_1, C_2, D_1, D_2, \psi)$
if $\mathsf{DDHVf}(g_1, z^t \cdot u_1, C_1, D_1) = 0$
$\vee \mathsf{DDHVf}(g_2, z^t \cdot u_2, C_2, D_2) = 0$
    return $\perp$
$\Delta = D_1 \cdot D_2$
$\Gamma = C_1^{c_1} \cdot C_2^{c_2}$
$K = (\Delta / \Gamma)^{1/(t - t^*)}$
return $m = \psi \cdot K^{-1}$

Figure 12: The Adapted Tag-Based Encryption Scheme of Kiltz `KiltzTBE` [Kil06]

*Theorem* 3. *Assume the* $\mathsf{GapDLin}$ *assumption is* $(t', \varepsilon')$*-hard. Then for any* $(q_h, q_s)$*, the* `KiltzTBE` *scheme is a* $(t, \varepsilon)$*-selective one-time puncturable secure encryption scheme, where*

$$\begin{aligned} \varepsilon' &= \varepsilon \\ t' &\approx t \end{aligned}$$

PROOF. To prove our theorem, we need to show how we generate our public key, our punctured key and how we embed our GapDLin challenge in the challenge ciphertext. We begin with our key simulation. Our gap oracle DDHVf is our bilinear pairing. After receiving the challenge $(g, h, k, g^u, h^v, k^w)$, the reduction will initialize $\mathcal{A}$ and get the challenge $t^*$ and it is now ready to program its keys. We begin by picking $\mathsf{sk}_{t^*} = (c_1, c_2) \in_R \mathbb{Z}_p^2$, which is distributed exactly as the normal punctured key. Now reduction sets $g_1 = g, g_2 = h, z = k$ and computes $u_1 = z^{-t^*} \cdot g_1^{c_1}, u_2 = z^{-t^*} \cdot g_2^{c_2}$ and sets $\mathsf{pk} = (g_1, g_2, u_1, u_2, z)$. The reduction will now pass $\mathsf{pk}, \mathsf{sk}_{t^*}$ to the adversary. After this, the adversary will return the challenge message. The reduction now computes the challenge ciphertext as $c^* = (g^u, h^v, (g^u)^{c_1}, (h^v)^{c_2}, m^* \cdot k^w)$ and sends this to the adversary. Finally the adversary will output their guess $b'$. Notice now that if $w = u + v$ then $c^*$ is indeed a well formed encryption of $m^*$ under $\mathsf{pk}$ and corresponds to the case $b = 0$. On the other hand if $w \in_R \mathbb{Z}_p$, then $c^*$ is an encryption of a random message and hence corresponds to the case $b = 1$. Thus, if we forward $b'$ as our guess, we have exactly the advantage of the adversary, giving us $\varepsilon' = \varepsilon$. The time bound comes from the time needed to compute the keys. This completes the proof. □

# 6 Conclusions

We have shown a general framework for constructing Offline Functional Witness Encryption, which is not only more efficient that, but also provides better security guarantees than the previous construction of Abusalah, Fuchsbauer and Pietrzak [AFP16]. Our framework can be easily adapted to additionally provide extractability, by replacing the iO with an eO. If we apply a similar transformation to the scheme of Abusalah, Fuchsbauer and Pietrzak [AFP16], the resultant scheme is still only selectively secure. In all cases, we have constant size ciphertext, 5 group elements, without the need for NIZK proofs, whereas Abusalah, Fuchsbauer and Pietrzak [AFP16] have a larger constant size ciphertext, of around 32 group elements, which includes NIZK proofs. This is due to the fact that we only need one ciphertext from our one-time puncturable encryption scheme, whereas they need two ciphertext and a NIZK proof.

# Acknowledgements

We would like to thank the anonymous reviewers for ACNS 2020. We would also like to thank Hamza Abusalah, for pointing us to additional relevant literature. The authors were funded by the German Federal Ministry of Education and Research (BMBF) project REZEIVER. Part of this work was completed while the authors were employed at Paderborn University.

# References

[AFH+15] Martin R. Albrecht, Pooya Farshim, Dennis Hofheinz, Enrique Larraia, and Kenneth G. Paterson. Multilinear maps from obfuscation. Cryptology ePrint Archive, Report 2015/780, 2015. http://eprint.iacr.org/2015/780. Cited in Sec. 1.2

[AFH+16] Martin R. Albrecht, Pooya Farshim, Dennis Hofheinz, Enrique Larraia, and Kenneth G. Paterson. Multilinear maps from obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference,*

*Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 446–473, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-49096-9_19. Cited in Sec. 1.2

[AFP16] Hamza Abusalah, Georg Fuchsbauer, and Krzysztof Pietrzak. Offline witness encryption. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16: 14th International Conference on Applied Cryptography and Network Security*, volume 9696 of *Lecture Notes in Computer Science*, pages 285–303, Guildford, UK, June 19–22, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-39555-5_16. Cited in Sec. 1, 1.1, 1.2, 1.4, 1.5, 2.2, 5, 6

[Agr19] Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 191–225, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-17653-2_7. Cited in Sec. 1.2

[AJN+16] Prabhanjan Ananth, Aayush Jain, Moni Naor, Amit Sahai, and Eylon Yogev. Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 491–520, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53008-5_17. Cited in Sec. 1

[And97] Ross Anderson. Two remarks on public key cryptography. Technical Report 549, University of Cambrige Computer Laboratry, 1997. Available at https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-549.pdf. Cited in Sec. 1.2

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-28628-8_3. Cited in Sec. 5.1

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 52–73, San Diego, CA, USA, February 24–26, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-54242-8_3. Cited in Sec. 1, 1.2, 1.5, 2.3, 2.3

[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, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. doi:10.1007/3-540-44647-8_1. Cited in Sec. 1.2, 2.3

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012. URL: https://doi.org/10.1145/2160158.2160159, doi:10.1145/2160158.2160159. Cited in Sec. 1.2, 2.3

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-54631-0_29. Cited in Sec. 1, 1

[BGI+17] Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-70700-6_10. Cited in Sec. 1

[BH15] Mihir Bellare and Viet Tung Hoang. Adaptive witness encryption and asymmetric password-based cryptography. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 308–331, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-46447-2_14. Cited in Sec. 1, 1

[BL18] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 500–532, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-78375-8_17. Cited in Sec. 1

[BMSZ16] Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 764–791, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-49896-5_27. Cited in Sec. 1

[BP15] Elette Boyle and Rafael Pass. Limits of extractability assumptions with distributional auxiliary input. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 236–261, Auckland, New Zealand, November 30 – December 3, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-48800-3_10. Cited in Sec. 2.3

[BS02] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. Cryptology ePrint Archive, Report 2002/080, 2002. http://eprint.iacr.org/2002/080. Cited in Sec. 1.2

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. In *Topics in algebraic and noncommutative geometry (Luminy/Annapolis, MD, 2001)*, volume 324 of *Contemp. Math.*, pages 71–90. Amer. Math. Soc., Providence, RI, 2003. URL: http://dx.doi.org/10.1090/conm/324/05731, doi:10.1090/conm/324/05731. Cited in Sec. 1.2

[CDG+17] Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-63715-0_2. Cited in Sec. 1

[DGJ+18] David Derler, Kai Gellert, Tibor Jager, Daniel Slamanig, and Christoph Striecks. Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange. Cryptology ePrint Archive, Report 2018/199, 2018. https://eprint.iacr.org/2018/199. Cited in Sec. 1.2, 5

[DJSS18] David Derler, Tibor Jager, Daniel Slamanig, and Christoph Striecks. Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 425–455, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-78372-7_14. Cited in Sec. 1.2

[FHHL18] Pooya Farshim, Julia Hesse, Dennis Hofheinz, and Enrique Larraia. Graded encoding schemes from obfuscation. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 371–400, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-76581-5_13. Cited in Sec. 1.2

[FNV17] Antonio Faonio, Jesper Buus Nielsen, and Daniele Venturi. Predictable arguments of knowledge. In Serge Fehr, editor, *PKC 2017: 20th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10174 of *Lecture Notes in Computer Science*, pages 121–150, Amsterdam, The Netherlands, March 28–31, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-54365-8_6. Cited in Sec. 1

[FVBG17] Ben Fisch, Dhinakaran Vinayagamurthy, Dan Boneh, and Sergey Gorbunov. IRON: Functional encryption using intel SGX. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 765–782, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press. doi:10.1145/3133956.3134106. Cited in Sec. 5

[GGH12] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. Cryptology ePrint Archive, Report 2012/610, 2012. http://eprint.iacr.org/2012/610. Cited in Sec. 1.2

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-38348-9_1. Cited in Sec. 1.2

[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 Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press. doi:10.1109/FOCS.2013.13. Cited in Sec. 1.2

[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, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 518–535, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-44371-2_29. Cited in Sec. 2.3

[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 Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press. doi:10.1145/2488608.2488667. Cited in Sec. 1

[GHJL17] Felix Günther, Britta Hale, Tibor Jager, and Sebastian Lauer. 0-RTT key exchange with full forward secrecy. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 519–548, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-56617-7_18. Cited in Sec. 1.2, 5

[GK18] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing for general access structures. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 501–530, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96884-1_17. Cited in Sec. 1

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 536–553, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-40084-1_30. Cited in Sec. 1, 1, 2.2, 4

[GLS15] S. Dov Gordon, Feng-Hao Liu, and Elaine Shi. Constant-round MPC with fairness and guarantee of output delivery. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 63–82, Santa Barbara, CA, USA, August 16–20,

2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-48000-7_4. Cited in Sec. 1

[GM15] Matthew D. Green and Ian Miers. Forward secure asynchronous messaging from puncturable encryption. In *2015 IEEE Symposium on Security and Privacy*, pages 305–320, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press. doi:10.1109/SP.2015.26. Cited in Sec. 1.2, 2.4, 5

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432, Istanbul, Turkey, April 13–17, 2008. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-78967-3_24. Cited in Sec. 1.1

[Had00] Satoshi Hada. Zero-knowledge and code obfuscation. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 443–457, Kyoto, Japan, December 3–7, 2000. Springer, Heidelberg, Germany. doi:10.1007/3-540-44448-3_34. Cited in Sec. 1.2

[HJKS10] Kristiyan Haralambiev, Tibor Jager, Eike Kiltz, and Victor Shoup. Simple and efficient public-key encryption from computational Diffie-Hellman in the standard model. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 1–18, Paris, France, May 26–28, 2010. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-13013-7_1. Cited in Sec. 5

[IPS15] Yuval Ishai, Omkant Pandey, and Amit Sahai. Public-coin differing-inputs obfuscation and its applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 668–697, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-46497-7_26. Cited in Sec. 2.3

[Kil06] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany. doi:10.1007/11681878_30. Cited in Sec. 1.2, 5, 5.1, 12

[KNY14] Ilan Komargodski, Moni Naor, and Eylon Yogev. Secret-sharing for NP. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 254–273, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-45608-8_14. Cited in Sec. 1

[KSY18] Ilan Komargodski, Gil Segev, and Eylon Yogev. Functional encryption for randomized functionalities in the private-key setting from minimal assumptions. *Journal of Cryptology*, 31(1):60–100, January 2018. doi:10.1007/s00145-016-9250-8. Cited in Sec. 1

[LJKW18] Jia Liu, Tibor Jager, Saqib A. Kakvi, and Bogdan Warinschi. How to build time-lock encryption. *Des. Codes Cryptography*, 86(11):2549–2586, 2018. URL: `https://doi.org/10.1007/s10623-018-0461-x`, `doi:10.1007/s10623-018-0461-x`. Cited in Sec. 1, 1, 1.3

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, Baltimore, MD, USA, May 14–16, 1990. ACM Press. `doi:10.1145/100216.100273`. Cited in Sec. 1.1

[OP01] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118, Cheju Island, South Korea, February 13–15, 2001. Springer, Heidelberg, Germany. `doi:10.1007/3-540-44586-2_8`. Cited in Sec. 5.1

[PS15] Omer Paneth and Amit Sahai. On the equivalence of obfuscation and multilinear maps. Cryptology ePrint Archive, Report 2015/791, 2015. `http://eprint.iacr.org/2015/791`. Cited in Sec. 1.2

[Zha16] Mark Zhandry. How to avoid obfuscation using witness PRFs. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 421–448, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany. `doi:10.1007/978-3-662-49099-0_16`. Cited in Sec. 1.5