# Witness Authenticating NIZKs and Applications<sup>\*</sup>

Hanwen Feng and Qiang Tang

School of Computer Science The University of Sydney {hanwen.feng, qiang.tang}@sydney.edu.au

Abstract. We initiate the study of witness authenticating NIZK proof systems (waNIZKs), in which one can use a witness w of a statement x to identify whether a valid proof for x is indeed generated using w. Such a new identification functionality enables more diverse applications, and it also puts new requirements on soundness that: (1) no adversary can generate a valid proof that will not be identified by any witness; (2) or forge a proof using some valid witness to frame others. To work around the obvious obstacle towards conventional zero-knowledgeness, we define entropic zero-knowledgeness that requires the proof to leak no partial information, if the witness has sufficient computational entropy. We give a formal treatment of this new primitive. The modeling turns out to be quite involved and multiple subtle points arise and particular cares are required. We present general constructions from standard assumptions. We also demonstrate three applications in non-malleable (perfectly one-way) hash, group signatures with verifier-local revocations and plaintext-checkable public-key encryption. Our waNIZK provides a new tool to advance the state of the art in all these applications.

# 1 Introduction

Non-interactive zero-knowledge (NIZK) proof systems [8,27] allow one to prove a statement by sending a single message to a verifier without revealing anything beyond the validity of the statement. NIZKs have been a ubiquitous tool in modern cryptography and play an essential role in constructing many important primitives such as chosen-ciphertext secure encryptions [38,41], anonymous authentication tools such as group and ring signatures [21,20], and many more.

While privacy is essential, some interesting functionalities become unattainable when considering the strong privacy definition where *all* partial information is protected. For example, doing a binary search for a plaintext in ciphertext is elusive when using a semantically secure encryption. How to construct secure schemes enabling certain functionalities, while maintaining the best possible privacy, is one of the central questions in modern cryptography and has been studied in a large amount of works in different contexts, e.g., [5,17,35,11,14].

In this paper, we turn our attention to NIZK proofs and consider to add an "identification" functionality: a witness w of a statement x (which potentially has many valid witnesses) in an NP language L can be used check whether a valid proof  $\pi$  showing  $x \in L$  was generated by the witness w, i.e., Identify $(x, w, \pi) \stackrel{?}{=} 1$ . It means that each witness w is "committed" to the proof  $\pi$  generated using w. Other than that, the proof will remain "zero-knowledge". Such an exclusive checking capability immediately enables many interesting applications. For instance, one could easily realize a private/covert communication channel between administrators of an anonymous token system [34] as follows: administrators may consider using shared two witnesses  $w_1, w_2$  to indicate whether a valid "anonymous certificate" falls into a certain blacklist (or whitelist) by using  $w_1$ ; in this way,

<sup>\*</sup> A preliminary version of this paper appeared in *Advances in Cryptology*- - CRYPTO 2021, Part IV, pp. 3–33. This full version includes proofs for all theorems and lemmas.

only the administrators obtain this extra information which remains hidden to everyone else in the system. As pointed out in the recent work of [34], such a tool is important to enable CDN providers to distinguish potentially malicious requests without breaching anonymity.

Adding this simple identification functionality also naturally posts new requirements on soundness: (1) if an attacker who knows a set of witnesses of a statement x generates a proof  $\pi$  for x, this proof must be identified by one of these witnesses; and (2) if a witness w is not known to an attacker (who may have other witnesses), none of the proofs generated by the attacker will be identified by w, i.e., Identify $(x, w, \pi) = 0$ .

We put forth a new notion called witness-authenticating NIZKs to capture all those requirements. Essentially, we add a way of distinguishing between different witnesses in NIZKs. As we will demonstrate soon in the applications, our new notion provides a new tool to advance the state of the art in multiple different domains: non-malleable (perfectly one-way) hash, group signature with verifier-local revocation, and plaintext-checkable public-key encryption.

#### 1.1 Our contributions

We overview our contributions in more detail below.

**Definitional contributions.** Adding a single identification functionality and defining the witnessauthenticating NIZK proof system turn out to be highly involved; we have to revisit essentially every single property of the conventional NIZK proof system, and multiple subtleties exist.

Syntax and identifier witness. The basic idea is to augment a non-interactive proof system with an  $\mathsf{Identify}(\cdot)$  algorithm to check whether a witness he is possessing was used to generate the proof. However, often in practice, only a part of the witness (such as a secret key) is bound to a user; while other parts, such as random coins, may not be always available. To avoid unnecessary restrictions on the applications, we introduce a generalization that we only require the **Identify** algorithm to take into a part of the witness. A bit more formally, we split each witness w into  $w^I$  and  $w^{NI}$ , we call the former  $w^I$  identifier witness, and the remaining part  $w^{NI}$  non-identifier witness. Using an identifier witness  $w^I$  of x, one can check whether a proof for x was generated using a witness in the form of  $(w^I, \star)$ . If  $\mathsf{Identify}(x, \pi, w^I) = 1$ , we say  $\pi$  is authenticated by  $w^I$ . When privacy is not considered in the context, we call such a proof system a witness-authenticating non-interactive proof system (waNIPS for short).

<u>Entropic zero-knowledgeness</u>. As a witness-authenticating proof has to convey at least one bit about the identifier witness to make the identification functionality possible, the conventional zeroknowledgeness that hides all partial information of witness becomes out of reach. Therefore, we study the best possible privacy definition that we call the entropic zero-knowledgeness (entropic ZK), and call a waNIPS with this property a waNIZK.

- Defining unpredictable sampler. Similar to that semantic security is impossible for deterministic encryption, if an identifier witness can be guessed easily by the adversary, the **Identify** algorithm enables the adversary to trivially distinguish a real proof from a simulated proof. It follows that the privacy definition should be defined for languages with "unpredictable" (identifier) witnesses. To model that, we introduce an unpredictable sampler G (provided by the adversary) which ensures that for a random sample  $(x, w^I, w^{NI}) \leftarrow G(1^{\lambda})$ , given x, finding the associated identifier witness  $w^I$  is hard.

Several subtle issues appear. (1) In applications, if the whole statement is generated by the sampler, it may cause a trivial impossibility; for example, if a waNIZK is deployed in a larger

system, which requires an honestly generated public parameter pp, the witness might be leaked completely if pp is malicious. We handle it by introducing a parameter generation algorithm that is not under the control of the adversary or sampler G. (2) In an adaptive setting, the sampler G could be generated by the attacker after seeing the CRS. But now, the sampled statement could simply contain one proof for which the corresponding witness is never output. This will enable a malicious prover to generate a proof without using any witness, which clearly violates the knowledge soundness. We get around this by requiring the unpredictability of the identifier witness to hold for *every* CRS value (instead of a randomly chosen one). Please see Sect. 3.1 for details.

- Defining entropic ZK. We define the entropic ZK, somewhat analogous to entropic security in encryptions [5], by capturing that adversaries still cannot learn anything more about  $w^{I}$ from  $\pi$  if  $w^{I}$  is sampled from the unpredictable sampler G (specified by the adversary). In conventional ZK, the whole witness is provided by the adversary; now adversary provides only a sampler. Directly integrating the unpredictable sampler to the conventional adaptive zeroknowledge definition would restrict adversary from learning side information about the witness via other or directly related proofs. We let the adversary to obtain proofs on related statements by querying a proof oracle. Please see Sect.3.2 for details.

<u>Soundness definitions</u>. As very briefly mentioned above, soundness definitions also require a major upgrade because of the new identification functionality. Besides the conventional (knowledge) soundness, we require two new properties to show that the identifier witness to be "committed" to the proof: (1) a proof must be identifiable by one of the identifier witnesses used in the proof generation; (2) a malicious prover cannot "forge" a proof that will be identified by some identifier witness she does not know. Concretely,

- To formulate the former property, we augment the knowledge soundness (named authenticating knowledge soundness), saying that a witness extracted by a knowledge extractor from a valid proof not only validates the statement being proven, but also authenticates the proof.
- The latter property, which we call *unforgeability*, also relies on the unpredictable sampler; it is analogous to "unforgeability" in MAC. Namely, for a target witness generated from the unpredictable sampler, the adversary who can obtain multiple proofs generated from it still cannot produce a new proof that will be authenticated by this identifier witness.

Note that unforgeability prevents a malicious prover from "framing" a witness. In some applications, a malicious prover may generate a proof that links to a string which is not even a witness. We thus also introduce a notion called *identifier uniqueness*, which ensures that it is infeasible to generate a valid proof that could be authenticated by two different strings.

We remark that unforgeability and identifier uniqueness are *incomparable*: an attacker that cannot forge a proof being authenticated by an unknown witness may be able to produce one being authenticated by two witnesses he possesses; on the other hand, for technical reasons in the definitions, the identifier uniqueness is not strictly stronger either. But each could be useful in various applications when working together with other properties from the context.

There are several versions of weakening, e.g., in the CRS-independent setting; and strengthening. We refer detailed discussions in Sec.3.3.<sup>1</sup>

<sup>&</sup>lt;sup>1</sup> We note that in the group signature of [2], a related notion called testable weak zero-knowledge (TwZK) was introduced as an attempt to add identification functionality. However, TwZK was only against uniform adversaries.

**Constructions of witness-authenticating NIZK proofs.** With the definitions and models settled, we are now ready to discuss the constructions.

<u>Basic challenges.</u> A natural idea of our waNIZK construction is to attach an authentication tag to the NIZK proof, and augment it with a proof of the validity of the tag. Verification could be easy, while security posts several challenges. Since we want to remain "zero-knowledge" when the witness is unpredictable, the tag should not leak any other partial information. I.e., it should be "simulatable", even if the same witness is used to generate multiple proofs; further dealing with "unforgeability" incur extra difficulties in following different cases.

<u>Warm-up constructions</u>. Let us start with a special case where the identifier witness is uniform (or pseudorandom) even conditioned on all public parameters. For example, in group/ring signatures, the identifier witness is each user's secret key. We notice that simulatability can be realized by pseudorandomness, and we could simply use the witness as the key to generate the tag using a PRF. Namely,

$$\mathsf{Tag}_{\mathsf{PRF}}(w^{I}) = (t, \mathsf{PRF}(w^{I}, t)), \text{ for a random } t.$$

The "simulatability" and unforgeability of this tag are simply implied by the pseudorandomness, which further preserve the entropic zero-knowledgeness and unforgeability (the underlying NIZK should also satisfy certain "non-malleability" to prevent from modifying a valid proof). If the identifier uniqueness is in need, we can further require the collision-resistance of the PRF [19]. We remark that this solution that enables very efficient instantiations, could be readily useful.

A more general solution needs to deal with a general unpredictable sampler. We may apply a strong randomness extractor [32] to the identifier witness to pump out a uniform key, then apply PRF to generate the tag. Some subtle issues arise immediately: (1) the same witness as a random source may be used to generate multiple proofs (choosing different seeds). Thus, the extractor has to be re-usable thus requiring much more entropy (or the outer layer PRF needs to be related-key secure, which is only known for special relations); (2) a malicious prover might choose a "bad" seed to break the unforgeability, as the security of randomness extractor requires a uniform and independent seed. We resolve it by simply leveraging the common reference string, namely, using a part of CRS as the *fixed seed*. Thus the PRF is evaluated multiple times under the same key. However, as a consequence, this technique can only be applied to the setting that the statements are from a *CRS-independent* sampler.

<u>Full-fledged solution for CRS-dependent samplers.</u> In many applications (e.g., in all three applications we will show), the unpredictable sampler may be generated after the adversary sees the CRS; thus, it depends on the CRS. The construction now cannot simply obtain a string (e.g., the seed) from the CRS. Instead, we need to somehow "force" the honest behavior.

Let us examine the two soundness issues above: first, if we do not want to get into the difficulty of reusable extractor or related-key secure PRF, it is not clear how to force the same fixed seed (as in the CRS) to be used for every prover; moreover, proving a seed is generated uniformly already seems elusive. These obstacles motivate us to deviate from the Extract-then-PRF path. We first note that there are alternatives for "simulatability". Also, to ensure the honest generation of randomness (such as seed) used in generating the tag, we may explore a parameter with structure

Thus it can only be applied to more restricted languages (where the restrictions were informally described) and was impossible for non-uniform adversaries. Besides, soundness definition and provable constructions were not discussed.

or certain functionality that we can prove to bind the witness to the tag. Since we still need the identification function, those observations together lead us to the choice of deterministic public-key encryption (DPKE).

More precisely, let **DEnc** be the encryption algorithm of a DPKE scheme. We first generate a fresh public key pk, encrypt  $w^I$  to c under pk, and set (pk, c) as the tag. One can easily check whether w' is the encrypted message (identify here) w.r.t c by checking  $\mathsf{DEnc}(pk, w') \stackrel{?}{=} c$ .

Now for entropic ZK, we note that the DPKE can provide simulation security if the message is unpredictable. More importantly, this needs to hold even facing multiple proofs on potentially related statements. Viewing the statements as auxiliary input on the identifier witness, we can obtain those from DPKE with multi-user security with auxiliary inputs, which can be based on d-linear assumption [13]. Next, for soundness, and particularly unforgeability, we first need to ensure the well-formedness of pk. We can leverage the correctness of encryption and just prove a well-formedness of the ciphertext. Furthermore, "unforgeability" can be obtained by using a simulation-extractable NIZK proof.

We remark that our construction offers a framework that can have a hierarchy of instantiations. If we want the resulting waNIZK systems to have stronger (or weaker) property, we can instantiate the underlying NIZK correspondingly. For details, we refer to Sect.4.2.

**Applications.** Our abstraction of waNIZK can provide a new tool for many interesting applications. Here we will showcase three non-trivial applications in hash functions, (revocable) anonymous authentication, and encryption in more detail. Each of them advances the state of the art in the corresponding topic. We believe there are many more applications which we leave for future exploration.

Non-malleable (perfectly one-way) hash from standard assumptions. Many works have been around trying to realize partial properties of random oracles, ideally, via standard assumptions. Perfectly one-way hash and non-malleable hash are two important primitives for this purpose, in settings that include Bellare-Rogaway encryption scheme [6], HMAC[29], and OAEP [10].

Perfectly one-way hash requires its (randomized) evaluation algorithm to hide all partial information of the pre-image, even with some auxiliary inputs, while providing a verification algorithm to check the correctness of evaluation. Non-malleable hash requires that one cannot "maul" a hash value into a related one even with some auxiliary information about the pre-image <sup>2</sup>. Both of them also require collision resistance. Currently, perfectly one-way hash w.r.t general auxiliary inputs is only known to exist under a not-efficiently-falsifiable assumption [18], which was shown to contradict the existence of iO [16]; while non-malleable hash are either from perfectly one-way hash [9] or in the random oracle model [3]. Given the recent progress [33] on iO, the mere existence of non-malleable hash or perfectly one-way hash (with general auxiliary inputs) is still open.

We confirm the feasibility by presenting a new framework for non-malleable (perfectly one-way) hash functions from waNIZKs that can be based on the standard assumptions like the *d*-linear assumption. The starting point is to view the hash as a commitment that allows others to verify the committed value: it computationally determines an input and hides all partial information. This view inspires us to obtain a non-malleable (and perfectly one-way) hash by adding a proof of well-formedness of the commitment via waNIZKs where the input is set as the identifier witness. Perfect one-wayness comes from entropic ZK, collision resistance from identifier uniqueness, while non-malleability comes from (related-witness) unforgeability. For details, we refer to Sect. 5.

<sup>&</sup>lt;sup>2</sup> The simulation based non-malleability was shown to imply perfect one-wayness [9].

<u>Auxiliary-input group signatures with verifier-local revocation</u>. Group signatures [21] allow a user to sign a message on behalf of a group while hiding the signer's identity. A major issue is the revocation of users whose membership should be cancelled without influencing others. In group signatures with verifier-local revocation (VLR) [12], the signing procedure and the group public key will be independent of the revocation list, making this primitive appealing for systems providing attestation capabilities. Indeed, some instantiations of VLR group signatures such as the direct anonymous attestation scheme [14] have been already widely deployed in trusted platform modules (TPM) including Intel's SGX.

Many works have shown these TPMs are vulnerable to "side channel" attacks by which attackers could learn partial information about the secret key. Indeed, existing VLR group signature schemes [35,12,14,15,11] do not provide any security guarantee when auxiliary information about secret key is leaked. We therefore study the problem of leakage-resilient VLR group signature scheme, particularly, in the auxiliary-input model, the strongest model capturing one-time memory leakage.

Interestingly, a VLR group signature scheme necessarily relies on a secret-key-based tag generation which is identifiable (for revocation), unforgeable, and does not leak any partial information about the identity of the signer (for anonymity). Existing constructions leverage either algebraic approaches [35,12,14,15] or generic approaches such as PRFs [11] to realize the tag via "pseudorandomness", which will not hold anymore facing auxiliary-input leakage.

We solve this dilemma by using waNIZKs. Our idea is to simply replace the simulation sound NIZK in the folklore construction of group signatures ( for proving knowledge of a group membership certificate) with our waNIZK.

<u>Plaintext-checkable encryption in the standard model.</u> Plaintext-checkable encryption (PCE) is a public-key encryption [17], allowing one to search encrypted data with plaintext. Compared with DPKE [5], a PCE could still be randomized and provides a stronger security ensuring two ciphertexts encrypting the same message are unlinkable. Besides being a more fine-grained security notion, PCE has also been shown useful for constructing other primitives such as group signatures with verifier-local verification.

Existing constructions [17,37] are mostly secure in the random oracle model. However, in several scenarios, including the application to VLR group signatures [17] and achieving CCA-security via Naor-Yung [38], we need to prove properties about the plaintext of a PCE ciphertext via NIZKs. Random oracles clearly become unfavorable. Attempts exist [17,37,36] for standard-model PCE, but unfortunately they only work for uniform message distributions. In most scenarios plaintext messages are unlikely uniformly distributed. It follows that designing a standard-model plaintext-checkable encryption scheme for general (biased) message distributions is a natural open question.

We also answer this question and present a general framework for plaintext-checkable encryption, from any standard-model IND-CPA secure PKE and waNIZKs. Our idea is simple: we first encrypt m with the PKE and then prove the ciphertext is well-formed by using waNIZKs and setting m as the identifier witness. This framework naturally gives standard-model instantiations. Moreover, the identifier witness in our full-fledged construction is only required to be unpredictable, which allows to remove the restriction of uniform messages.

**Discussion: the benefit of abstracting out waNIZK.** We were faced with the decision of how to approach these applications: either individually addressing them or treating the tag as a broader technique. However, we opted to introduce this additional attribute into NIZK proofs and initiated a comprehensive investigation for several compelling reasons. Firstly, by enhancing a NIZK proof

system with specific functionalities, we not only pave the way for innovative constructions geared towards privacy and accountability-driven applications, but we also unearth potential applications in domains that may not seem directly related at first glance. Secondly, although we currently provide modular constructions, it's plausible that direct constructions could emerge without the need for explicit tags. This implies that the overall construction for specific applications through our proposed framework, known as waNIZK, might exhibit enhanced compactness. Thirdly, while the identifying functionality we explore is a basic one, it's crucial to recognize that a more encompassing Functional NIZK approach could introduce an intriguing "information-carrying proof" paradigm that stands independently from the captivating proof-carrying data paradigm. Such an avenue of exploration has the potential to stimulate fresh conceptual breakthroughs in the field.

# 2 Preliminary

**Notations.** Throughout the paper, we use  $\lambda$  for security parameter. For an NP language L, we let  $R_L$  denote its membership verification relation;  $(x, w) \in R_L$  or  $w \in R_L(x)$  denote that  $R_L(x, w) = 1$ ,  $R_L(x)$  denote the set of all witnesses of x, and  $L_n$  denote  $L_n = L \bigcap \{0, 1\}^n$ .

We use [i, n] to denote the set  $\{i, i + 1, \dots, n\}$  where i, n are two integers and i < n. We may abbreviate [1, n] as [n]. We say a function f(n) is negligible in n, denoted by  $f(n) \leq \text{negl}(n)$ , if we have  $\forall$  integer c > 0,  $\exists n_0, \forall n > n_0, f(n) < n^{-c}$ . A non-negligible function f(n) is denoted by f(n) > negl(n).

By a "non-uniform PPT adversary", we mean a polynomial-time probabilistic Turing machine M along with an infinite set of advice strings  $\{\varphi_n\}_{n\in\mathbb{N}}$ , where M can read  $\varphi_n$  when taking an n-bit input, and the length of  $\varphi_n$  is polynomial in n. All adversaries considered in this work are non-uniform PPT. When there is no ambiguity, we may simply call such an adversary an efficient adversary.

For a set  $X, x \leftarrow X$  denotes sampling x from the uniform distribution over X. For a distribution  $X, x \leftarrow X$  denotes sampling x from X. If A is a probabilistic algorithm,  $A(x_1, x_2, \dots; r)$  is the result of running A on the input  $x_1, x_2, \dots$  and the random coins r. We use  $y \leftarrow A(x_1, x_2, \dots)$  to denote the experiment that choosing r at random and getting  $y = A(x_1, x_2, \dots; r)$ .

#### 2.1 Definitions of Non-interactive Proof Systems

A non-interactive proof system  $\Pi$ , for an NP language L, enables the prover, who holds a witness of an instance  $x \in L$ , to convince the verifier that  $x \in L$  via a single proof. Typically, it can be described by the following a triple of probabilistic polynomial-time (PPT) algorithms:

- $-\sigma \leftarrow \mathsf{Setup}(1^{\lambda})$ . The setup algorithm outputs a CRS  $\sigma$ .
- $-\pi \leftarrow \mathsf{Prove}(\sigma, x, w)$ . The prover algorithm takes as inputs the CRS  $\sigma$ , an instance  $x \in L$  with its witness  $w \in R_L(x)$ , and outputs a string  $\pi$  called a proof.
- $-b \leftarrow \text{Verify}(\sigma, x, \pi)$ . The verifier algorithm takes as inputs  $\sigma$ , an instance x and a proof  $\pi$ , and outputs either 1 accepting it or 0 rejecting it.

We present NIZK definitions by following [41].

**Definition 1 (Non-interactive proof system).** Let L be an NP language. A non-interactive proof system  $\Pi = (\text{Setup}, \text{Prove}, \text{Verify})$  for L should satisfy:

1. Perfect Completeness: For all security parameters  $\lambda \in \mathbb{N}$  and for all  $x \in L_{\lambda}$  and  $w \in R_L(x)$ ,

$$\Pr[\sigma \leftarrow \mathsf{Setup}(1^{\lambda}); \pi \leftarrow \mathsf{Prove}(\sigma, x, w) : \mathsf{Verify}(\sigma, x, \pi) = 1] = 1$$

2. Adaptive Soundness: For any non-uniform PPT P<sup>\*</sup>, we have,

$$\Pr[\sigma \leftarrow \mathsf{Setup}(1^{\lambda}); (x, \pi) \leftarrow P^*(\sigma) : \mathsf{Verify}(\sigma, x, \pi) = 1 \land x \notin L_{\lambda}] \le \mathsf{negl}(\lambda).$$

In the above definition, the malicious prover  $P^*$  is restricted to be computational bounded. In some literature, this soundness is known as adaptive computational soundness, and a protocol (Setup, Prove, Verify) with completeness and computational soundness is called a non-interactive argument system. In this paper, we do not distinguish a proof system and an argument system.

The knowledge soundness (a.k.a., proof of knowledge) is a stronger variant of soundness, which captures one can prove a statement only when he knows the witness.

**Definition 2 (Knowledge soundness).** Let  $\Pi$  be a non-interactive proof system for an NP language L. We say  $\Pi$  satisfies the knowledge soundness, if there is an efficient knowledge extractor (Ext<sub>0</sub>, Ext<sub>1</sub>), s.t., for any non-uniform PPT adversary A, the output of Ext<sub>0</sub> and that of Setup are indistinguishable, i.e.

$$|\Pr[(\sigma,\xi) \leftarrow \mathsf{Ext}_0(1^{\lambda}) : 1 \leftarrow \mathcal{A}(\sigma)] - \Pr[\sigma \leftarrow \mathsf{Setup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}(\sigma)]| \le \mathsf{negl}(\lambda),$$

and  $\mathsf{Ext}_1$  can extract a witness from any valid proof:

$$\Pr\begin{bmatrix} (\sigma,\xi) \leftarrow \mathsf{Ext}_0(1^\lambda), (x,\pi) \leftarrow \mathcal{A}(\sigma), (w) \leftarrow \mathsf{Ext}_1(\sigma,\xi,x,\pi) :\\ \mathsf{Verify}(\sigma,x,\pi) = 1 \land w \notin R_L(x) \end{bmatrix} \leq \mathsf{negl}(\lambda)$$

The adaptive unbounded NIZK ensures that polynomial many proofs under the same CRS will not leak anything beyond the validity of statements, even if these proofs are generated on adversarially chosen statements. In this work, the zero-knowledge property refers to adaptive unbounded NIZK unless otherwise specified.

**Definition 3 (Adaptive unbounded NIZK ).** Let  $\Pi = ($ Setup, Prove, Verify) be a NIPS. It satisfies the zero-knowledge property, if there is a PPT simulator

(SimSetup, SimProve), s.t. for every nonuniform polynomial-time adversary A, we have

$$\Pr[\sigma \leftarrow \mathsf{Setup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_1(\sigma,\cdot,\cdot)}(\sigma)] - \\\Pr[(\sigma,\tau \leftarrow \mathsf{Sim}\mathsf{Setup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_2(\sigma,\tau,\cdot,\cdot)}(\sigma)] \le \mathsf{negl}(\lambda).$$

Both the oracles  $\mathcal{O}_1$  and  $\mathcal{O}_2$  take as input a pair  $(x, w) \in R_L(x)$ . While  $\mathcal{O}_1$  returns  $\pi \leftarrow \mathsf{Prove}(\sigma, x, w)$ ,  $\mathcal{O}_2$  returns  $\pi \leftarrow \mathsf{SimProve}(\sigma, \tau, x)$ . The probability is taken over the randomness of algorithms Setup, Prove, SimSetup, SimProve and  $\mathcal{A}$ .

We call a non-interactive proof system satisfying zero-knowledge property a NIZK.

The zero-knowledge property does not exclude that a verifier, after seeing a proof, can obtain the ability of proving what he cannot prove before. To address this issue, there are enhanced soundness definitions including simulation soundness, and simulation-extractability. Simulation soundness says one cannot prove a false statement even after seeing simulated proofs.

**Definition 4 (Simulation soundness).** Let  $\Pi$  be a NIZK for an NP language L, and (SimSetup, SimProve) be its simulator. We say  $\Pi$  satisfies the simulation soundness, if for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\Pr\begin{bmatrix} (\sigma, \tau) \leftarrow \mathsf{SimSetup}(1^{\lambda}); (x^*, \pi^*) \leftarrow \mathcal{A}^{\mathcal{O}_2(\sigma, \tau, \cdot)}(\sigma) :\\ \mathsf{Verify}(\sigma, x^*, \pi^*) = 1 \land x^* \notin L \end{bmatrix} \le \mathsf{negl}(\lambda).$$

The simulation extractability ensures that after seeing polynomial many simulated proofs a verifier cannot prove anything it cannot prove before except duplicating the simulated proofs.

**Definition 5 (Simulation extractability).** Let  $\Pi$  = (Setup, Prove, Verify) be an adaptive unbounded NIZK proof system, and let (SimSetup, SimProve) be its simulator. We say the NIZK proof system is simulation-extractable, if there exist two PPT algorithms SE and Ext, s.t. for every polynomial-time non-uniform adversary ( $A_1, A_2$ ), we have

$$\Pr[(\sigma, \tau) \leftarrow \mathsf{SimSetup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}_1(\sigma, \tau)] - \\\Pr[(\sigma, \tau, \xi) \leftarrow \mathsf{SE}(1^{\lambda}) : 1 \leftarrow \mathcal{A}_1(\sigma, \tau)] \leq \mathsf{negl}(\lambda);$$

and

$$\Pr[(\sigma,\tau,\xi) \leftarrow \mathsf{SE}(1^{\lambda}); (x,\pi) \leftarrow \mathcal{A}_2^{\mathcal{O}(\sigma,\tau,\cdot)}(\sigma); w \leftarrow \mathsf{Ext}(\sigma,\xi,x,\pi): \\ ((x,\pi) \notin \mathsf{Hist}) \land ((x,w) \notin R_L) \land (\mathsf{Verify}(\sigma,x,\pi)=1)] \le \mathsf{negl}(\lambda),$$

where the oracle  $\mathcal{O}$  takes as input a string x and returns  $\pi \leftarrow \mathsf{SimProve}(\sigma, \tau, x)$ , and Hist denotes the query history of  $\mathcal{O}$ . The probability is taken over the randomness of algorithms SimSetup, SE,  $\mathcal{A}_1, \mathcal{A}_2, \mathsf{Ext}$ .

Assumptions. We note that a general adaptive unbounded NIZK proof system with simulation soundness exists assuming the existence one-way functions and a general adaptive NIZK proof system [41], while the latter can be based on the trapdoor permutations or the LWE assumption. In addition, a simulation-extractable NIZK makes an extra assumption on the existence of public-key encryption schemes.

#### 2.2 Definitions of (Computational) Entropy and Randomness Extraction

Conditional entropy measures the randomness of a random variable conditioned on some correlated event.

**Definition 6 (Conditional min-entropy** [32]). For a joint random variable (X, Z), we say X has at least k min-entropy conditioned on Z, denoted by  $\mathbf{H}_{\infty}(X|Z) \geq k$ , if

$$-\log(\mathbf{E}_{(z\leftarrow Z)}[\max_{x\in\mathsf{Sup}(X)}\Pr[X=x|Z=z]])\geq k,$$

where Sup(X) is the support of X and  $\mathbf{E}[\cdot]$  denotes an expect of a random variable.

Computational entropy is to quantify the appearance of entropy of a random variable X, for computationally bounded observers, which is usually more than what X really has. Hill entropy and unpredictability entropy are widely used computational entropy notions. Particularly, X has k Hill entropy only when it is computationally indistinguishable with a random variable Y with k-min-entropy.

Unpredictability entropy [32] is a kind of computational entropy, describing the maximum predicting probability of a random variable for computational bounded predictors.

**Definition 7 (Unpredictability entropy [32]).** For a joint random variable (X, Z), we say X has unpredictability entropy at least k conditioned on Z, denoted by  $\mathbf{H}^{unp}(X|Z) \ge k$ , if there exists a collection of distributions  $\{Y|_z\}$  (giving rise to a joint distribution (Y, Z)), such that (Y, Z) and (X, Z) are computationally indistinguishable, and for any non-uniform PPT algorithm  $\mathcal{A}$ , we have

$$\Pr[(y, z) \leftarrow (Y, Z) : \mathcal{A}(z) = y] \le 2^{-k}.$$

Remark 1. In the original definition [32], the unpredictability entropy of (X, Z), denoted by  $\mathbf{H}_{\epsilon,s}^{unp}$ , is additionally parameterized by s and  $\epsilon$ . s is the circuit size of an adversary  $\mathcal{A}$ , and  $\epsilon$  is the advantage of  $\mathcal{A}$  to distinguish (X, Z) with (Y, Z).

The definition above could be regarded as an enhancement of the original definition. Concretely, we let  $2^{-k}$  be the upper bound on the advantage of all polynomial-size circuits. Therefore, all results on the original definition also hold on this one. We do such modifications to fit the definition of the expression of asymptotic complexity.

It is easy to see conditional min-entropy implies unpredictability entropy. However, for a joint random variable (X, Z) with  $\mathbf{H}^{\mathrm{Unp}}(X|Z) \geq k$ , it is possible that X is information-theoretically determined by Z, and thus X does not have any min-entropy conditioned on Z. For example, consider a Discrete-log group  $\mathbb{G}$  with order q. Let X a random variable uniformly sampled from  $\mathbb{Z}_q$ , and let Z be  $(g^X)$  where g is a generator of  $\mathbb{G}$ . Obviously, X is information-theoretically determined by Z. However, predicting X given Z is believed to be exponentially hard.

#### 2.3 Definitions of Deterministic Public-key Encryption

A deterministic public-key encryption (DPKE) scheme  $\Sigma$  is defined by a triple of algorithms {KeyGen, Enc, Dec} where Enc and Dec are deterministic. Below, we first formalize the hard-to-invert auxiliary inputs, and then introduce the PRIV-IND-MU-security with respect to  $\epsilon$ -hard-to-invert auxiliary inputs, by following definitions presented in [13].

**Definition 8.** A (randomized) function  $\mathcal{F} = \{f_{\lambda}\}_{\lambda \in \mathbb{N}}$  is  $\epsilon(\lambda)$ -hard-to-invert with respect to an efficient samplable distribution  $\mathcal{D} = \{\mathcal{D}_{\lambda}\}_{\lambda \in \mathbb{N}}$ , if for every non-uniform PPT adversary  $\mathcal{A}$  it holds that

$$\Pr[x \leftarrow \mathcal{D}_{\lambda}; \mathcal{A}(1^{\lambda}, f_{\lambda}(x)) = x] \le \epsilon(\lambda),$$

for all sufficient large  $\lambda$ , where the probability is taken over the randomness of  $\mathcal{A}$  and  $f_{\lambda}$ .

**Definition 9.** Perfect correctness A DPKE scheme  $\Sigma$  is perfectly correct, if there is an NP relation  $R_{pk}$ , such that

$$\Pr[(pk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda}) : (pk, sk) \in R_{pk}] = 1,$$

and for every  $(pk, sk) \in R_{pk}$  and every message m, it holds that

$$\Pr[\mathsf{Dec}(sk,\mathsf{Enc}(pk,m)) = m] = 1.$$

**Definition 10.** A DPKE scheme  $\Sigma$  is PRIV-IND-MU-secure with respect to  $\epsilon$ -hard-to-invert auxiliary inputs, if for any non-uniform PPT adversary  $\mathcal{A}$ , for any efficiently samplable distributions  $\mathcal{M} = \{\mathcal{M}_{\lambda}\}_{\lambda \in \mathbb{N}}$ , and for any function  $\mathcal{F} = \{f_{\lambda}\}_{\lambda \in \mathbb{N}}$  that is  $\epsilon$ -hard-to-invert w.r.t.  $\mathcal{M}$ , such that

$$|\Pr[\mathsf{Exp}_{\mathcal{D},\mathcal{A},\mathcal{M},\mathcal{F}}^{\mathsf{priv},0}(\lambda)=1] - \Pr[\mathsf{Exp}_{\mathcal{D},\mathcal{A},\mathcal{M},\mathcal{F}}^{\mathsf{priv},1}(\lambda)=1]| \le \mathsf{negl}(\lambda).$$

The experiments are defined as follows, where L is a polynomial-bounded integer function.

$Exp^{priv,0}_{\varSigma,\mathcal{A},\mathcal{M},\mathcal{F}}(\lambda)$	$Exp^{priv,1}_{\varSigma,\mathcal{A},\mathcal{M},\mathcal{F}}(\lambda)$
$\overline{m_0 \leftarrow \mathcal{M}_\lambda}$	$m_0 \leftarrow \mathcal{M}_\lambda$
for $i \in [L(\lambda)]$	for $i \in [L(\lambda)]$
$(pk_i, sk_i) \leftarrow KeyGen(1^{\lambda})$	$m_i \leftarrow \{0,1\}^{ m_0 }$
$c_i \leftarrow Enc(pk_i, m_0)$	$(pk_i, sk_i) \leftarrow KeyGen(1^{\lambda})$
$b \leftarrow \mathcal{A}(\{(pk_i, c_i)\}_{i \in [\ell(\lambda)]}, f_{\lambda}(m_0))$	$c_i \leftarrow Enc(pk_i, m_i)$
	$b \leftarrow \mathcal{A}(\{(pk_i, c_i)\}_{i \in [\ell(\lambda)]}, f_{\lambda}(m_0))$

We remark our definition is slightly weaker than [13, Definition 4.5], as the message being encrypted under different public keys, are identical, while the original definition allows messages to be related w.r.t. affine transformations. We present the weaker form since it is what we need for our construction and can be satisfied by the *d*-linear-based construction presented in [13].

#### 3 Syntax and Security Models of waNIZKs

As explained in the introduction, we consider a non-interactive proof system working for an NP language L, where a statement may have multiple witnesses. There is an extra mechanism Identify, such that anyone having a witness  $w \in R_L(x)$  can efficiently check whether a proof  $\pi$  for  $x \in L$  was generated using w. On the other hand, we require such mechanism to be robust, *i.e.*, anyone who does not know w cannot produce a valid proof for  $x \in L$  that will be identified as generated from w. We call such a proof system a *witness-authenticating* non-interactive proof system (waNIPS), since now every proof essentially is authenticated by the corresponding witness. Though intuitive, formulating the new properties while adapting existing properties turns out to be involved.

**Identifier witness.** We first notice that the straightforward formulation of waNIPS, in which the extra identification algorithm **Identify** takes the *whole* witness, sometimes, limits the applications – some part of witness, such as the randomness (or other information) used for generating the proof, may not be functionally important or even be available, but are still required for the identification.

Consider a class of applications (including the non-malleable hash and plaintext-checkable PKE applications that we will present soon), in which we may just use the proof to carry a bit covertly that can be extracted by Identify. Now other users who may know the actual *secret* witness cannot figure out the randomness freshly sampled by prover during the proof generation; thus, they will not be able to run Identify. It is easy to see that the actual secret is necessary and sufficient for the identification purpose.

We thus consider the notion of identifier witness. Formally, for a statement  $x \in L$ , its witness  $w = (w^I, w^{NI})$  consists of an identifier part  $w^I$  and a non-identifier part  $w^{NI}$ , where  $w^I$  will be explicitly specified by a relation  $R_L^I$  (called an identifier relation of L),  $R_L^I((x, w^{NI}), w^I) = 1$ , or

 $w^I \in R_L^I(x)$  for short. Now we only need the identifier witness for the identification algorithm.<sup>3</sup> Formally, we provide definitions below.

**Definition 11 (waNIPS).** Let L be an NP language, and  $R_L^I$  be an identifier relation of L. A waNIPS on  $(L, R_L^I)$  is defined by four efficient algorithms:

- $-\sigma \leftarrow \mathsf{Setup}(1^{\lambda})$ . The setup algorithm outputs a CRS  $\sigma$ .
- $-\pi \leftarrow \mathsf{Prove}(\sigma, x, w)$ . The prover algorithm takes as inputs  $\sigma$ , an instance  $x \in L$  with its witness  $w \in R_L(x)$ , and outputs a string  $\pi$  called a proof.
- $-b \leftarrow \text{Verify}(\sigma, x, \pi)$ . The verifier algorithm takes as inputs  $\sigma$ , an instance x and a proof  $\pi$ , and outputs either 1 accepting it or 0 rejecting it.
- $-d \leftarrow \text{Identify}(\sigma, x, \pi, w^I)$ . This algorithm takes as input a valid proof  $\pi$  for some  $x \in L$  and a string  $w^I$ . It returns either 1 indicating  $\pi$  was generated by a witness in the form of  $(w^I, \star)$ , or 0 otherwise.

The first three describe a non-interactive proof system for L. We say  $\pi$  is authenticated by  $w^{I}$  if Identify $(\sigma, x, \pi, w^{I}) = 1$ .

*Completeness* of waNIPS could be easily defined by describing the identification functionality and the proving functionality over honestly generated proofs, which covers the standard completeness of non-interactive proof systems.

**Definition 12 (Completeness of waNIPS).** We say a waNIPS for  $(L, R_L^I)$  is complete, if for every  $x \in L_{\lambda}$ ,  $(w^I, w^{NI}) \in R_L(x)$ , for  $\sigma \leftarrow \text{Setup}(1^{\lambda}), \pi \leftarrow \text{Prove}(\sigma, x, (w^I, w^{NI}))$ , the following holds:

 $\Pr[\mathsf{Verify}(\sigma, x, \pi) = 1 \land \mathsf{Identify}(\sigma, x, \pi, w^I) = 1] = 1.$ 

#### 3.1 Defining unpredictable sampler

**Incompatibility between identification and zero-knowledgeness.** Before introducing the formal security definitions, we first clarify a basic question: when is a waNIZK meaningful (or even feasible)? The question arises given that the identification functionality is clearly incompatible with the standard zero-knowledge property.

As a concrete example, consider the range proof system where we use a NIZK to prove a committed integer m w.r.t. a commitment *com* belongs to a range, say (1, 20). Seeing such a proof, the adversary learns nothing about m except its range. However, if we use a waNIZK to support identification, then everyone can simply check for all values in (1, 20) and completely recover the value of m! This simple example hints a trivial impossibility for conventional zero-knowledgeness of waNIZK, for the languages whose identifier witness can be easily guessed. Similar situation appears in other settings, e.g., encryption schemes equipped with a plaintext-search functionality [5].

It follows that we should focus on "hard" statements that one cannot guess the identifier witnesses easily. The notion that a statement is "hard" clearly cannot stand in the worst case if we are considering a non-uniform adversary, since its advice string may encode the witness already. We thus consider a distribution over a language such that for any efficient adversary, a random sample from this distribution is "hard" to predict, and a waNIZK proof system is expected to work for languages admitting such "hard" distributions.

<sup>&</sup>lt;sup>3</sup> We stress that the notion of identifier witness *does not put any restriction* on the languages that can be proved, as the non-identifier part can be empty. In this case, the identifier part is simply the whole witness.

A natural way to describe a distribution is to specify an (adversarial) sampler G which is a non-uniform PPT algorithm, which on input a security parameter outputs an element  $x \in L_{\lambda}$  and its witness  $(w^{I}, w^{NI}) \in R_{L}(x)$ <sup>4</sup>. The unpredictability of this sampler can then be quantified by unpredictability entropy [32] of the identifier witness  $w^{I}$ . More precisely, G is k-unpredictable when  $\mathbf{H}^{unp}(W^{I}|X) \geq k(\lambda)$ , where  $(X, W^{I}, W^{NI})$  is a joint random variable output by  $G(1^{\lambda})$ . While such a formulation is simple, we find it unnecessarily restrictive in certain situations. We present a more general formulation below.

Modeling a more general unpredictable sampler. When applying our waNIZKs in a larger cryptosystem, the statement may involve system parameters that are not under the control of the adversary. This seemingly minor point is actually essential. A subtle issue is that letting the adversarial sampler to generate the whole statement sometimes makes it hard to enforce the unpredictability of witness. For example, consider a public-key encryption scheme and a simple language  $L_{\mathsf{Enc}} := \{(pk, c); (m, r) : c = \mathsf{Enc}(pk, m; r)\}$  where m is the identifier witness. Let  $G_{pk}$  be the following sampler:

 $pk = pk^{*}, m \leftarrow M_{\lambda}, r \leftarrow \{0, 1\}^{\lambda}, c = \operatorname{Enc}(pk^{*}, m; r) :$ return  $(x = (pk^{*}, c), w^{I} = m, w^{NI} = r),$ 

where  $M_{\lambda}$  is a high-entropy message distribution. Is  $G_{pk}$  an unpredictable distribution? In general, the answer is no since the adversary might have the secret key sk of  $pk^*$ . However, simply excluding such a sampler is not the right choice. In typical applications (for example, in our application of plaintext checkable encryption, cf. Sec.7), the public key is generated honestly and not under the control of the adversary. And the message distribution is specified by the adversary after seeing the public key.

This oddity arises due to that the larger system where a waNIZK is employed already requires some honestly generated parameter. To capture this intuition, we define a separate parameter generation as a PPT algorithm PG. We let the sampler algorithm to take as input the parameter pp generated by PG, asking the distribution conditioned on PG = pp to be unpredictable. Note that PG is not a part of our waNIZK syntax, usually specified by the applications. We remark that this is optional (which could be empty if there is no PG in the application).

Modeling CRS-dependent unpredictability. As a waNIZK assumes a CRS, which is publicly available and usually generated once for all, in some scenarios, adversaries might be able to specify an unpredictable sampler after seeing the CRS. In this most general case, we allow the adversary and the sampler algorithm G to take CRS as an input.

One tricky issue exists when measuring the unpredictability of the output (particularly the identifier witness) of a CRS-dependent sampler. At the first glance, the minimal requirement should be as follows: the adversary, who specified the sampler after seeing an honestly generated CRS, cannot find the identifier for a statement outputted by this sampler (on input this CRS). However, we got stuck with the case that the statement itself could be containing CRS-dependent auxiliary input of the identifier witness. Consider the following extreme example.  $L^*$  is an NP language such that a statement  $x^* = (x, \star) \in L^*$  iff x belongs to another NP language L, and  $G^*$  is a sampler that uses an unpredictable sampler G for L to produce  $(x, w^I, w^{NI})$  and outputs  $(x^* = (x, \pi), w^I, w^{NI})$  where  $\pi$  is a valid proof demonstrating  $x \in L$  (thus  $x^* \in L^*$ ). On the one hand,  $G^*$  should

<sup>&</sup>lt;sup>4</sup> Note that in general, it is unclear how to generate a witness from a statement, so we let the sampler to output  $x, w^{NI}$  together with  $w^{I}$ , but we put no restrictions on them. In principle,  $x, w^{NI}$  could even be fixed by the attacker and hardcoded into G as long as an unpredictable  $w^{I}$  can be generated.

be unpredictable since a "zero-knowledge" proof won't give the adversary extra advantage. On the other hand, it destroys knowledge soundness. If a malicious prover simply outputs such a hardcoded proof, she generates a proof without knowing any witness! More serious issues will occur at a new "unforgeability" property we will introduce. We will give a more detailed discussion when we present the soundness definitions (see Remark 4).

To rule out those trivial "attacks", we have to exclude such samplers. We observe that "zero knowledgeness" holds only when the CRS is honestly genereated such that the adversary does not have auxilairy information about it; certain trapdoor information could enable the adversary to recover the witness from the proof. We leverage this observation and define unpredictability more "aggressively": a sampler (which is an algorithm taking input as a CRS) is unpredictable if for any adversary and on input any CRS (about which the adversary may have arbitrary auxiliary information), the adversary cannot find the sampled identifier witness given the statement.

Taking all above discussions into consideration, we present the formal definition of unpredictable samplers.

**Definition 13 (Unpredictable sampler).** Let G be a sampler for  $(L, R_L^I)$  and PG be a trusted parameter generation procedure. Let the random variable PP be the output of  $PG(1^{\lambda})$ , and define a class of random variables  $\{(X_{\sigma}, W_{\sigma}^I, W_{\sigma}^{NI})\}_{\sigma \in Supp(Setup(1^{\lambda}))}$  where  $(X_{\sigma}, W_{\sigma}^I, W_{\sigma}^{NI}) = G(PP, \sigma)$ . Then, we say G is k-unpredictable w.r.t. PG, if for any CRS  $\sigma$ , it always holds that

$$\mathbf{H}^{\mathsf{unp}}(W^{I}_{\sigma}|X_{\sigma}, PP) \ge k(\lambda).$$

Clearly, the basic requirement is  $k = \omega(\log \lambda)$ .<sup>5</sup>

If we are considering CRS-independent samplers, G simply does not take as input the CRS  $\sigma$ .

#### 3.2 Entropic zero-knowledgeness

We present a new definition of entropic zero-knowledgeness, ensuring that nothing else is leaked except the identification bit to attackers who know the exact identifier witness (to attackers who do not know the exact witness, actually the zero-knowledgeness remains). Or, to put it another way, to rule out the "trivial attacks" caused by the added identification functionality, we consider a zero-knowledge property w.r.t unpredictable samplers. Since now the attacker does not know the witness (while in conventional ZK, Definition. 3, witness is chosen by the attacker), we need to give the attacker the capability to learn extra side information from other related proofs using the same witness, and this again should exclude the trivial impossibilities. Formally defining this new property requires care. We illustrate the intuition and the definition below.

Integrating the unpredictable sampler. Let us first recall the conventional zero-knowledge property: for any statement x along with its witness w, the procedure that generates a CRS  $\sigma$ , and the procedure generating a valid proof  $\pi$  using  $(x, w, \sigma)$ , can be emulated by a simulator without using the witness. The adaptive counterpart further allows the attacker to specify a statement after seeing the CRS.

Now the identifier witness  $w^{I}$  (along with  $(x, w^{NI})$ ) is produced by an unpredictable sampler G, which is specified by the attacker. The prover (denoted as a prover oracle  $\mathcal{O}_{P1}$ ) takes the tuple

<sup>&</sup>lt;sup>5</sup> We can also measure the unpredictability by HILL entropy [32]. On the one hand, it brings more restrictions on the languages to be proven; On the other hand, for samplers with sufficient HILL entropy we can give more efficient constructions which we explain in details in Appendix A.

 $(x, w^{I}, w^{NI})$  from G and the CRS as input and generates a proof. We want that this proof can be simulated via a simulator (denoted as  $\mathcal{O}_{S1}$ ) without using the witness  $(w^{I}, w^{NI})$ .

Allowing attackers to learn side information from related proofs. In the conventional zero-knowledge definition, since the attacker (distinguisher) is given the witness, just asking the simulator to emulate the proof is sufficient. While in our new definition, since the distinguisher does not have the exact witness, directly plugging in the unpredictable sampler to the zero-knowledge definition is too weak, in the sense that the prover only proves once. But in practical applications, this is not the case. For example, in group signatures, adversaries are allowed to obtain multiple signatures, possibly for different messages, from one user. To lift this restriction, we will allow the distinguisher to adaptively obtain multiple proofs, which could be generated from independently sampled statements. Also, seeing a statement  $\bar{x}$ , which has the same identifier witness  $w^{I}$ .

Formally, we let the prover oracle  $\mathcal{O}_{P1}$  (or  $\mathcal{O}_{S1}$ ) be stateful, and augment a pair of new oracles  $\mathcal{O}_{P2}$  and  $\mathcal{O}_{S2}$ , which, with access to the states of  $\mathcal{O}_{P1}$  and  $\mathcal{O}_{S1}$ , take as inputs an index (that specifies a previously sampled tuple) and an extended sampler EG. Seeing x, EG generates an extended statement  $\bar{x}$  (and corresponding non-identifier witness), which is associated with the same  $w^{I}$ . However, an arbitrarily extended statement may leak the entire  $w^{I}$  even if the original statement hides it. To rule out the trivial impossibility, we put a restriction on the extended statement w.r.t a  $w^{I}$  that it will not leak more information than the original statement, and thus  $w^{I}$  is still unpredictable. We model this restriction by asking a "dual-mode" extended sampler.<sup>6</sup> In its real mode, it honestly extends the original tuple; in the sim mode, without using the original withess it generates a statement that is computationally indistinguishable with that in the real mode. The sim mode will be only used in the security proof.

**Definition 14.** We say  $\mathsf{EG} = \{\mathsf{EG}_{real}, \mathsf{EG}_{sim}\}$  is a dual-mode extended sampler w.r.t.  $\mathsf{PG}$ , if for any  $\sigma$  and any non-uniform  $PPT \mathcal{A} := (\mathcal{A}_0, \mathcal{A}_1)$ , the following holds:  $pp \leftarrow \mathsf{PG}$ ,  $(x, w^I, w^{NI}, st) \leftarrow \mathcal{A}_0(pp)$ ,  $(\bar{x}, \bar{w}^{NI}) \leftarrow \mathsf{EG}_{real}(pp, \sigma, x, w^I, w^{NI})$ ,  $\tilde{x} \leftarrow \mathsf{EG}_{sim}(pp, x)$ ,

$$\Pr[(w^I, \bar{w}^{NI}) \in R_L(\bar{x})] = 1 \land |\Pr[\mathcal{A}_1(\sigma, pp, \bar{x}, st) = 1] - \Pr[\mathcal{A}_1(\sigma, pp, \tilde{x}, st) = 1]| \le \mathsf{negl}(\lambda).$$

Remark 2. We require that the indistinguishability holds for  $(x, w^I, w^{NI})$  specified by the adversary, instead of a randomly sampled tuple. This requirement may unnecessarily exclude extended samplers that won't leak more information about a randomly sampled identifier witness (yet it still captures many natural cases including all our applications). However, if we just ask indistinguishability for a random tuple, it is hard to ensure the indistinguishability still holds when plugging the extended sampler into entropic ZK definition where the adversary could adaptively invoke extended samplers on a tuple for multiple times and see other auxiliary information. Instead, the indistinguishability at the current form will be trivially preserved since the adversary can have any auxiliary information about the input. We leave it as a feature work to figure out the best possible definition.

We are now ready to present the formal definition of entropic ZK.

<sup>&</sup>lt;sup>6</sup> In the conference version [28], this restriction is modeled by the existence of another "simulated" extender. Here we pack both them into a "dual-mode" one, to emphasize that the simulated one is easy to be found and thus facilitate the security proof. We also let the indistinguishability hold w.r.t. any instance specified by the adversary due to the reason in Remark 2.

**Definition 15 (Entropic ZK).** A waNIPS  $\Pi$  for  $(L, R_L^I)$  satisfies the (multi-theorem) entropic zero-knowledgeness w.r.t. a parameter generation procedure PG and a class of unpredictable samplers  $\mathcal{G}$ , if there is a PPT simulator {SimSetup, SimProve}, such that for every non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\left| \Pr \begin{bmatrix} \sigma \leftarrow \mathsf{Setup}(1^{\lambda}) \\ pp \leftarrow \mathsf{PG}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma) : \\ 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P1}}, \mathcal{O}_{\mathsf{P2}}}(\sigma, pp) \end{bmatrix} - \Pr \begin{bmatrix} (\sigma, \tau) \leftarrow \mathsf{Sim}\mathsf{Setup}(1^{\lambda}) \\ pp \leftarrow \mathsf{PG}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma) : \\ 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{S1}}, \mathcal{O}_{\mathsf{S2}}}(\sigma, pp) \end{bmatrix} \right| \le \mathsf{negl}(\lambda).$$

where the real prover oracles  $\mathcal{O}_{P1}, \mathcal{O}_{P2}$  and the simulator oracles  $\mathcal{O}_{S1}, \mathcal{O}_{S2}$  are defined in Fig.1. The sampler G should belong to  $\mathcal{G}$ . EG shall be a dual-mode extended sampler w.r.t. PG(cf. Def.14).

$\mathcal{O}_{P1}(\sigma, pp)$	$\mathcal{O}_{S1}(\sigma, \tau, pp)$
i + +;	i + +;
$(x_i, (w_i^I, w_i^{NI}), \boxed{z_i}) \leftarrow G(\sigma, pp);$	$(x_i, (w_i^I, w_i^{NI}), \boxed{z_i}) \leftarrow G(\sigma, pp);$
$st \leftarrow st \cup (i, x_i, (w_i^I, w_i^{NI}));$	$st \leftarrow st \cup (i, x_i, (w_i^I, w_i^{NI}));$
$\pi_i \leftarrow Prove(\sigma, x_i, w_i^I, w_i^{NI})$	$\pi_i \leftarrow SimProve(\sigma, \tau, x_i)$
<b>return</b> $(x_i, \pi_i)$	$\mathbf{return}\ (x_i,\pi_i)$
$\mathcal{O}_{P2}(\sigma, pp, x_i, EG, st)$	$\mathcal{O}_{S2}(\sigma, \tau, pp, x_i, EG; st)$
$\boxed{\text{Find}(i, x_i, (w_i^I, w_i^{NI}), \boxed{z_i}) \in st}$	$\overline{\text{Find}(i, x_i, (w_i^I, w_i^{NI}), [z_i])} \in st$
$(\bar{x}, \bar{w}^{NI}) \leftarrow EG_{real}(pp, \sigma, x_i, (w_i^I, w_i^{NI}))$	$(\bar{x}, \bar{w}^{NI}) \leftarrow EG_{\mathtt{real}}(pp, \sigma, x_i, (w_i^I, w_i^{NI}))$
$  \bar{\pi} \leftarrow Prove(\sigma, \bar{x}, w_i^I, \bar{w}^{NI})$	$\bar{\pi} \leftarrow SimProve(\sigma, \tau, \bar{x})$
<b>return</b> $(\bar{x}, \bar{\pi})$	$\mathbf{return}\ (\bar{x},\bar{\pi})$

**Fig. 1.** The oracles.  $\mathcal{O}_{P1}$  (resp.  $\mathcal{O}_{S1}$ ) and  $\mathcal{O}_{P2}$  (resp.  $\mathcal{O}_{S2}$ ) share the state *st* which is initialized to be  $\emptyset$ . The counter *i* is initialized to be 0. The boxed items are only available for auxiliary-input definitions (see Sect.3.4)

*Remark 3.* Entropic ZK for CRS-independent samplers can be easily obtained by removing the CRS from the input of G and the first stage of A; it also suffices in interesting applications and admits more efficient constructions.

#### 3.3 Soundness definitions

The conventional (knowledge) soundness of non-interactive proof systems ensures that a prover that can generate a valid proof must possess a witness. In our setting with an extra identification functionality, we essentially require the identifier witness to be "committed" to the proof. Naturally, the soundness property also needs to be upgraded. In particular, we would need to ensure that a used witness must be identifiable; and a malicious prover could not "forge" a proof that points to a witness that is not known to her. (1) The former property can be realized augmenting the conventional knowledge soundness such that: from a valid proof, a witness not only can be extracted but also is bound to the proof. (2) The latter models that an attacker has access to multiple witnesses for a statement but still cannot frame any others that hold another witness unknown to the attacker. We call it *unforgeability*. Formulating those notions turns out to be highly involved, especially when considering slightly more advanced notions.

Authenticating knowledge soundness. As briefly discussed above, we now require a witness extractable from the proof to authenticate the proof. Formally, we have the following definition:

**Definition 16 (Authenticating knowledge soundness).** We say a waNIPS  $\Pi$  for  $(L, R_L^I)$  satisfies the authenticating knowledge soundness, if there exists a PPT extactor  $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ , s.t., for any non-uniform PPT adversary  $\mathcal{A}$ , (1) the output of  $\mathsf{Ext}_0$  is computationally indistinguishable with the real CRS:

 $|\Pr[(\sigma,\xi) \leftarrow \mathsf{Ext}_0(1^\lambda) : 1 \leftarrow \mathcal{A}(\sigma)] - \Pr[\sigma \leftarrow \mathsf{Setup}(1^\lambda) : 1 \leftarrow \mathcal{A}(\sigma)]| \le \mathsf{negl}(\lambda),$ 

and (2) any valid proof must be authenticated by the extracted witness:

$$\Pr\begin{bmatrix} (\sigma,\xi) \leftarrow \mathsf{Ext}_0(1^{\lambda}), (x,\pi) \leftarrow \mathcal{A}(\sigma), (w^I, w^{NI}) \leftarrow \mathsf{Ext}_1(\sigma,\xi,x,\pi) :\\ \mathsf{Verify}(\sigma,x,\pi) = 1 \land \left[ (w^I, w^{NI}) \notin R_L(x) \lor \mathsf{Identify}(\sigma,x,\pi,w^I) \neq 1 \right] \end{bmatrix} \le \mathsf{negl}(\lambda)$$

We also consider a weaker definition called authenticating soundness that only requires the existence of such  $(w^I, w^{NI})$  instead of that  $\mathcal{A}$  must know the witness. This notion will be useful when the knowledge extraction procedure can be done by external primitives such as PKE.

**Definition 17 (Authenticating soundness).** We say a waNIPS  $\Pi$  for  $(L, R_L^I)$  satisfies the authenticating soundness, if for any non-uniform PPT adversary  $\mathcal{A}$ , we have

$$\Pr[\sigma \leftarrow \mathsf{Setup}(1^{\lambda}); (x, \pi) \leftarrow \mathcal{A}(\sigma) : \text{ if } \mathsf{Verify}(\sigma, x, \pi) = 1, \text{ then} \\ x \in L \text{ and } \exists w^{I} \in R_{L}^{I}(x), \text{ s.t. } \mathsf{Identify}(\sigma, \pi, w^{I})] \geq 1 - \mathsf{negl}(\lambda).$$

**Unforgeability.** This property captures the "authenticity" that an adversary cannot forge a proof that will be authenticated by an identifier witness that the adversary does not know. Like our entropic ZK definition, we will leverage the unpredictable sampler for  $(L, R_L^I)$  to capture an unpredictable target witness. More importantly, we would like this to hold even if the adversary can adaptively obtain many proofs from witnesses unknown to her (the "forgery" thus should be a new proof) as she wishes. Note that this property indeed ensures that an adversary cannot simply "maul" a proof, and thus it (along with authenticating knowledge soundness) will suffice for many applications (such as non-malleable hash and VLR group signatures) which originally need a simulation-extractable NIZK for realizing non-malleability.<sup>7</sup>

**Definition 18 (Unforgeability).** Let  $\Pi$  be a waNIPS for  $(L, R_L^I)$ . We say  $\Pi$  satisfies unforgeability w.r.t. PG and a collection of unpredictable samplers  $\mathcal{G}$  (cf. Def.13), if for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\Pr \begin{bmatrix} pp \leftarrow \mathsf{PG}(1^{\lambda}); \sigma \leftarrow \mathsf{Setup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \overline{\sigma}); \\ (x^*, \pi^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma, pp) : (x^*, \pi^*) \notin \mathsf{Hist} \\ \wedge \mathsf{Verify}(\sigma, x^*, \pi^*) = 1 \land \exists w^I \in st, \mathsf{Identify}(\sigma, x^*, \pi^*, w^I) = 1 \end{bmatrix} \leq \mathsf{negl}(\lambda),$$

<sup>&</sup>lt;sup>7</sup> Different from the conventional simulation soundness, where the adversary is given simulated proofs, here we provide real proofs, which will be needed in applications.

where  $G \in \mathcal{G}$ , and  $\mathcal{O}_{P1}$ ,  $\mathcal{O}_{P2}$  are prover oracles specified in Fig.1. Hist denotes the query-response history of  $\mathcal{O}_{P1}$  and  $\mathcal{O}_{P2}$ , and "st" denotes the set of identifier witnesses generated by all calls (made by  $\mathcal{A}$ ) to  $\mathcal{O}_{P1}$ .

Remark 4. Recall that in the CRS-dependent sampler definition, we insist that the unpredictability holds for every CRS. One reason is that the unforgeability may not be achievable when unpredictability only holds for a randomly sampled CRS. Now we can give a concrete example. Assume L is an NP language and admits an unpredictable sampler  $G_L$ . We define an extended language L' that  $x' = (x, y) \in L'$  iff  $x \in L$ , and a sampler  $G_{L'}$  which on input a CRS  $\sigma$ , directly outputs  $(x, \pi)$ , where  $(x, w^I, w^{NI}) \leftarrow G_L(1^{\lambda})$  and  $\pi \leftarrow \text{Prove}(\sigma, x, w^I, w^{NI})$ . Given the entropic ZK of  $\pi$ , the identifier witness output by  $G_{L'}$  is unpredictable. However,  $\mathcal{A}$  can directly output  $\pi$  to break the unforgeability.

In some applications like non-malleable hash functions [9,3], it is required that an adversary cannot frame not only a target identifier witness, but also any identifier witness related to the target one. We formalize the security goal by the related-witness unforgeability below. Following the terminology developed in other non-malleable/related-key primitives, we capture that an identifier witness  $w^I$  is related to another  $w_0^I$  by a transformation  $\phi$ , namely  $w^I = \phi(w_0^I)$ . To make the definition more general, the admissible transformation set should be as large as possible.

**Definition 19 (Related-witness unforgeability).** Let  $\Pi$  be a waNIPS for  $(L, R_L^I)$ . We say  $\Pi$  satisfies the related-witness unforgeability w.r.t. PG,  $\mathcal{G}$  and a transformation set  $\Phi$ , if for any non-uniform PPT adversary  $\mathcal{A}$ , it hold that

$$\Pr \begin{bmatrix} pp \leftarrow \mathsf{PG}(1^{\lambda}); \sigma \leftarrow \mathsf{Setup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \overline{\sigma}); \\ (x^*, \pi^*, \phi) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma, pp) : (x^*, \pi^*) \notin \mathsf{Hist} \\ \wedge \mathsf{Verify}(\sigma, x^*, \pi^*) = 1 \land \exists w^I \in st, \mathsf{Identify}(\sigma, x^*, \pi^*, \phi(w^I)) = 1 \end{bmatrix} \leq \mathsf{negl}(\lambda),$$

where G belongs to  $\mathcal{G}$ ,  $\phi \in \Phi$ , and  $\mathcal{O}_{P1}$ ,  $\mathcal{O}_{P2}$  are prover oracles specified in Fig.1. Hist denotes the query-response history of  $\mathcal{O}_{P1}$ , and st denotes the set of identifier witnesses generated by all calls (made by  $\mathcal{A}$ ) to  $\mathcal{O}_{P1}$ .

Remark 5. Bounded root space and samplable root space. The related-witness unforgeability is defined w.r.t. a transformation set  $\Phi$  rather than any transformation  $\phi$  since there exists some relation such as constant transformations, for which this definition is hopeless. In other non-malleable primitives, many works [3,22] have been devoted to defining an admissible transformation  $\Phi$  set while keeping it as general as possible. In this work, we consider the most general transformation class that has been considered in the literature, that is, the so-called bounded root space (BRS) and samplable root space (SRS) developed by Chen *et al.* [22] and denoted by  $\Phi_{brs}^{srs}$ . More precisely, a transformation  $\phi$  has the  $p(\lambda)$ -BRS if  $|\phi^{-1}(0)| \leq p(\lambda)$ ;  $\phi$  has the SRS if we can efficiently sample an element from  $\phi^{-1}(0)$  uniformly at random. A transformation  $\Phi$  has the  $p(\lambda)$ -BRS (resp. SRS) if for every  $\phi \in \Phi$  and every constant  $c, \phi - x$  and  $\phi - c$  is  $p(\lambda)$ -BRS (resp. SRS).

**Identifier uniqueness.** Next, we discuss a special property of identifier uniqueness, (like unique signatures), which is useful when handling a case that the attacker may output a proof that will be identified by a string that is not even a witness. In certain applications (e.g., in our application of plaintext-checkable encryption), the attacker may try to fool the identify algorithm used by others.

Note that unforgeability does not address such an attack. The identifier uniqueness of a waNIPS says it is infeasible to produce a valid proof and two different identifier witnesses such that the proof is authenticated by both of them.

**Definition 20 (Identifier uniqueness).** We say a waNIPS  $\Pi$  for  $(L, R_L^I)$  satisfies the identifier uniqueness, if any non-uniform PPT  $\mathcal{A}$ , it holds that

$$\Pr \begin{bmatrix} \sigma \leftarrow \mathsf{Setup}(1^{\lambda}); (x, \pi, w_1^I, w_2^I) \leftarrow \mathcal{A}(\sigma) : \mathsf{Verify}(\sigma, x, \pi) = 1 \land \\ \mathsf{Identify}(\sigma, x, \pi, w_1^I) = 1 \land \mathsf{Identify}(\sigma, x, (\pi, w_2^I)) = 1 \end{bmatrix} \leq \mathsf{negl}(\lambda).$$

#### 3.4 Definitions with auxiliary inputs

All definitions built upon samplers can be further strengthened by allowing adversaries to obtain other auxiliary information (beyond the statements) about the identifier witness. This strengthening will be useful when applying waNIZKs to applications with auxiliary inputs (*e.g.*, in our applications of non-malleable hash and group signatures with verifier-local revocation). We formalize those by considering an enhanced sampler G, which outputs an auxiliary information z about  $w^I$  as well. Specifically, a k-unpredictable sampler G (w.r.t. PG and for  $(L, R_L^I)$ ) on inputs a CRS  $\sigma$  and PP outputs  $(X, W^I, W^{NI}, Z)$  that satisfies

$$\mathbf{H}^{\mathsf{unp}}(W^{I}_{\sigma}|X_{\sigma}, Z, PP) \ge k(\lambda).$$

Next, the auxiliary-input secure forms of the entropic ZK and the unforgeability (including the related-witness unforgeability) are almost identical to their original forms, except the oracle accesses to  $\mathcal{O}_{P1}$ ,  $\mathcal{O}_{P2}$ ,  $\mathcal{O}_{S1}$  and  $\mathcal{O}_{S2}$  are instantiated as in Fig.1 with boxed items. In the strengthened definitions, the prover oracle  $\mathcal{O}_{P1}$  and the simulation oracle  $\mathcal{O}_{S1}$  will also return the auxiliary input z for the sampled  $w^I$ .

On the one hand, the auxiliary-input entropic ZK and unforgeability clearly subsume the original definitions. On the other hand, considering auxiliary inputs does not seem to introduce any additional difficulty in constructing waNIZKs, since the statement itself is already an auxiliary information about the identifier witness. Thereafter, when we refer to entropic ZK and unforgeability, we mean the stronger auxiliary-input counterparts.

## 4 Constructing Witness-Authenticating NIZKs

In this section, we present our general constructions for waNIZKs.

**Basic challenges.** A folklore approach for adding a new property to NIZKs is to add some "tag" and extend the statement being proved <sup>8</sup>. For example, when transforming a NIZK to a knowledge-sound NIZK [41], one attaches the encryption of the witness to the proof, which enables the "extractability" by decrypting the ciphertext. Like this folklore, the main idea behind our constructions is also to attach an "identifiable" tag (and proof of validity) to a NIZK proof, *s.t.* it can be identified with the corresponding identifier witness. The challenge is that the tag has to satisfy several seemingly conflicting constraints.

<sup>&</sup>lt;sup>8</sup> This is a natural idea of constructing waNIZK, however, there may be more direct construction without the tag, and the resulting construction could be more compact. We leave this as an interesting question for future study.

- For "zero-knowledgeness". The tag should not leak any information about the identifier witness except the bit to a verifier knowing the corresponding  $w^I$ . Particularly, a tag generated from an unpredictable  $w^I$  should be "simulatable" (without using  $w^I$ ), even conditioned on the potential auxiliary information about  $w^I$ . Moreover, as the identifier witness may be used to prove multiple times, the "simulatability" shall be ensured across multiple tags from  $w^I$ .
- For soundness. First, we note that the tag generation should have a form of unforgeability. Namely, without the identifier witness  $w^{I}$ , a malicious prover cannot produce a tag that can be identified by  $w^{I}$  (even when it knows the statement). If we further want identifier uniqueness, it should be infeasible to find two identifier witnesses identifying one tag, which essentially requires a form of collision resistance. While for authenticating (knowledge) soundness, we will have to make sure the extracted witness is *exactly* the one used to generate the proof (comparing to the standard knowledge soundness, which only requires extracting *one* witness).

#### 4.1 Warm-up constructions

First, as a warm-up, we show how to easily build waNIZKs for distributions where the identifier witness is pseudorandom (conditioned on statements). This construction can be already useful in, e.g., group-oriented (*accountable*) authentications where users'secret keys can be pseudorandom. We present a very simple construction from a NIZK and a PRF. We then show how to easily lift this construction to be secure for unpredictable distributions that are *independent of the CRS* by using randomness extractors.

**PRF-based tag: a construction for pseudorandom identifier witnesses.** In many applications such as group-oriented anonymous authentication (*e.g.*, group signatures, ring signatures), the identifier witness is usually a secret key. In this case,  $w^{I}$  the identifier witness could be pseudorandom even conditioned on all public information (*e.g.*, a public key can be a commitment to the identifier). As a natural idea to generate a simulatable tag is to create a tag that is also pseudorandom, we use the witness as a key to generate the tag using a PRF, i.e.,

$$\mathsf{Tag}_{\mathsf{PRF}}(w^{I}) \to (t, \mathsf{PRF}(w^{I}, t)), \text{ for a random } t.$$

It is easy to verify that, when  $w^{I}$  is pseudorandom (with sufficient length and conditioned on all side information available to adversaries),  $\mathsf{PRF}(w^{I},t)$  is pseudorandom for any t. Thus the tag  $(t,\mathsf{PRF}(w^{I},t))$  is "simulatable" (for a random t) and unforgeable (for every t). Using such a tag generation mechanism, we can construct a simple waNIZK for a language L that admits a pseudorandom witness distribution. To identify whether the proof was generated by  $(w^{I}_{*},\star)$ , one just checks  $\mathsf{PRF}(w^{I}_{*},t) \stackrel{?}{=} \mathsf{PRF}(w^{I},t)$ . Moreover, by further requiring the PRF function to be collision-resistant, we can also achieve *identifier uniqueness*.

Formal construction and analysis are presented in Appendix A.

Lifting via randomness extraction: a construction for general CRS-independent distributions. The above approach cannot be applied to general unpredictable witness distributions. A natural idea is to transform an identifier witness into a uniform string. Randomness extractors [4] [32] are such a tool for generating a nearly uniform string from a random variable with enough entropy (called *source*), with the help of a short uniformly random string called *seed*. A computational extractor [32] would also be applicable even if the witness distribution is only computationally unpredictable. Several tricky issues remain: (1) For "zero-knowledgeness", the attacker may obtain multiple proofs generated using  $w^{I}$ . Since the seed is randomly chosen, the attacker essentially forces the same witness to be re-used with multiple different seeds and then the resulting outputs are used as the PRF keys; Thus we will require a reusable extractor [25,23] (or related-key secure PRF [1]). Unfortunately, there are only a few reusable (computational) extractor constructions, which either have entropy requirements on the source [23], or rely on non-standard assumptions [25]. In our setting, the witness distribution sometimes is only computationally unpredictable. The status of related-key secure PRF is neither promising as existing constructions only allow simple correlations.

(2) For *soundness*, a malicious prover may not generate the seed honestly. In this case, we won't have the properties of extractors, which could be devastating for unforgeability. To see this, let us view the inner product as the special Goldreich-Levin extractor, but the malicious prover will simply use *all-0* string as the seed. Now every witness can be used to identify such proof!

Luckily, since we are working in the CRS model, a first idea is that we could simply let the CRS include one uniform seed, and consider static notions (that the CRS is generated after the adversarial sampler, thus independent with the witness). The tag can be generated as follows:

$$\mathsf{Tag}_{\mathsf{Ext}-\mathsf{PRF}}(w^{I}) = (r, t, \mathsf{PRF}(\mathsf{Ext}(w^{I}, r), t)), \text{ for } r \text{ in CRS and a random } t$$

Leveraging this tag generation mechanism, we can have a construction for a language L that is secure w.r.t. k-unpredictable distributions.

Formal construction and analysis are also presented in Appendix A.

Unfortunately, once we do not have the luxury that the sampler is *independent* of the CRS, (e.g., when we consider the adaptive model), we will need new ideas to tackle those challenges.

#### 4.2 The full-fledged construction for CRS-dependent distributions

It is known that in general, a randomness extractor is secure only when the source is independent of the seed (otherwise, seeing the seed, there will always exist a source distribution that makes the first bit of the extractor output to be 1). Thus, the unpredictable statement distribution must be independent of the CRS in the above approach. However, in many applications, the statement (and the corresponding witness distribution) might be correlated with the CRS, e.g., all three applications we will present soon. It follows that we need a more general solution that can handle a CRS-dependent witness distribution.

A more flexible tag generation. It is not hard to see that for any tag generation function  $f(\text{params}, w) = \tau$ , if params is from CRS, the adversary can always find a witness distribution that depends on params such that the output  $\tau$  can be recognizable. However, in the above approach using extract-then-PRF, moving the seed out of CRS and letting prover generate it will put us back facing the challenges of malicious seed and reusability, as described above.

To circumvent such a dilemma, we first note that realizing simulatability and unforgeability does not have to be via pseudorandomness. For simulatability, another alternative is encryption primitives. For the ease of checking, we consider using deterministic public-key encryptions (DPKE) to generate a tag. <sup>9</sup> Regarding the unforgeability, we note it can be realized by adding a simulation-

<sup>&</sup>lt;sup>9</sup> Another potential tool could be perfectly one-way hash with auxiliary inputs [18]. Those are probabilistic functions that satisfy collision-resistance and hide all partial information about its input even under with auxiliary input. Unfortunately, such a strong primitive is only known to exist under a not-efficiently-falsifiable assumption [25]; thus, its existence is elusive. In fact, it even contradicts with a form of obfuscation [16]. We would like to have a construction that relies on standard assumptions.

extractable NIZK proof to the tag. As the NIZK is already a component of our waNIZK construction, we can make the tag unforgeable by enforcing the "collision resistance". Let DEnc be the encryption algorithm of a DPKE scheme, and we illustrate the tag generation mechanism below.

$$\mathsf{Tag}_{\mathsf{DPKE}}(w^{I}) \to (pk, \mathsf{DEnc}(pk, w^{I})), \text{ for a random public key } pk.$$
(1)

Next, we examine the previous challenges more closely.

- For "zero-knowledge", simulatability via pseudorandomness requires each output to be "independently" pseudorandom, thus requiring "reusability" in strong extractors. The latter is highly non-trivial as there is only a fixed amount of entropy available in the witness. While for ciphertext as output, however, we do not have to insist on a pseudorandom ciphertext distribution. Actually, "reusability" is trivial in standard public-key encryption schemes as each ciphertext is like an independent sample. Of course, in the setting of DPKE (when considering the multi-user security), things get more complicated as no private randomness is used for encryption; we also need to consider the auxiliary input of the witness. Fortunately, Brakerski and Segev's DLIN based construction [13] can satisfy auxiliary-input, and multi-user security simultaneously.
- For soundness, it was difficult to deal with malicious (prover-generated) seeds in the extractor setting, as there is no way to prove a seed is sampled uniformly. Nevertheless, if the parameters have some algebraic structure or functional properties, we may be able to enforce those features (for unforgeability) instead of proving distributional properties. For example, the decryptability condition (correctness) is such a property, when using encryption. In more detail, a malicious prover may still want to choose a malformed pk, but now we can ask the prover to attach a proof of well-formedness of pk, simply attesting there exists a secret key. The perfect correctness of encryption requires that for every valid key pair pk, sk, and every message m, Dec(sk, DEnc(pk, m)) = m. This automatically implies that the encryption function DEnc for each valid pk defines an injective function, i.e., for any  $w_1 \neq w_2$ ,  $DEnc(pk, w_1) \neq DEnc(pk, w_2)$ . Moreover, it is indeed the case for the DPKE instantiation we chose in [13]. In this way, a malicious prover cannot evade the checking or frame other witness holders!

The construction. Let us firstly specify the building blocks we will use.

- A deterministic public-key encryption (DPKE) scheme  $\Sigma_{de} = \{K_{de}, E_{de}, D_{de}\}$ . We assume w.l.o.g. that the plaintext space contains all identifier witnesses of L. Particularly, we require the DPKE to be perfectly correct and PRIV-IND-MU-secure with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs which captures the security when one message is encrypted under multiple keys and the auxiliary input about the message are available to adversaries. We assume w.l.o.g. that there is a relation  $R_{L_{de}}$  s.t. a key pair (pk, sk) is valid iff  $R_{L_{de}}(pk, sk) = 1$ .
- A NIZK proof system  $\varPi_{zk} = \{S_{zk}, \mathsf{P}_{zk}, \mathsf{V}_{zk}\}$  for an NP language

$$L_{\mathsf{CD}} := \{ (x, pk, c); (w^{I}, w^{NI}, sk) : \\ (w^{I}, w^{NI}) \in R_{L}(x) \land w^{I} \in R_{L}^{I}(x) \land c = \mathsf{E}_{\mathsf{de}}(pk, w^{I}) \land R_{L_{\mathsf{de}}}(pk, sk) = 1 \};$$
(2)

The full-fledged construction  $\Pi_{CD} = \{\text{Setup}, \text{Prove}, \text{Verify}, \text{Identify}\}\$  for an NP language L with identifier relation  $R_L^I$  is presented in Fig.2.

Security analysis. The completeness directly follows the completeness of the underlying NIZK proof system  $\Pi_{zk}$  and of the DKPE scheme  $\Sigma_{de}$ . Particularly, under an honest pk,  $c = \mathsf{E}_{de}(pk, w^I)$  uniquely determines  $w^I$  and thus the proof will not be mis-identified by another identifier witness.

$Setup(1^{\lambda})$	
$\sigma_{zk} \leftarrow S_{zk}(1^{\lambda}) \qquad /\!\!/ \text{ generate a CRS of the underlying NIZK}$	
$\mathbf{return}  \sigma = \sigma_{zk}$	
$Prove(\sigma, x, (w^I, w^{NI}))$	
$(pk, sk) \leftarrow K_{de}(1^{\lambda}) \qquad /\!\!/ \text{ generate the public key and the secret key}$	
$c \leftarrow E_{de}(pk, w^{I}) \qquad /\!\!/ \text{ encrypt the identifier witness under pk}$	
$\pi_{zk} \leftarrow P_{zk}(\sigma_{zk}, (x, pk, c), (w^{I}, w^{NI}, sk))$	
$/\!\!/ \text{ prove } x \in L \wedge (pk,c) \text{ are well-formed}$	
$\mathbf{return} \ \pi = (pk, c, \pi_{zk})$	
$Verify(\sigma,x,\pi)$	
$b \leftarrow V_{zk}(\sigma_{zk},(x,pk,c),\pi_{zk}) \qquad /\!\!/ \text{ check the validity of the proof } \pi_{zk}$	
return b	
$\boxed{Identify(\sigma, x, \pi, w^I)}$	
$c' \leftarrow E_{de}(pk, w^I)$ // encrypt the identifier witness under the public key	
if $(c = c')$ then return 1 else return 0	

Fig. 2. The full-fledged construction.

We claim the security of  $\Pi_{CD}$  in the following theorem and present here only a security sketch: the statement being proved by  $\Pi_{zk}$  is formed by (x, pk, c). (1) the knowledge soundness of  $\Pi_{zk}$ ensures one can extract a  $w^I$  and  $c = \mathsf{E}_{\mathsf{de}}(pk, w^I)$ . By the description of Identify, these together imply the authenticating knowledge soundness. (2) When  $\Pi_{zk}$  is sound, the public key pk contained in a valid proof should be a valid public key, and thus (pk, c) determines a unique plaintext (as the identifier), which ensures the identifier uniqueness. (3) Moreover, as the DPKE is PRIV-IND-MUsecure with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs, entropic ZKcan be achieved.

Regarding the unforgeability, at a high level, we show a contradication that a successful adversary  $\mathcal{A}$  against this property will give rise to a successful adversary  $\mathcal{B}$  that could recover messages from DPKE ciphertexts. Specifically, since  $\Pi_{zk}$  is a simulation-extractable NIZK,  $\mathcal{B}$  can answer all prover oracle queries via a "hybrid" prover algorithm which returns a "proof" formed by  $(pk, c, \pi_{zk})$ where (pk, c) is an honest encryption of the identifier witness while  $\pi_{zk}$  is a simulated proof. Note that  $\mathcal{A}$  cannot distinguish the real prover oracle and the hybrid prover oracle. Next,  $\mathcal{A}$  will issue a challenge proof  $(pk^*, c^*, \pi_{zk}^*)$ , for a challenge statement  $x^*$ , satisfying  $c^* = \mathsf{E}_{\mathsf{de}}(pk^*, w^I)$ , and  $\mathcal{B}$ can further leverage the knowledge extractor of  $\Pi_{zk}$  to extract  $w^I$ , which is the plaintext of these deterministic encryptions, from  $\pi_{zk}$ .

Actually, our construction can satisfy the stronger related-witness unforgeability. Specifically, in the definition, a successful adversary will output  $(x^*, \pi^*, \phi^*)$  such that  $\pi^* = (pk^*, c^*, \pi_{zk}^*)$  is authenticated by  $\phi^*(w^I)$ , where  $\phi^*$  is a transformation that all preimages can be efficiently found (as formalized by Chen et al.[22] and recalled in Remark 5). Note that the adversary  $\mathcal{B}$  can still leverage the attacker to recover messaages from DPKE encryptions: all queries to the prover oracle can be simulated as before; after extracting  $\phi(w^I)$  from the challenge proof  $\pi^*$ ,  $\mathcal{B}$  just outputs one preimage of  $\phi^*(w^I)$  which will be equal to  $w^I$  with a non-negligible probability.

**Theorem 1.** Let  $\Pi_{CD}$  be the construction in Fig.2, and the following results hold:

- $\Pi_{CD}$  satisfies the authenticating (knowledge) soundness, if  $\Pi_{zk}$  satisfies the (knowledge) soundness;
- $\Pi_{CD}$  satisfies the identifier uniqueness, if  $\Pi_{zk}$  is sound, and the DPKE satisfies perfect correctness;
- $\Pi_{CD}$  satisfies the entropic ZK w.r.t. all k-unpredictable samplers and the transformation set  $\Phi_{\text{brs}}^{\text{srs}}$ , if  $\Pi_{zk}$  is zero-knowledge, and  $\Sigma_{de}$  is PRIV- IND-MU-secure with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs.<sup>10</sup>
- $\Pi_{CD}$  satisfies the related-witness unforgeability w.r.t. all k-unpredictable samplers and , if  $\Pi_{zk}$  is a simulation-extractable NIZK, and  $\Sigma_{de}$  is PRIV- IND-MU-secure with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs.

The formal proof is deferred to Sect.4.3. We first discuss how to instantiate the full-fledged construction.

Sketch of instantiation. Since the underlying DPKE scheme  $\Sigma_{de}$  shall satisfy the perfect correctness and the PRIV-IND-MU-security with respect to hard-to-invert auxiliary inputs, the only candidate so far is Brakerski and Segev's *d*-linear based construction [13]. Particularly, this construction allows  $2^{-k}$ -hard-to-invert auxiliary inputs where  $2^{-k} \leq \frac{\nu(\lambda)}{q^{2d}}$ . Here,  $\nu$  is a negligible function in  $\lambda$ , *d* can be 1 when considering the DDH assumption, and *q* is the order of the DDH group which is usually  $2^{\Theta(\lambda)}$ . Accordingly, if we set  $\nu(\lambda) = 2^{-\omega(\log \lambda)}$ , the admissible samplers of our waNIZK construction  $\Pi_{CD}$  should be *k*-unpredictable for some  $k \geq 2 \log q + \omega(\log \lambda)$ .<sup>11</sup> Regarding the underlying simulation-extractable NIZK  $\Pi_{zk}$  for  $L_{CD}$ , we note it could be realized via simulation-extractable NIZKs for general NP languages. Particularly, adaptive NIZKs for general NP languages are known to exist under the RSA assumption [27] or the LWE assumption [40], and we can add simulation extractability to them using standard tools including one-way functions and public-key encryptions as noted in [41]. In addition, since the tag generation procedure is algebraic (and Groth-Sahai-friendly), we can leverage the (simulation-extractable) Groth-Sahai proof system [31] to instantiate  $\Pi_{zk}$ , if the statement  $x \in L$  that we wish to prove is also Groth-Sahai-friendly.

#### 4.3 Proof of Theorem 1

**Preparation.** To facilitate our security proof, we show a DPKE scheme, which is PRIV-IND-MUsecure with respect to  $\epsilon$ -hard-to-invert auxiliary inputs, also applies to messages from a source with *k*-unpredictability entropy where  $k(\lambda) = -\log \epsilon(\lambda)$ . More precisely,

**Corollary 1.** Let  $\Sigma$  be PRIV-IND-MU-secure with respect to  $\epsilon$ -hard-to-invert auxiliary inputs, and let  $(M_{\lambda}, AUX_{\lambda})$  be a source satisfying  $\mathbf{H}^{\mathsf{unp}}(M_{\lambda}|AUX_{\lambda}) \geq -\log \epsilon(\lambda)$ . For any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$|\Pr[\mathsf{Exp}_{\varSigma,\mathcal{A},(\mathcal{M},AUX)}^{\mathsf{priv},0}(\lambda)=1] - \Pr[\mathsf{Exp}_{\varSigma,\mathcal{A},(\mathcal{M},AUX)}^{\mathsf{priv},1}(\lambda)=1]| \le \mathsf{negl}(\lambda)$$

The experiments are defined as follows, where L is a polynomial-bounded integer function.

<sup>&</sup>lt;sup>10</sup> A basic requirement is  $k = \omega(\log \lambda)$  s.t. it is possible to have such a DPKE scheme.

<sup>&</sup>lt;sup>11</sup> In certain applications, we may be insterested in the relation between  $n = |w^I|$  and k of admissible samplers. Note that for any constant  $0 < \mu \leq 1$ , there exists a sufficient large polynomial n such that  $n^{\mu} \geq 2\log q + \omega(\log \lambda)$ . Namely, k can be sublinear in n, and in this case given (X, Z, PP) finding  $W^I$  is sub-exponentially hard.

$Exp^{priv,0}_{\varSigma,\mathcal{A},(\mathcal{M},AUX)}(\lambda)$	$Exp^{priv,1}_{\varSigma,\mathcal{A},(\mathcal{M},AUX)}(\lambda)$
$(m_0, aux) \leftarrow (\mathcal{M}_\lambda, AUX_\lambda)$	$(m_0, aux) \leftarrow (\mathcal{M}_\lambda, AUX_\lambda)$
for $i \in [L(\lambda)]$	for $i \in [L(\lambda)]$
$(pk_i, sk_i) \leftarrow KeyGen(1^\lambda)$	$m_i \leftarrow \{0,1\}^{ m_0 }$
$c_i \leftarrow Enc(pk_i, m_0)$	$(pk_i, sk_i) \leftarrow KeyGen(1^{\lambda})$
$b \leftarrow \mathcal{A}(\{(pk_i, c_i)\}_{i \in [\ell(\lambda)]}, aux)$	$c_i \leftarrow Enc(pk_i, m_i)$
	$b \leftarrow \mathcal{A}(\{(pk_i, c_i)\}_{i \in [\ell(\lambda)]}, aux)$

*Proof.* By the definition of the unpredictability entropy, for  $(M_{\lambda}, AUX_{\lambda})$  satisfying  $\mathbf{H}^{\mathsf{unp}}(M_{\lambda}|AUX_{\lambda}) \geq$  $-\log \epsilon(\lambda)$ , there is a collection of distributions  $Y|_{aux}$  such that  $(M_{\lambda}, AUX_{\lambda})$  and  $(Y_{\lambda}, AUX_{\lambda})$  are computational indistinguishable, and given aux finding y is  $\epsilon$ -hard. Let the function f be sampling procedure for AUX conditioned on y. Notice the DPKE scheme essentially does not require an efficiently computable function. Therefore,  $(Y_{\lambda}, f)$  is an admissible message distribution along with an auxiliary input function.

From the indistinguishability,  $(\overrightarrow{pk}, \overrightarrow{\mathsf{Enc}}_{\overrightarrow{pk}}(m_0), aux)$  and  $(\overrightarrow{pk}, \overrightarrow{\mathsf{Enc}}_{\overrightarrow{pk}}(y), aux)$  are computationally

indistinguishable, where  $\overrightarrow{pk} = (pk_1, \dots, pk_{L(\lambda)})$  and  $\overrightarrow{\mathsf{Enc}}_{\overrightarrow{pk}}(m)$ =  $(\mathsf{Enc}_{pk_1}(m), \dots, \mathsf{Enc}_{pk_{L(\lambda)}}(m))$ . Then, since finding y is  $\epsilon$ -hard, by the definition of DPKE,  $(\overrightarrow{pk}, \overrightarrow{\mathsf{Enc}}_{\overrightarrow{pk}}(y), aux)$  and  $(\overrightarrow{pk}, (\mathsf{Enc}_{pk_1}(m_1), \dots, \mathsf{Enc}_{pk_{L(\lambda)}}(m_{L(\lambda)})), aux)$  are computationally indistinguishable, where  $m_1, \dots, m_{L(\lambda)} \leftarrow \{0, 1\}^{|m_0|}$ . Thus we have the result. 

Finally, for our sepcial purpose, we show the above result can be extended to the following form.

**Corollary 2.** Let L and M be two polynomial-bounded integer functions in  $\lambda$ , and  $\Sigma$  be PRIV-IND-MU-secure with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs. Let PG be a trusted parameter generation procedure (cf.Def,13). Then, for any non-uniform PPT adversary  $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ , it follows that

$$|\Pr[\mathsf{Exp}_{\varSigma,\mathcal{A},\mathsf{PG}}^{\mathsf{priv},0}(\lambda)=1] - \Pr[\mathsf{Exp}_{\varSigma,\mathcal{A},\mathsf{PG}}^{\mathsf{priv},1}(\lambda)=1]| \le \mathsf{negl}(\lambda).$$

The experiments are defined in the following, where  $\mathcal{A}_1$  is only allowed to output a sampler G which is k-unpredictable w.r.t. PG.

$Exp^{priv,0}_{\varSigma,\mathcal{A},PG}(\lambda)$	$Exp^{priv,1}_{\varSigma,\mathcal{A},G,PG}(\lambda)$
$\boxed{pp \leftarrow PG(\lambda)}$	$pp \leftarrow PG(\lambda)$
$(G, st) \leftarrow \mathcal{A}_1(pp)$	$(G, st) \leftarrow \mathcal{A}_1(pp)$
for $i \in [M(\lambda)]$	for $i \in [M(\lambda)]$
$(x_i, w_i^I, w_i^{NI}, z_i) \leftarrow G(pp)$	$(x_i, w_i^I, w_i^{NI}, z_i) \leftarrow G(pp)$
$\eta_i = (x_i, z_i, pp)$	$\eta_i = (x_i, z_i, pp)$
for $j \in [L(\lambda)]$	for $j \in [L(\lambda)]$
$(pk_{i,j}, sk_{i,j}) \leftarrow KeyGen(1^{\lambda})$	$m_{i,j} \leftarrow \$ \{0,1\}^{ w_i^I }$
$c_{i,j} \leftarrow Enc(pk_{i,j}, w_i^I)$	$(pk_{i,j}, sk_{i,j}) \leftarrow KeyGen(1^{\lambda})$
$b \leftarrow \mathcal{A}_2(\{(pk_{i,j}, c_{i,j}), \eta_i\}_{i \in [L], j \in [M]}, st)$	$c_{i,j} \leftarrow Enc(pk_{i,j}, m_{i,j})$
	$b \leftarrow \mathcal{A}_2(\{(pk_{i,j}, c_{i,j}), \eta_i\}_{i \in [L], j \in [M]}, st)$

*Proof.* Let us consider a sequence of (M+1) hybrid experiments  $\{\mathsf{Exp}_{\Sigma,\mathcal{A},\mathsf{PG}}^{\mathsf{hybrid}-i}\}_{i\in[M+1]}$ . The first one is identical to  $\mathsf{Exp}_{\Sigma,\mathcal{A},\mathsf{PG}}^{\mathsf{priv},0}$  and the (M+1)-th one is identical to  $\mathsf{Exp}_{\Sigma,\mathcal{A},\mathsf{G},\mathsf{PG}}^{\mathsf{priv},1}$ . In the *T*-th one, for every i < T and  $j \in [L]$ ,  $c_{i,j}$  is an encryption of a randomly sampled  $m_{i,j}$ ; for every  $i \geq T$  and  $j \in [L]$ ,  $c_{i,j}$  is an encryption of the *i*-th identifier witness  $w_i^I$ . Then, by standard argument, if there exists an efficient adversary  $\mathcal{A}$  making  $|\Pr[\mathsf{Exp}_{\Sigma,\mathcal{A},G,\mathsf{PG}}^{\mathsf{priv},0}(\lambda) = 1] - \Pr[\mathsf{Exp}_{\Sigma,\mathcal{A},G,\mathsf{PG}}^{\mathsf{priv},1}(\lambda) = 1]| > \mathsf{negl}(\lambda)$ , there must exist an index  $T^* \in [M]$ , such that there is an efficient adversary  $\mathcal{A}^{T^*}$  making

$$|\Pr[\mathsf{Exp}_{\varSigma,\mathcal{A}^{T^*},G,\mathsf{PG}}^{\mathsf{hybrid}-T^*}(\lambda)=1] - \Pr[\mathsf{Exp}_{\varSigma,\mathcal{A}^{T^*},G,\mathsf{PG}}^{\mathsf{hybrid}-(T^*+1)}(\lambda)=1]| > \mathsf{negl}(\lambda).$$

Then, it is easy to construct an efficient adversary  $\mathcal{B}$  to break the deterministic encryption  $\Sigma$ . Let us specify a message distribution  $(M^*, AUX^*)$ , where each sample (m, aux) is generated as follows:  $pp \leftarrow \mathsf{PG}(\lambda), (G, st) \leftarrow \mathcal{A}_1^{T^*}(pp), (x, w^I, w^{NI}, z) \leftarrow G(pp), m = w^I, aux = (x, z, pp, st, G)$ . By definition,  $(M^*, AUX^*)$  is an admissible message distribution.

Next,  $\mathcal{B}$  can issue the distribution  $(\mathcal{M}^*_{\lambda}, AUX^*_{\lambda})$  and obtain  $(\{pk^*_j, c^*_j\}_{j \in [L]}, x^*, z^*, pp, st, G)$ . Then,  $\mathcal{B}$  sets  $(pk^*_{(T^*+1),j}, c_{(T^*+1),j}) = (pk^*_j, c^*_j)$  for all  $j \in [L]$  and  $\eta_{T^*+1} = (x^*, z^*, pp)$ . For all  $i \neq (T^*+1)$ ,  $\mathcal{B}$  executes as follows.

 $\begin{aligned} &-\text{ If } i < T^*, \text{ for all } j \in [L], \text{ run } (pk_{i,j}, sk_{i,j}) \leftarrow \mathsf{KeyGen}(1^{\lambda}), m_{i,j} \leftarrow \$\{0,1\}^{|w_I|}, c_{i,j} \leftarrow \mathsf{Enc}(pk_{i,j}, m_{i,j}). \\ &-\text{ If } i \geq T^* + 1, \text{ run } (x_i, w_i^I, w_i^{NI}, z_i) \leftarrow G(pp), \text{ and for all } j \in [L], \text{ run } (pk_{i,j}, sk_{i,j}) \leftarrow \mathsf{KeyGen}(1^{\lambda}), \\ &c_{i,j} \leftarrow \mathsf{Enc}(pk_{i,j}, w_i^I). \end{aligned}$ 

Finally,  $\mathcal{B}$  returns all these public keys, ciphertexts and auxiliary inputs to  $\mathcal{A}_2^{T^*}$ . It is easy to see when in the received tuple  $(\{pk_j^*, c_j^*\}_{j \in [L]}, x^*, z^*, pp)$ , each  $c_j^*$  is an encryption of  $w^I$ , the environment around  $\mathcal{A}^{T^*}$  is identical to that in the  $T^*$ -th experiment; otherwise, the environment is identical to that in the  $T^* + 1$ -th experiment. Therefore,  $\mathcal{B}$  can leverage  $\mathcal{A}^{T^*}$  to break the PRIV-IND-MUsecurity with respect to  $2^{-k}$ -hard-to-invert auxiliary inputs of  $\Sigma$ .

Then we proceed the proof of Theorem 1. We first prove the entropic ZK of our construction  $\Pi_{CD}$ , and then prove the soundness definitions.

**Proof of entropic ZK.** We first present the simulator (SimSetup, SimProve) of  $\Pi_{CD}$ . Let (SS<sub>zk</sub>, SP<sub>zk</sub>) be the simulator of the underlying NIZK proof system  $\Pi_{zk}$ , and then we construct (SimSetup, SimProve) for  $\Pi_{CD}$  as follows.

1. SimSetup $(1^{\lambda})$ . Firstly invoke the simulator of  $\Pi_{zk}$ :  $(\sigma_{zk}, \tau_{zk}) \leftarrow SS_{zk}(1^{\lambda})$ . Return the CRS  $\sigma = \sigma_{zk}$  and the trapdoor  $\tau = \tau_{zk}$ . 2. SimProve $(\sigma, \tau, x)$ . Firstly generate a pair of keys  $(pk, sk) \leftarrow K_{de}(1^{\lambda})$ , and sample  $m \leftarrow \infty$ 

 $\{0,1\}^{|w^I|}$ ; Compute  $c = \mathsf{E}_{\mathsf{de}}(pk,m)$ ; Then invoke the simulator of  $\Pi_{zk}$  to prove  $(x, pk, c) \in L_{CD}$ :  $\pi_{\mathsf{zk}} \leftarrow \mathsf{SP}_{\mathsf{zk}}(\sigma_{\mathsf{zk}}, \tau_{\mathsf{zk}}, (x, pk, c))$ ; Return the simulated proof  $\pi = (pk, c, \pi_{\mathsf{zk}})$ .

We denote the event

$$[pp \leftarrow \mathsf{PG}(1^{\lambda}); \sigma \leftarrow \mathsf{Setup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma)]$$

by  $\mathtt{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{real}},$  and denote the event

 $[pp \leftarrow \mathsf{PG}(1^{\lambda}); (\sigma, \tau) \leftarrow \mathsf{SimSetup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{S1}}, \mathcal{O}_{\mathsf{S2}}}(\sigma)]$ 

by Event<sup>sim</sup><sub>zk,A</sub>. Our goal is to show the difference between  $\Pr[\text{Event}^{\text{real}}_{zk,A}]$  and  $\Pr[\text{Event}^{\text{sim}}_{zk,A}]$  is negligible in  $\lambda$ , for any non-uniform PPT adversary A. To do this, we define hybrid events  $\text{Event}^1_{zk,A}$  and  $\text{Event}^2_{zk,A}$ .

# The hybrid event $Event^1_{zk,A}$ .

 $\overline{[pp \leftarrow \mathsf{PG}(1^{\lambda}); (\sigma, \tau) \leftarrow \mathsf{SimSetup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma) : 1 \leftarrow \mathcal{A}^{\mathsf{OP1}_1, \mathsf{OP1}_2}(\sigma)]}$ , where the oracle  $\mathsf{OP1}_1, \mathsf{OP1}_2$  are defined by modifying  $\mathcal{O}_{\mathsf{P1}}, \mathcal{O}_{\mathsf{P2}}$ . Precisely,

$OP1_1(\sigma, \tau, pp)$	$OP1_2(\sigma, \tau, pp, x_i, EG, st)$
$\overline{i++;}$	$\overline{\text{Find}(i, x_i, (w_i^I, w_i^{NI}), z_i) \in st}$
$(x_i, (w_i^I, w_i^{NI}), z_i) \leftarrow G(pp);$	$(\bar{x}, \bar{w}^{NI}) \leftarrow EG_{real}(pp, \sigma, x_i, (w_i^I, w_i^{NI}))$
$st \leftarrow st \cup (i, x_i, (w_i^I, w_i^{NI}), z_i);$	$\bar{\pi} \leftarrow \boxed{Prove^{(1)}(\sigma, \tau, \bar{x}, w_i^I)}$
$\pi_i \leftarrow Prove^{(1)}(\sigma, \tau, x_i, w_i^I)$	$\mathbf{return}\ (\bar{x},\bar{\pi},z_i)$
$\mathbf{return}\ (x_i,\pi_i,z_i)$	

The boxed items are different from the prover oracle  $\mathcal{O}_{\mathsf{P}}$ . Here, the hybrid prover algorithm  $\mathsf{Prove}^{(1)}(\sigma, \tau, x, (w^I, w^{NI}))$  is executed as follows, where the boxed items are different from the prover algorithm **Prove**.

Prove<sup>(1)</sup> $(\sigma, \tau, x, w^{I})$ . 1. Parse  $\sigma = \sigma_{zk}$ , and  $\tau = \tau_{zk}$ ; 2. Generate  $(pk, sk) \leftarrow \mathsf{K}_{\mathsf{de}}(1^{\lambda})$ , and encrypt  $w^{I} : c \leftarrow \mathsf{E}_{\mathsf{de}}(pk, w^{I})$ ; 3. Prove  $(x, pk, c) \in L_{CD}$  by using the  $\Pi_{zk} : \pi_{zk} \leftarrow \mathsf{SP}_{zk}(\sigma_{zk}, \tau_{zk}, (x, pk, c))$ . 5. Output the final proof  $\pi = (pk, c, \pi_{zk})$ .

The hybrid event  $\text{Event}_{zk,\mathcal{A}}^2$ . This event is almost identical to  $\text{Event}_{zk,\mathcal{A}}^1$ , except that the oracle  $\overline{\text{OP1}_2}$  is replaced with  $\text{OP2}_2$ .

 $\begin{array}{l} \displaystyle \frac{\mathsf{OP2}_2(\sigma,\tau,pp,x_i,\mathsf{EG},st)}{\mathrm{Find}(i,x_i,(w_i^I,w_i^{NI}),z_i)\in st} \\ \displaystyle (\bar{x},\bar{w}^{NI})\leftarrow\mathsf{EG}_{\mathtt{sim}}(pp,\sigma,x_i) \\ \displaystyle \bar{\pi}\leftarrow \boxed{\mathsf{Prove}^{(1)}(\sigma,\tau,\bar{x},w_i^I)} \\ \displaystyle \mathbf{return}\ (\bar{x},\bar{\pi},z_i) \end{array}$ 

We have the following lemmas.

**Lemma 1.** Assume that any non-uniform PPT adversary cannot break the adaptively unbounded zero-knowledge property of  $\Pi_{zk}$  with an advantage greater than  $\mathsf{Adv}_{\Pi_{zk}}^{zk}(\lambda)$ . It follows that

$$|\Pr[\textit{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{real}}] - \Pr[\textit{Event}_{\mathsf{zk},\mathcal{A}}^1]| \leq \mathsf{Adv}_{\varPi_{\mathsf{zk}}}^{\mathsf{zk}}(\lambda),$$

for any non-uniform PPT adversary A.

*Proof (sketch).* Assume there is an adversary  $\mathcal{A}$  making

$$|\Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{real}}] - \Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^{1}]| = \epsilon.$$

Then, we can construct an efficient adversary  $\mathcal{B}$  breaking the adaptively unbounded zero-knowledge of  $\Pi_{zk}$  with an advantage  $\epsilon$ , by leveraging  $\mathcal{A}$ .  $\mathcal{B}$  is given a CRS  $\sigma_{zk}$  and a prover oracle  $\mathcal{O}_{zk}$ , which are either an honestly generated CRS and a real prover oracle, or a simulated CRS and a simulation prover oracle. Then,  $\mathcal{B}$  mimics all steps in  $\texttt{Event}_{zk}^1$  for  $\mathcal{A}$ :  $\mathcal{B}$  provides  $\sigma_{zk}$  to  $\mathcal{A}$ , and answers queries to  $\mathsf{OP1}_1, \mathsf{OP1}_2$  by querying  $\mathcal{O}_{zk}$  whenever it needs to execute the algorithm  $\mathsf{SP}_{zk}$ . It is easy to see that, when  $\sigma_{zk}$  is a simulated CRS and  $\mathcal{O}_{zk}$  is a simulation oracle, at the point of  $\mathcal{A}$ 's view, all steps are identical to that in  $\texttt{Event}_{zk}^1$ ; when  $\sigma_{zk}$  and  $\mathcal{O}_{zk}$  are real, these steps are identical to that in  $\texttt{Event}_{zk}^{\mathsf{real}}$ . Therefore, it must follow that  $\epsilon \leq \mathsf{Adv}_{\Pi_{zk}}^{\mathsf{zk}}$ .

**Lemma 2.** Under the requirement that every EG is a dual-mode extended sampler, it holds that for any non-uniform adversary A,

$$|\Pr[\textit{Event}_{\mathsf{zk},\mathcal{A}}^1] - \Pr[\textit{Event}_{\mathsf{zk},\mathcal{A}}^2]| \le \mathsf{negl}(\lambda).$$

*Proof.* Consider a sequence of experiments  $\{\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{1,i}\}_{i\in[K]}$  for some polynomial K that bounds the number of extended samplers the adversary  $\mathcal{A}$  issues. In  $\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{1,i}$  the oracle answering the query of extended samplers is denoted by  $\mathsf{OP}i_2$ , and it answers j-the query either using  $\mathsf{OP}2_2$  if j < i or using  $\mathsf{OP}1_2$  if  $j \geq i$ . So, it holds that  $\texttt{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{1,i} = \texttt{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^1$  and  $\texttt{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{1,K} = \texttt{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^2$ .

By standard arguments, if there is a non-uniform PPT adversary  $\mathcal{A} \ s.t. |\Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^1] - \Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^2]| > \mathsf{negl}(\lambda)$ , there will be  $i^* \in [1, K-1] \ s.t. |\Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*}] - \Pr[\texttt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*+1}]| > \mathsf{negl}(\lambda)$  for this adversary.

We show that this result implies that the  $i^*$ -th sampler  $\mathsf{EG}^{i^*}$  is not a dual-mode sampler w.r.t. PG, which contradicts our requirement. Specifically, we consider an adversary  $\mathcal{B}$  against  $\mathsf{EG}^{i^*}$  w.r.t. PG, who is given  $pp \leftarrow \mathsf{PG}(1^{\lambda})$ , an an oracle access to  $\mathcal{O}_{\mathsf{EG}}$  that either returns the statement outputted by  $\mathsf{EG}^{i^*}_{\mathsf{real}}$  or that outputted by  $\mathsf{EG}^{i^*}_{\mathsf{sim}}$ .

$$\begin{split} & \frac{\mathcal{B}(pp)}{(\sigma,\tau) \leftarrow \mathsf{SimSetup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp,\sigma); b \leftarrow \mathcal{A}^{(\cdot)}(\sigma,pp); \mathbf{return} \ b} \\ & \frac{\mathsf{Answering} \ \mathcal{A}}{\mathsf{All queries are answered as in } \mathbf{Event}_{\mathsf{zk},\mathcal{A}}^1, \operatorname{except}(\mathsf{EG}^{i^*}, x_i)} \\ & \mathsf{Query} \ \mathcal{O}_{\mathsf{EG}} \ \text{with} \ (pp, x_i, w_i^I, w^{NI}) \ \text{and obtain} \ \bar{x} \\ & \mathsf{Run} \ \bar{\pi} \leftarrow \mathsf{Prove}^{(1)}(\sigma, \tau, \bar{x}, w_i^I) \ \text{and Return} \ (\bar{x}, \bar{\pi}, z_i) \end{split}$$

When  $\mathcal{O}_{\mathsf{EG}}$  returns the statement outputted by the **real** mode, at  $\mathcal{A}$ 's view the environment is identical to that in  $\mathtt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*+1}$ ; otherwise, the environment is identical to that in  $\mathtt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*+1}$ . Therefore, if there exists an adversary making  $|\Pr[\mathtt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*}] - \Pr[\mathtt{Event}_{\mathsf{zk},\mathcal{A}}^{1,i^*+1}]| > \mathsf{negl}(\lambda)$ ,  $\mathcal{B}$  has non-negligible advantage in distinguishing the statements outputed by  $\mathsf{EG}_{\mathsf{real}}^{i^*}$  and  $\mathsf{EG}_{\mathsf{sim}}^{i^*}$ .

**Lemma 3.** Assume that any non-uniform PPT adversary  $\mathcal{A}$  cannot break the PRIV-IND-MUsecurity (w.r.t.  $2^{-k}$ -hard-to-invert auxiliary inputs) of the DPKE scheme  $\Sigma_{de}$  with an advantage greater than  $\operatorname{Adv}_{\Sigma_{de}}^{\operatorname{priv}}(\lambda)$ . Or more precisely (cf. Corollary.2),

$$|\Pr[\mathsf{Exp}_{\Sigma_{\mathsf{de}},\mathcal{A},\mathsf{PG}}^{\mathsf{priv},0}(\lambda)=1] - \Pr[\mathsf{Exp}_{\Sigma_{\mathsf{de}},\mathcal{A},\mathsf{PG}}^{\mathsf{priv},1}(\lambda)=1]| \le \mathsf{Adv}_{\Sigma_{\mathsf{de}}}^{\mathsf{priv}}(\lambda).$$

It follows that

$$|\Pr[\textit{Event}_{\mathsf{zk}}^{\mathsf{sim}}] - \Pr[\textit{Event}_{\mathsf{zk}}^2]| \leq \mathsf{Adv}_{\varSigma_{\mathsf{de}}}^{\mathsf{priv}},$$

for any non-uniform PPT adversary A.

*Proof.* Assuming there is an adversary  $\mathcal{A}$  s.t.  $|\Pr[\texttt{Event}_{\mathsf{zk}}^{\mathsf{sim}}] - \Pr[\texttt{Event}_{\mathsf{zk}}^2]| = \epsilon$ , we can construct an adversary  $\mathcal{B}$  making  $|\Pr[\mathsf{Exp}_{\Sigma_{\mathsf{de}},\mathcal{B},\mathsf{PG}}^{\mathsf{priv},0}(\lambda) = 1] - \Pr[\mathsf{Exp}_{\Sigma_{\mathsf{de}},\mathcal{B},\mathsf{PG}}^{\mathsf{priv},1}(\lambda) = 1]| = \epsilon$ . More precisely, we write the code of  $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$  in the following.

$\mathcal{B}_1(pp)$
$\overline{(\sigma,\tau) \leftarrow SimSetup(1^{\lambda}); G \leftarrow \mathcal{A}(pp,\sigma);  \mathbf{return} \ (G,st = (\sigma,\tau))}$
$\mathcal{B}_{2}(\{(pk_{i,j}, c_{i,j}), x_{i}, z_{i}\}_{i \in [M], j \in [L]}, pp, st)$
$\overline{\operatorname{Run}b\leftarrow\mathcal{A}^{(\cdot)}(pp,\sigma);\mathbf{return}b}$
Queries of $\mathcal{A}$ are answered as follows:
-For the <i>i</i> -th query to $OP1_1$ : pick the tuple $(x_i, z_i, (pk_{i,1}, c_{i,1}));$
$\pi_{zk} \leftarrow SP_{zk}(\sigma_{zk}, \tau_{zk}, (x_i, pk_{i,1}, c_{i,1}));  \text{return } (x_i, (pk_{i,1}, c_{i,1}, \pi_{zk}), z_i) \text{ to } \mathcal{A}$
-For the <i>j</i> -th query $(EG, x_i)$ that specifies $x_i$ to $OP1_2$ : $\bar{x} \leftarrow EG_{sim}(pp, x_i)$
$\pi_{zk} \leftarrow SP_{zk}(\sigma_{zk}, \tau_{zk}, (\bar{x}, pk_{i,j}, c_{i,j}));  \text{return } (\bar{x}, (pk_{i,j}, c_{i,j}, \pi_{zk})) \text{ to } \mathcal{A}.$

Here we assume w.l.o.g. that  $\mathcal{A}$  queries with G at most M times, and uses an extended sampler to extend one statement  $x_i$  at most L times. It is easy to see that when  $\mathcal{B}$  is in  $\mathsf{Exp}_{\Sigma,\mathcal{B},\mathsf{PG}}^{\mathsf{priv},1}$ , *i.e.*, each  $(pk_{i,j}, c_{i,j})$  is generated by encrypting a uniformly chosen message  $m_{i,j}$  with a freshly generated  $pk_{i,j}$ , the above steps simulated by  $\mathcal{B}$  are identical to that in  $\mathsf{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{sim}}$ ; when  $\mathcal{B}$  is in  $\mathsf{Exp}_{\Sigma,\mathcal{B},\mathsf{PG}}^{\mathsf{priv},0}$ , these steps are identical to that in  $\mathsf{Event}_{\mathsf{zk},\mathcal{A}}^2$ . We have  $|\Pr[\mathsf{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{sim}}] - \Pr[\mathsf{Event}_{\mathsf{zk},\mathcal{A}}^2]| \leq \mathsf{Adv}_{\Sigma_{\mathsf{de}}}^{\mathsf{priv}}(\lambda)$  for any non-uniform PPT adversary  $\mathcal{A}$ .

Combining above lemmas, we have that for any non-uniform PPT adversary

$$|\Pr[\texttt{Event}_{zk}^{sim}] - \Pr[\texttt{Event}_{zk}^{\mathsf{real}}]| \leq \mathsf{Adv}_{\Sigma_{\mathsf{de}}}^{\mathsf{priv}}(\lambda) + \mathsf{Adv}_{\Pi_{zk}}^{zk}(\lambda) + \mathsf{negl}(\lambda).$$

Since we assume the zero-knowledgeness of  $\Pi_{zk}$  and the PRIV-MU-security of  $\Sigma_{de}$ , it holds that  $|\Pr[\texttt{Event}_{zk}^{sim}] - \Pr[\texttt{Event}_{zk}^{real}]| \leq \mathsf{negl}(\lambda)$ .

**Proof of authenticating soundness.** Now we prove the authenticating soundness of  $\Pi_{CD}$ . Assume there is a non-uniform PPT adversary  $\mathcal{A}$  which breaks the authenticating soundness, *i.e.*,  $\mathcal{A}$  outputs a valid proof  $\pi$  for a statement x such that either x is a false statement or no identifier witness of x can authenticate  $\pi$ . We first parse  $\pi = (pk, c, \pi_{zk})$ . From the definition of CD.Verify,  $\pi_{zk}$  is a valid proof for (x, pk, c) w.r.t. the proof system  $\Pi_{zk}$ . Then, from the soundness of  $\Pi_{zk}$ , the statement  $(x, pk, c) \in L_{CD}$  must be true, *i.e.*, there must be a witness  $(w^I, w^{NI}) \in R_L(x)$  and  $c = \mathsf{E}_{\mathsf{de}}(pk, w^I)$ . According the description of CD.Identify, this identifier witness  $w^I$  authenticates  $\pi$ . The above arguments contradicts our assumption that  $\mathcal{A}$  breaks the authenticating soundness. Therefore, we complete our proof.

**Proof of authenticating knowledge soundness.** We firstly establish the knowledge extractor for our construction. Let  $(\mathsf{Ext}_{\mathsf{zk},0},\mathsf{Ext}_{\mathsf{zk},1})$  be knowledge extractor of  $\Pi_{\mathsf{zk}}$ . The knowledge extractor  $(\mathsf{Ext}_0,\mathsf{Ext}_1)$  of  $\Pi_{CI}$  can be constructed as follows.

1.  $\mathsf{Ext}_0(1^{\lambda})$ . Firstly invoke the knowledge extractor of  $\Pi_{\mathsf{zk}}$ :  $(\sigma_{\mathsf{zk}}, \xi_{\mathsf{zk}}) \leftarrow \mathsf{Ext}_{\mathsf{zk},0}(1^{\lambda})$ ; Return the CRS  $\sigma = \sigma_{\mathsf{zk}}$  and the trapdoor  $\xi = \xi_{\mathsf{zk}}$ . 2.  $\mathsf{Ext}_1(\sigma, \xi, \pi)$ . Firstly parse the proof  $\pi = (pk, c, \pi_{\mathsf{zk}})$ ; Then, use the knowledge extractor of  $\Pi_{\mathsf{zk}}$ :  $(w^I, w^{NI}) \leftarrow \mathsf{Ext}_{\mathsf{zk},1}(\sigma_{\mathsf{zk}}, \xi_{\mathsf{zk}}, \pi_{zk}, (x, pk, c))$ .

Then we show the extracted witness  $(w^I, w^{NI}) \in R_L(x)$  and  $w^I$  authenticates  $\pi$ . By the definition of knowledge soundness, since  $\pi_{\mathsf{zk}}$  is a valid proof for (x, pk, c), we have  $(w^I, w^{NI}) \in R_L(x)$  and  $c = \mathsf{E}_{\mathsf{de}}(pk, w^I)$ . By the definition of the algorithm CD.Identify,  $w^I$  authenticates  $\pi$ . Thus, we complete the proof for the authenticating knowledge soundness.

**Proof of identifier uniqueness.** We prove the identifier uniqueness, by constructing a nonuniform PPT adversary C to break the soundness of  $\Pi_{zk}$  under the assumption that  $\Sigma_{de}$  is perfectly correct, by leveraging an adversary  $\mathcal{A}$  that can break the identifier uniquenessof  $\Pi_{CD}$ . Specifically, C is given a CRS  $\sigma_{zk}$ , and gives  $\sigma = (\sigma_{zk})$  to  $\mathcal{A}$ . C can honestly answer all queries made by  $\mathcal{A}$ using  $\sigma_{zk}$ . Therefore,  $\mathcal{A}$  will output  $(x, \pi = (pk, c, \pi_{zk}), w_1^I, w_2^I)$  s.t.  $\pi$  is authenticated by  $w_1^I$  and  $w_2^I$ . According to the definition of CD.Identify,  $\mathsf{E}_{\mathsf{de}}(pk, w_1^I) = \mathsf{E}_{\mathsf{de}}(pk, w_2^I)$ . However, according to the perfect correctness, if pk is valid,  $\mathsf{E}_{\mathsf{de}}$  is an injection. Therefore, pk is not valid, and the statement (x, pk, c) is not true. Thus  $((x, pk, c), \pi_{zk})$  contradicts the soundness of  $\Pi_{zk}$ .

**Proof of related-witness unforgeability.** Recall the definition of related-witness unforgeability in Def.19, and denote the event that the adversary  $\mathcal{A}$  succeeds by

$$\operatorname{Event}_{\operatorname{runf},\mathcal{A}} := \begin{bmatrix} pp \leftarrow \operatorname{PG}(1^{\lambda}); \sigma \leftarrow \operatorname{Setup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp, \sigma); \\ (x^*, \pi^*, \phi) \leftarrow \mathcal{A}^{\mathcal{O}_{\operatorname{P1}}, \mathcal{O}_{\operatorname{P2}}}(\sigma, pp) : (x^*, \pi^*) \notin \operatorname{Hist} \\ \wedge \operatorname{Verify}(\sigma, x^*, \pi^*) = 1 \land \exists w^I \in st, \operatorname{Identify}(\sigma, x^*, \pi^*, \phi(w^I)) = 1 \end{bmatrix},$$

where  $\phi$  is a transformation satisfying the bounded root space and sampleable root space (cf. Remark.5 after Def.19). We prove  $\Pr[\text{Event}_{runf,\mathcal{A}}] \leq \operatorname{negl}(\lambda)$  by considering the following hybrid event. Here, we denote the simulation extractor of  $\Pi_{zk}$  by  $\mathsf{SE}_{zk}, \mathsf{SP}_{zk}, \mathsf{SExt}_{zk}$ .

The hybrid event  $\text{Event}_{\text{runf},\mathcal{A}}^1$ . This event is almost identical to  $\text{Event}_{\text{runf},\mathcal{A}}$ , except that Setup is replaced by the following SimSetup:

- SimSetup $(1^{\lambda})$ : Run the simulation extraction setup  $\mathsf{SE}_{\mathsf{zk}}$  of  $\Pi_{\mathsf{zk}}$ :  $(\sigma_{\mathsf{zk}}, \tau_{\mathsf{zk}}, \xi_{\mathsf{zk}}) \leftarrow \mathsf{SE}_{\mathsf{zk}}(1^{\lambda})$ . Set  $\sigma = (\sigma_{\mathsf{zk}}), \ \tau = \tau_{\mathsf{zk}}$ .

And the oracles  $\mathcal{O}_{P1}$ ,  $\mathcal{O}_{P2}$  are replaced with the oracles  $(OP_1, OP_2)$ , which are similar to  $(OP1_1, OP1_2)$  that we introduced in Event<sup>1</sup><sub>zk</sub> (given in the proof for entropic ZK), except that the used  $(\sigma_{zk}, \tau_{zk})$  is generated by  $SE_{zk}$  instead of  $SS_{zk}$ .

Following the similar arguments in Lemma.1, we have: for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$|\Pr[\texttt{Event}^{1}_{\mathsf{runf},\mathcal{A}}] - \Pr[\texttt{Event}_{\mathsf{runf},\mathcal{A}}]| \le \mathsf{negl}(\lambda).$$

Next, we show that  $Event_{runf}^1$  happens with negligible probability.

**Lemma 4.** If the DPKE scheme  $\Sigma_{de}$  is PRIV-IND-MU-secure w.r.t.  $2^{-k}$ -hard-to-invert auxiliary inputs, and  $\Pi_{zk}$  is a simulation-extractable NIZK, it follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,  $\Pr[Event^{1}_{runf,\mathcal{A}}] \leq negl(\lambda)$ .

*Proof.* To facilitate our proof, we present a special form of security definition of  $\Sigma_{de}$ , which captures the message recovery security.

$Exp^{rec}_{\varSigma,\mathcal{B},PG}(\lambda)$
$pp \leftarrow PG(\lambda)$
$(G, st) \leftarrow \mathcal{B}_1(pp)$
for $i \in [M(\lambda)]$
$(x_i, w_i^I, w_i^{NI}, z_i) \leftarrow G(pp); \eta_i = (x_i, z_i, pp)$
for $j \in [L(\lambda)]$
$(pk_{i,j}, sk_{i,j}) \leftarrow KeyGen(1^{\lambda}); c_{i,j} \leftarrow Enc(pk_{i,j}, w_i^I)$
$w_*^I \leftarrow \mathcal{B}_2(\{(pk_{i,j}, c_{i,j}), \eta_i\}_{i \in [L], j \in [M]}, st)$
<b>return</b> 1 if $\exists i \in [M]$ s.t. $w_i^I = w_*^I$

Following Corollary.2, it holds that for any non-uniform PPT adversary  $\mathcal{B}$ ,  $\Pr[\mathsf{Exp}_{\Sigma,\mathcal{B},\mathsf{PG}}^{\mathsf{rec}}(\lambda)] \leq \mathsf{negl}(\lambda)$ . Otherwise, one can use the  $w_*^I$  to decide wether  $c_{i,j}$  is an encryption for  $w_i^I$ . Assuming there is an efficient adversary  $\mathcal{A}$  s.t.  $\Pr[\mathsf{Event}_{\mathsf{runf},\mathcal{A}}^1] = \epsilon$ , we can construct an adversary  $\mathcal{B}$  making  $\Pr[\mathsf{Exp}_{\Sigma,\mathcal{B},\mathsf{PG}}^{\mathsf{rec}}(\lambda) = 1] = \frac{\epsilon + \mathsf{negl}(\lambda)}{q}$ , where  $q = \mathsf{Max}_{\phi \in \Phi, \forall y}(|\phi^{-1}(y)|)$ . We write the code of  $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$  in the following.

$\mathcal{B}_1(pp)$
$\overline{(\sigma_{zk}, \tau_{zk}, \xi_{zk})} \leftarrow SE_{zk}(1^{\lambda});  G \leftarrow \mathcal{A}(pp, \sigma_{zk});  \mathbf{return} \ (G, st = (\sigma_{zk}, \tau_{zk}, \xi_{zk}))$
$\mathcal{B}_{2}(\{(pk_{i,j}, c_{i,j}), x_{i}, z_{i}\}_{i \in [M], j \in [L]}, pp, st)$
Run $(x^*, \pi^*, \phi) \leftarrow \mathcal{A}^{(\cdot)}(pp, \sigma);$ Parse $\pi^* = (pk^*, c^*, \pi^*_{zk})$
$(\hat{w}_*^I, \hat{w}_*^{NI}) \leftarrow SExt_{zk}(\sigma_{zk}, \xi_{zk}, (x^*, pk^*, c^*), \pi_{zk}^*);  \mathbf{return} \ w_*^I \leftarrow \$ \ \phi^{-1}(\hat{w}_*^I)$
Queries of $\mathcal{A}$ are answered as follows:
-For the <i>i</i> -th query to $OP1_1$ : pick the tuple $(x_i, z_i, (pk_{i,1}, c_{i,1}))$
$\pi_{zk} \leftarrow SP_{zk}(\sigma_{zk}, \tau_{zk}, (x_i, pk_{i,1}, c_{i,1}));  \text{return } (x_i, (pk_{i,1}, c_{i,1}, \pi_{zk}), z_i) \text{ to } \mathcal{A}$
-For the <i>j</i> -th query (EG, $x_i$ ) that specifies $x_i$ to $OP1_2$ : $\bar{x} \leftarrow EG_{sim}(pp, x_i)$
$\pi_{zk} \leftarrow SP_{zk}(\sigma_{zk}, \tau_{zk}, (\bar{x}, pk_{i,j}, c_{i,j})); \qquad \text{return } (\bar{x}, (pk_{i,j}, c_{i,j}, \pi_{zk})) \text{ to } \mathcal{A}.$

Note that the ouputs of  $\mathsf{EG}_{\mathsf{real}}$  and  $\mathsf{EG}_{\mathsf{sim}}$  are computationally indistinguishable. At the point of  $\mathcal{A}$ 's view, the experiment simulated by  $\mathcal{B}$  is computationally indistinguishable with that in  $\mathsf{Event}^1_{\mathsf{runf},\mathcal{A}}$ . By assumption,  $\mathcal{A}$  will output  $(x^*, \pi^* = (pk^*, c^*, \pi^*_{\mathsf{zk}}), \phi)$  that will be authenticated by some  $(\phi(w_i^I))$  with an advantage  $\epsilon$ , *i.e.*,  $\mathsf{E}_{\mathsf{de}}(pk^*, \phi(w_i^I)) = c^*$ . Since pk is a valid public key and  $\Sigma_{\mathsf{de}}$  has the perfect correctness, we have  $w^* = \phi(w_i^I)$ . According to the bounded root space and samplable root space requirement on  $\phi$ , the root space of  $\phi$  is bounded by a polynomial q. It follows that  $\Pr[\mathsf{Exp}_{\Sigma,\mathcal{B},\mathsf{PG}}^{\mathsf{rec}}(\lambda) = 1] = \frac{\epsilon + \mathsf{negl}(\lambda)}{q}$ . Then, from the security of DPKE, we have  $\Pr[\mathsf{Event}^1_{\mathsf{runf}}] \leq \mathsf{negl}(\lambda)$ .

# 5 Non-malleable (Perfectly One-way) Hash Functions from Standard Assumptions

We will present three different applications in (non-malleable) hash (in this section), (group) signature (in Sect.6), and (plaintext-checkable) public key encryption (in Sect.7) respectively, and we will show how to advance the state of the art in each domain.

Many efforts have been made formalizing meaningful cryptographic properties to realize random oracles. Perfectly one-way hash [18] and non-malleable hash functions [9] are notable examples. They are used to instantiate random oracles in e.g., Bellare-Rogaway encryption [6], HMAC [29], and OAEP [10] respectively. In particular, a perfectly one-way hash is a probabilistic function that requires the output to hide all partial information about the input (even with auxiliary information about the input), while still enabling the check of the validity of an evaluation. And non-malleable hash requires that one cannot malleate a hash value into a related one even with some auxiliary information about the pre-image. Moreover, collision resistance is also required in both as it is necessary for many of their interesting applications, such as instantiating random oracles in Bellare-Rogaway encryption [6,9].



Fig. 3. Known results about non-malleable hash

Unfortunately, both perfectly one-way hash and non-malleable hash (with general auxiliary inputs) have no construction from any efficiently falsifiable assumption [18,25,9,3]. For clarity, we illustrate known results about non-malleable hash in Fig.3. Canetti's construction [18] is the only known one satisfying perfect one-wayness with general auxiliary inputs and collision resistance, and it is based on a non-efficiently-falsifiable form of DDH assumption. The initial work of Boldyreva *et al.* [9] presented constructions of non-malleable hash from perfectly one-way hash functions [18] and simulation-extractable NIZKs [41], thus directly inherits the non-efficiently falsifiable assumption from [18]. Note that collision resistance of the underlying perfectly one-way hash is essential for proving non-malleability in [9]. Thus, those standard-assumptions-based perfectly one-way functions without collision resistance [7] cannot imply a non-malleable function (even without collision resistance) along the way of [9]. Furthermore, Baecher *et al.* [3] showed another construction for a non-malleable hash, but it requires a random oracle. Recall that the primary motivation of nonmalleable hash was to instantiate random oracles.

Note that the drawbacks the non-standard assumptions made by [18] have become much more serious: the assumption is known to contradict the existence of iO [16], while recent progress [33] demonstrated the feasibility of iO from some well-studied assumptions. The mere existence of such a non-malleable hash or perfectly one-way hash becomes unclear, and a basic question remains:

# Does there exist a non-malleable (or perfectly one-way) hash function w.r.t. general auxiliary information from standard assumptions?

We solve both problems by using waNIZKs. Our framework could give concrete constructions for non-malleable *and* perfectly one-way hash functionss with any sub-exponentially hard-to-invert auxiliary inputs, assuming only the standard assumptions like *d*-linear assumption. We directly construct a hash function that satisfies *perfect one-wayness*, *non-malleability* and *collision resistance* simultaneously. We simply name it non-malleable (perfectly one-way) hash.

## 5.1 Definition

A hash function  $\mathcal{H}$  is defined by a triple of PPT algorithms:

- $\mathsf{HK}(1^{\lambda})$ . Generate a key hk of the hash function.
- H(hk, s). On inputs a key hk and an input s output a hash value y.
- HVf(hk, s, y). On inputs hk, s and y return a decision bit.

The correctness requires for any hk, s, it holds that  $\mathsf{HVf}(hk, s, \mathsf{H}(hk, s)) = 1$ . For security, the hash function  $\mathcal{H}$  is required to first satisfy:

- Perfect one-wayness w.r.t.  $\epsilon$ -hard-to-invert auxiliary inputs. I.e., for any distribution  $S = \{S_{\lambda}\}_{\lambda \in \mathbb{N}}$  and any hint function hint such that hint is  $\epsilon$ -hard-to-invert w.r.t. S, and for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\Pr\begin{bmatrix} hk \leftarrow \mathsf{HK}(1^{\lambda}), s_0 \leftarrow S_{\lambda}, s_1 \leftarrow \{0, 1\}^{|s_0|}, b \leftarrow \$\{0, 1\}, \\ y \leftarrow \mathsf{H}(hk, s_b), b' \leftarrow \mathcal{A}(hk, y, \mathsf{hint}(hk, s_0)) : b = b' \end{bmatrix} \le \mathsf{negl}(\lambda).$$

- Collision resistance. I.e., for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$\Pr \begin{bmatrix} hk \leftarrow \mathsf{HK}(1^{\lambda}), (s, s', y) \leftarrow \mathcal{A}(hk) : \\ s \neq s' \land \mathsf{HVf}(hk, s, y) = \mathsf{HVf}(hk, s', y) = 1 \end{bmatrix} \leq \mathsf{negl}(\lambda)$$

For definition of non-malleability, we adopt it from [3] as this game-based definition is easier to use (than the simulation definition from [9]), and sufficient for all major applications including Bellare-Rogaway encryption [6], HMAC [29], and OAEP [10]. Informally, non-malleability requires that an adversary, seeing a hash value y = H(hk, s) and an auxiliary input hint(hk, s), cannot find another  $y^*$  whose pre-image is meaningfully related to s. We note the "relation" between the preimages is described a transformation set  $\Phi$ , namely, s' is  $\Phi$ -related to s if  $s' = \phi(s)$  for some  $\phi \in \Phi$ . The non-malleability is defined w.r.t. a transformation set  $\Phi$  rather than any transformation  $\phi$ , since there exists some relation such as constant transformations, for which this definition is hopeless. In this work, we will adopt on transformations that have the so-called bounded root space (BRS) and samplable root space (SRS) (denoted by  $\Phi_{brs}^{srs}$ ) developed in [22], which are the currently most general yet achievable class.

**Definition 21 (Non-malleability of hash [3]).** A hash function  $\mathcal{H}$  is non-malleable w.r.t. a transformation set  $\Phi$  and  $\epsilon$ -hard-to-invert inputs, if for any non-uniform PPT adversary  $\mathcal{A} = (\mathcal{A}_d, \mathcal{A}_y)$ ,

$$\Pr[\mathsf{Exp}^{\mathsf{nm}}_{\Phi,\mathcal{H},\mathcal{A}}(\lambda) = 1] \le \mathsf{negl}(\lambda),$$

where the experiment is defined as follows.

$Exp^{nm}_{\varPhi,\mathcal{H},\mathcal{A}}(\lambda)$	
$\overline{hk \leftarrow HK(1^{\lambda});  (S_{\lambda}, st) \leftarrow \mathcal{A}_d(hk);  s \leftarrow S_{\lambda}, h_s \leftarrow hint(hk, s);}$	$y \leftarrow H(hk,s)$
$(y^*,\phi) \leftarrow \mathcal{A}_y(y,h_s,st_d)$	
<b>return</b> 1 <b>if</b> $\phi(s) \neq s \land HVf(hk, \phi(s), y^*) = 1 \land \phi \in \Phi$	

The distribution  $S_{\lambda}$  outputted by  $\mathcal{A}_d$  should be  $\epsilon$ -hard-to-invert w.r.t. hint. That is for every hk, for any non-uniform PPT adversary  $\mathcal{B}$ , it holds that  $\Pr[s \leftarrow S_{\lambda} : s = \mathcal{B}(hk, \mathsf{hint}(hk, s), st)] \leq \epsilon(\lambda)$ .

#### 5.2 Construction

Observe that non-malleable (perfectly one-way) hash has three security requirements and a verify algorithm. If we start just with perfect one-wayness (without the verifier algorithm) which hides all partial information, there are plenty of candidates; for example, a *commitment* scheme. For the remaining challenges of collision resistance and validity checking (while maintaining best possible privacy), our waNIZK becomes an immediate choice. For non-malleability, it can come from related-witness unforgeability. We define the evaluation as first committing to its input and then attaching a proof of the well-formedness of the commitment using our waNIZK proof!

More precisely, let  $COM = \{K_{com}, C_{com}\}$  be a commitment scheme, and let  $\Pi_{wa} = \{S_{wa}, P_{wa}, V_{wa}, I_{wa}\}$  be a WA-NIZK for an NP language  $L_{nm} := \{(c, k_{com}); (s, r) : c = C_{com}(k_{com}, s; r)\}$ , in which s is the identifier witness. Here we require  $\Pi_{wa}$  to satisfy the identifier uniqueness, the entropic ZK and the related-witness unforgeability w.r.t. all  $(-\log \epsilon)$ -unpredictable samplers and the transformation set  $\mathcal{P}_{brs}^{srs}$ . We present the detailed description in Fig.4.

$HK(1^{\lambda})$
$\sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \text{ and } k_{com} \leftarrow K_{com}(1^{\lambda}); \mathbf{return} \ hk = (\sigma_{wa}, k_{com})$
H(hk,s)
$\overline{c_{com} \leftarrow C_{com}(k_{com}, s; r); \pi_{wa} \leftarrow \Pi_{wa}(\sigma_{wa}, (c_{com}, k_{com}), (s, r)); \mathbf{return} \ y = (c_{com}, \pi_{wa})}$
HVf(hk,s,y)
$\textbf{return 1}  \textbf{if } V_{wa}(\sigma_{wa}, (c_{com}, k_{com}), \pi_{wa}) = 1 \land I_{wa}(\sigma_{wa}, (c_{com}, k_{com}), \pi_{wa}, s) = 1$

Fig. 4. Non-malleable Hash from commitment+ waNIZKs

#### 5.3 Security Analysis

The correctness follows the correctness of underlying primitives. Regarding collision resistance, if two distinct inputs  $(s_1, s_2)$  (which are identifier witnesses) authenticate the sample proof, it immediately breaks identifier uniqueness. Notice that the hash value y consists of a hiding commitment and a WA-NIZK proof, both of which won't leak partial information about an unpredictable input. Thus, the perfect one-wayness follows easily. Regarding the non-malleability, notice that a mauled hash value must contain a mauled waNIZK proof, which is prevented by the related-witness unforgeability of the waNIZK. **Theorem 2.**  $\mathcal{H}$  satisfies the perfect one-wayness w.r.t.  $\epsilon$ -hard-to-invert auxiliary inputs, collision resistance, and non-malleability w.r.t. the transformation set  $\Phi_{brs}^{srs}$  and  $\epsilon$ -hard-to-invert auxiliary inputs, if the commitment scheme COM satisfies computationally hiding, and  $\Pi_{wa}$  satisfies the identifier uniqueness, the entropic ZK and the related-witness unforgeability w.r.t. the transformation set  $\Phi_{brs}^{srs}$  and all  $(-\log \epsilon)$ -unpredictable samplers.

**Proof for non-malleability.** We prove the non-malleability w.r.t transformations with bounded and sampable root space, under the assumption that  $\Pi_{wa}$  satisfies the related-witness unforgeability. Precisely, assuming there exists an efficient adversary breaking the non-malleability, we can construct an adversary  $\mathcal{B}$  that breaks the related-witness unforgeability. Recall the related-witness unforgeability definition, where  $\mathcal{B}$  is given a CRS  $\sigma_{wa}$ , a trusted parameter  $pp = k_{com} \leftarrow \mathsf{K}_{com}(1^{\lambda})$ , and oracle accesss to  $\mathcal{O}_{\mathsf{P}_1}$  and  $\mathcal{O}_{\mathsf{P}_2}$ . For  $\mathcal{A}_d$ , we define a sampler  $G_{\mathcal{A}}$  as follows, where  $hk = (\sigma_{wa}, k_{com})$ :

$$(S_{\lambda}, st) \leftarrow \mathcal{A}_{d}(hk); s \leftarrow S_{\lambda}; r \leftarrow \mathbb{R}_{\mathsf{com}}; c_{\mathsf{com}} \leftarrow \mathsf{C}_{\mathsf{com}}(k_{\mathsf{com}}, s; r); h_{s} \leftarrow \mathsf{hint}(hk, s);$$
  
**return**  $x = (k_{\mathsf{com}}, c_{\mathsf{com}}), w^{I} = s; w^{NI} = r, z = (h_{s}, st),$ 

where  $R_{com}$  is the randomness space of COM. It is easy to verify that such a sampler is k-unpredictable w.r.t.  $K_{com}$ 

We describe the strategy of  $\mathcal{B}$  as follows.

$$\begin{aligned} & \frac{\mathcal{B}^{\mathcal{O}_{\mathsf{P}_{1}},\mathcal{O}_{\mathsf{P}_{2}}}(\sigma_{\mathsf{wa}},k_{\mathsf{com}})}{\text{Set }hk = (\sigma_{\mathsf{wa}},k_{\mathsf{com}}); \text{Query }\mathcal{O}_{\mathsf{P}1} \text{ with } G_{\mathcal{A}} \text{ and obtain } (c_{\mathsf{com}},\pi_{\mathsf{wa}},h_{s},st)} \\ \text{Set }y = (c_{\mathsf{com}},\pi_{\mathsf{wa}}); (y^{*},\phi) \leftarrow \mathcal{A}_{y}(y,h_{s},st_{d}) \\ \mathbf{return } (x = (hk,c_{\mathsf{com}}),\pi_{\mathsf{wa}},\phi) \end{aligned}$$

Notice at the point of  $\mathcal{A}$ 's view, the experiments simulated by  $\mathcal{B}$  is identical to the real nonmalleability experiment. Therefore,  $\mathcal{A}$  will output  $(y^*, \phi)$  such that  $\mathsf{HVf}(hk, y^*, \phi(s)) = 1$ . Parse  $y^* = (c^*_{\mathsf{com}}, \pi^*_{\mathsf{wa}})$ , and it follows that  $\pi^*_{\mathsf{wa}}$  will be authenticated by s. Notice  $y^* \neq y$ , and then  $(k_{\mathsf{com}}, c^*_{\mathsf{com}}, \pi^*_{\mathsf{wa}})$  gives a successful attack on the related-witness unforgeability.

**Proof for perfect one-wayness.** We prove the perfect one-wayness under assumptions that COM is hiding and  $\Pi_{wa}$  satisfies the entropic ZK w.r.t. all k-unpredictable samplers. Specifically, for a distribution  $\{S_{\lambda}\}_{\lambda \in \mathbb{N}}$  and a hint function hint, we define a sampler  $G_S$  as follows, where  $hk = (\sigma_{wa}, k_{com})$ :

$$\begin{split} \mathbf{s} \leftarrow S_{\lambda}; r \leftarrow & \$ \ R_{\mathsf{com}}; c_{\mathsf{com}} \leftarrow \mathsf{C}_{\mathsf{com}}(k_{\mathsf{com}}, s; r); h_s \leftarrow \mathsf{hint}(hk, s); \\ & \mathbf{return} \ x = (k_{\mathsf{com}}, c_{\mathsf{com}}), w^I = s; w^{NI} = r, z = (h_s, st) \end{split}$$

It is easy to verify that such a sampler is k-unpredictable w.r.t.  $K_{com}$ , if COM is hiding.

Assume the simulator of  $\Pi_{wa}$  is  $(SS_{wa}, SP_{wa})$ . We consider the an experiment  $Exp_{pow}$  in which the hash key  $hk = (k_{com}, \sigma_{wa})$  is generated as  $k \leftarrow K_{com}(1^{\lambda})$  and  $(\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda})$ , the hash function  $H(hk, s_b)$  returns  $c_{com} \leftarrow C_{com}(k_{com}, s_b; r)$  and  $\pi_{wa} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, c_{com}, k_{com})$ . First, from the entropic ZK of  $\Pi_{wa}$ , the two experiments are indistinguishable, at the point of  $\mathcal{A}$ 's view. Therefore, the probability that  $\mathcal{A}$  correctly guess the value b in this experiment is computationally indistinguishable with that in the original experiment. Next, notice that  $\pi_{wa}$  is generated independently of  $s_b$ , and the advantage of  $\mathcal{A}$  thus is equal to the advantage of breaking the hiding property of COM. Therefore, our construction enjoys the perfect one-wayness. **Proof for collision resistance.** We show our construction is collision-resistant assuming the identifier uniqueness of  $\Pi_{wa}$ . More precisely, if for an honestly generated  $hk = (\sigma_{wa}, k_{com})$ , there are two inputs x and x' satisfying HVf(hk, x, y) = HVf(hk, x', y) = 1 for some  $y = (c_{com}, \pi_{wa})$ . It means both x and x' can authenticate  $\pi_{wa}$ , which contradicts the identifier uniqueness of  $\Pi_{wa}$ .

#### 6 Group signatures with verifier-local revocation with auxiliary input

In group signatures with verifier local revocation (VLR) [12], we insist that the verifier can check by himself whether a signature is generated by a revoked group member, so that the group public key and the signing complexity are *independent* of revocation list which could be potentially long. In this section, we show how waNIZKs give rise to a simple VLR group signature scheme. Particularly, our construction enjoys auxiliary-input security, which is against a "side-channel" attacker who is allowed to see some computationally hard-to-invert function of the user's secret key. To the best of our knowledge, known VLR group signatures cannot guarantee auxiliary-input security.

Why we consider the auxiliary-input security. Besides that "side-channel attacks" are a threat for every cryptographic primitive, and that the auxiliary-input model is currently the strongest model capturing memory leakage (more details about the model are referred to [26]), we find the auxiliary-input security for VLR group signatures is interesting both practice-wise and technical-wise.

Practice-wise, some instantiation of VLR group signatures, such as the direct anonymous attestation (DAA) [14] (along with its improved version, the EPID signature [15]), is adopted by the Trusted Computing Group as the standard for remote authentication, and implemented in several trusted platform modules (TPM) including Intel's SGX. These TPMs are essential for computer security but are shown, by numerous works, vulnerable to side-channel attacks [39]. The study of auxiliary-input secure VLR group signature could enhance the security of TPMs against side-channel attacks.

Technique-wise, constructing auxiliary-input secure VLR group signatures turns out to be a non-trivial task. First, it is unclear how to easily "lift" existing constructions. Most of existing VLR group signature schemes (such as [35,12,14,15]) leverage certain "pseudorandom functions" on a secret to preserve the anonymity while enable verifier-local checking. Such "pseudorandomness" either comes from underlying algebraic assumptions or directly from a PRF (e.g., a recent construction from Boneh et al. [11]). Unfortunately, with the auxiliary input on the secret, "pseudorandomness" collapses. Essentially, in a VLR group signature, it will need an auxiliary-input secure secret-key-based tag generation mechanism that is identifiable (for realizing the revocation functionality), unforgeable, and does not leak any partial information about the signer identity (for anonymity). Our waNIZK provides a perfect tool.

### 6.1 The Definitions

A VLR group signature scheme  $\Sigma_{gs}$  is defined by a tuple of three PPT algorithms.

- GS.KeyGen $(1^{\lambda}, n)$ . It outputs a group public key gpk, and for each user  $i \in [n]$ , outputs the secret key gsk[i] along with the revocation token grk[i].
- GS.Sign $(gpk, \mathbf{gsk}[i], m)$ . It outputs a valid signature  $\vartheta$  for m under gpk.
- GS.Verify $(gpk, RL, \vartheta, m)$ . It returns either 1 indicating that  $\vartheta$  is a valid signature for m and was not signed by a revoked user whose token is in RL, or 0 otherwise. Here RL is a set of revocation tokens.

A VLR group signature scheme  $\Sigma_{gs}$  is correct, if for  $(gpk, gsk, grk) \leftarrow \mathsf{GS.KeyGen}(1^{\lambda})$ , every  $RL \subset \mathbf{grk}$ , every message  $m \in \{0, 1\}^*$ ,

 $\mathsf{GS.Verify}(gpk, RL, \mathsf{GS.Sign}(gpk, \mathbf{gsk}[i], m), m) = 1 \Leftrightarrow \mathbf{grk}[i] \notin RL.$ 

Note that this verification algorithm allows the group manager, who knows all revocation tokens, to *trace* signer's identifier for every valid signatures. Specifically, if GS.Verify $(gpk, \emptyset, \vartheta, m) = 1$  and GS.Verify $(gpk, \mathbf{grk}[i^*], \vartheta, m) = 0$ , the signer of  $\vartheta$  will be traced to user  $i^*$ .

A VLR group signature scheme should satisfy the anonymity and the traceability. Anonymity ensures that the identity of an uncorrupted signer is indistinguishable from all possible signers, even when the adversary is allowed to see many signatures from all users and to corrupt some  $\mathbf{gsk}[i]$  and  $\mathbf{grk}[i]$ . Traceability captures that any non-uniform PPT adversary  $\mathcal{A}$  can neither produce a valid signature-message pair  $(\vartheta, m)$  that won't be traced to any user, *i.e.*,  $\mathsf{GS.Verify}(gpk, \mathbf{grk}, \vartheta, m) = 1$ , nor frame an uncorrupted user  $i^*$ , *i.e.*,  $\mathsf{GS.Verify}(gpk, \mathbf{grk}[i^*], \vartheta, m) = 0$ , even when the adversary are allowed to obtain signatures from all users and to corrupt some  $\mathbf{gsk}[i]$  and  $\mathbf{grk}[i]$ .

In this paper, we present the auxiliary-input counterparts of them. Particularly, we consider the auxiliary inputs as a hard-to-invert function on users' secret keys along with the group public key, since user's devices are much more vulnerable than the group manager's device that is usually supposed to be well-protected. We use the following oracles to model the adversary's capability.

- $Corr_{gsk,grk}(i)$ . The corruption oracle takes as input an index *i*, and outputs (gsk[i], grk[i]).
- $\mathsf{OSign}_{\mathbf{gsk}}(i, m)$ . The signing oracle takes as inputs an index *i* and a message *m*. It returns  $\vartheta \leftarrow \mathsf{GS.Sign}(gpk, \mathbf{gsk}[i], m)$ .

**Definition 22 (Anonymity).** A VLR group signature scheme  $\Sigma_{gs}$  satisfies the (auxiliary-input) anonymity (w.r.t. a family of leakage functions  $\mathcal{F}$ ), if for any non-uniform PPT adversary  $\mathcal{A}$  (and any  $f \in \mathcal{F}$ ), it follows that

$$|\Pr[\mathsf{Exp}^{\mathsf{anon},0}_{\Sigma_{\mathsf{gs}},\mathcal{A}}(\lambda,n)=1] - \Pr[\mathsf{Exp}^{\mathsf{anon},1}_{\Sigma_{\mathsf{gs}},\mathcal{A}}(\lambda,n)=1]| \le \mathsf{negl}(\lambda).$$

The experiments  $\operatorname{Exp}_{\Sigma_{gs},\mathcal{A}}^{\operatorname{anon},\beta}$  for  $\beta = 0,1$  are defined in the following, where  $\overrightarrow{f}(gpk, \mathbf{gsk})$  denotes  $(f(gpk, \mathbf{gsk}[i]))_{i \in [n]}$ , and the challenge users  $(i_0^*, i_1^*)$  are not corrupted.



**Definition 23 (Traceability).** A VLR group signature scheme  $\Sigma_{gs}$  satisfies the (auxiliary-input) traceability (w.r.t. a family of leakage functions  $\mathcal{F}$ ), if for any non-uniform PPT adversary  $\mathcal{A}$  (and any  $f \in \mathcal{F}$ ), it follows that

$$\Pr[\mathsf{Exp}^{\mathsf{trace}}_{\Sigma_{\mathsf{gs}},\mathcal{A}}(\lambda,n)=1] \le \mathsf{negl}(\lambda).$$

The experiment  $\mathsf{Exp}_{\Sigma_{gs},\mathcal{A}}^{\mathsf{trace}}$  is defined in the following, where U denotes the set of all corrupted users.

$$\begin{split} & \frac{\mathsf{Exp}_{\Sigma_{\mathsf{gs}},\mathcal{A}}^{\mathsf{trace}}(\lambda,n)}{(gpk,\mathsf{gsk},\mathsf{grk}) \leftarrow \mathsf{GS}.\mathsf{KeyGen}(1^{\lambda},n)} \\ & (RL^*,\vartheta^*,m^*,i^*) \leftarrow \mathcal{A}^{\mathsf{Corr},\mathsf{OSign}_{\mathbf{gsk}}}(gpk,\overrightarrow{f}(gpk,\mathsf{gsk})) \\ & \texttt{return 1} \quad \texttt{if} \quad \mathsf{GS}.\mathsf{Verify}(gpk,RL^*,\vartheta^*,m^*) = 1 \land \\ & ((\mathsf{GS}.\mathsf{Verify}(gpk,\mathsf{grk}[i^*],\vartheta^*,m^*) = 0,i^* \notin U) \\ & \lor (\mathsf{GS}.\mathsf{Verify}(gpk,\mathsf{grk},\vartheta^*,m^*) = 1)) \end{split}$$

Remark 6. We note in the original definition the adversary does not specify a user  $i^*$  to be "framed", and the success condition GS.Verify $(gpk, \mathbf{grk}[i^*], \vartheta^*, m^*) = 0$  is only required to hold for an arbitrary  $i^* \notin U$ . Nonetheless, it is easy to verify the two definitions are equivalent since n is polynomial-bounded.

### 6.2 Construction

Our construction is based on the following observation. On rough terms, if a group signature scheme allows one to efficiently check whether a group signature was generated by using (a part of)  $\mathbf{gsk}[i]$ , then a VLR group signature can be built upon this as follows. 1) Set  $\mathbf{grk}[i]$  to be (the specific part of)  $\mathbf{gsk}[i]$ , and 2) the verification algorithm performs as follows.

- Verify the group signature as the underlying verification algorithm does.
- If valid, for each  $\mathbf{gsk}[i] \in RL$ , identify whether  $\vartheta$  was created by  $\mathbf{gsk}[i]$ . If  $\vartheta$  is not identified by any  $\mathbf{gsk}[i]$ , accept it; otherwise, reject it.

The remaining part is to design a group signature with this identifiability. Note that the folklore of designing group signatures is to employ a simulation extractable NIZK to prove the knowledge of a group membership certificate [20] where the proof is taken as the signature. Then, our waNIZK will be an immediate choice to add the identifiability, by *replacing* the NIZK in this folklore. Moreover, the authenticating knowledge soundness and unforgeability of waNIZK would be enough to replace simulation extractability. Actually, with the two properties, we can bind a message m to a waNIZK proof (for preventing adversaries from forging signatures), by taking m as a "dummy" part of the statement being proved; this approach is rather standard when constructing signatures from simulation-extractable NIZKs.

Specifically, we consider a pair (ID, Sig), where ID is a bit string with sufficient length, and Sig is a digital signature for ID under a verification key  $vk_{sig}$  of the group manager. To sign a message m on behalf of the group, one just uses a waNIZK to prove the knowledge of such a pair w.r.t.  $(vk_{sig}, m)$  where ID is set to be the identifier. The auxiliary-input security follows the fact that all security guarantee of waNIZKs are preserved when auxiliary-information about witnesses <sup>12</sup> is leaked to adversaries.

More formally, let  $\Sigma_{sig} = \{K_{sig}, S_{sig}, V_{sig}\}$  be a standard-model digital signature scheme that satisfies the standard *existentially unforgeability against chosen message attacks* (EU-CMA). Let  $\Pi_{wa} = \{S_{wa}, P_{wa}, V_{wa}, I_{wa}\}$  be a WA-NIZK for the following language:

 $\mathsf{L}_{\mathsf{VLR}}: \{(vk_{\mathsf{sig}}, m); (ID, \mathsf{Sig}) : \mathsf{V}_{\mathsf{sig}}(vk_{\mathsf{sig}}, \mathsf{Sig}, ID) = 1\},\$ 

<sup>&</sup>lt;sup>12</sup> Since in the auxiliary-input model, this leakage could depend on the public parameter, which requires the underlying waNIZK to work for CRS-dependent samplers.

where ID is the identifier witness. The VLR group signature  $\Sigma_{gs}$  is presented in Fig.5, where  $id(\lambda)$  is a integer function polynomial in  $\lambda$ .

$GS.KeyGen(1^\lambda,n)$	
$\overline{(vk_{sig}, sk_{sig}) \leftarrow K_{sig}(1^{\lambda})}; \sigma_{wa} \leftarrow S_{wa}(1^{\lambda})$	
For $i \in [n], ID_i \leftarrow \{0, 1\}^{id(\lambda)}, Sig_i \leftarrow S_{sig}(vk_{sig}, sk_{sig}, ID_i)$	
<b>return</b> $gpk = (\sigma_{wa}, vk_{sig})$ , and for $i \in [n], \mathbf{gsk}[i] = (ID_i, \mathbf{Sig}_i), \mathbf{grk}[i] = ID_i$	
$\frac{GS.Sign(gpk,\mathbf{gsk}[i],m)}{2}$	
Parse $gpk = (\sigma_{wa}, vk_{sig}), \mathbf{gsk}[i] = (ID_i, Sig_i)$	
$\mathbf{return} \ \vartheta = \pi_{wa} \leftarrow P_{wa}(\sigma_{wa}, (vk_{sig}, m), (ID_i, Sig_i))$	
$GS.Verify(gpk, RL, \vartheta, m)$	
Parse $gpk = (\sigma_{wa}, vk_{sig}), RL = \{ID_t\}_{t \in T \subset [n]}, \vartheta = \pi_{wa}$	
$ \textbf{if } V_{wa}(\sigma_{wa}, (vk_{sig}, m), \pi_{wa}) = 1 \land (\forall t \in T, I_{wa}(\sigma_{wa}, (vk_{sig}, m), \pi_{wa}, ID_t) = 0 ) $	
then return 1; else return 0	

Fig. 5. VLR group signature with auxiliary inputs

#### 6.3 Security Analysis

The correctness is easy to follow. Regarding the security, we first specify the admissible leakage function family  $\mathcal{F}$ . Assume the underlying waNIZK  $\Pi_{wa}$  satisfies the entropic ZK and the unforgeability w.r.t. all k-unpredictable samplers.

**Definition 24.** We say  $\mathcal{F}$  is admissible w.r.t.  $\Sigma_{gs}$ , if for every  $\sigma_{wa}$  in the range of  $S_{wa}(1^{\lambda})$ , and every  $(vk_{sig}, sk_{sig})$  in the range of  $K_{sig}(1^{\lambda})$ , it holds that

 $\mathbf{H}^{\mathsf{unp}}(ID|gpk = (\sigma_{\mathsf{wa}}, vk_{\mathsf{sig}}), f(gpk, \mathsf{S}_{\mathsf{sig}}(vk_{\mathsf{sig}}, sk_{\mathsf{sig}}, ID), ID)) \ge k(\lambda),$ 

where ID is a uniformly distributed random variable over  $\{0,1\}^{id(\lambda)}$  and id is an integer function polynomial in  $\lambda$ .

Note that the group public parameter gpk is independent of ID, and thus the admissible leakage function family  $\mathcal{F}$  is surely non-empty. Moreover, notice that the class of admissible leakage functions gets larger, if k is smaller.

**Theorem 3.** Let  $\Sigma_{sig}$  be a standard-model digital signature scheme satisfying EU-CMA security,  $\Pi_{wa}$  be a waNIZK for the language  $L_{VLR}$  that satisfies the authenticating knowledge soundness, the entropic ZK and unforgeability w.r.t all k-unpredictable samplers for  $L_{VLR}$ .  $\Sigma_{gs}$  is a secure VLR group signature scheme in terms of the auxiliary-input anonymity and the auxiliary-input traceability w.r.t. all admissible functions.

Before presenting the formal proof, we give the intuition. Recall that in the anonymity experiment, the goal of an adversary  $\mathcal{A}$  is to decide the signer's identity for a signature that was generated by an uncorrupted user. The main idea of our proof is to show such a signature can be obtained by querying the prover oracles of  $\Pi_{wa}$  with an unpredictable sampler or an admissible extended sampler. By the definition of entropic ZK, a signature by an uncorrupted secret key (which is a proof  $\pi_{wa}$ ) will not leak any useful information to adversaries beyond that its validity. The anonymity follows.

Regarding the auxiliary-input traceability, we note the adversary  $\mathcal{A}$  wins either (1) when (GS.Verify(gpk,  $\mathbf{grk}[i^*]$ ,

 $\vartheta^*, m^*) = 0$  for the target user  $i^*$ , or (2) when  $(\mathsf{GS}.\mathsf{Verify}(gpk, \mathbf{grk}, \vartheta^*, m^*) = 1$ . If  $\mathcal{A}$  wins via the first condition, we can show it contradicts the unforgeability of  $\Pi_{wa}$  by following similar arguments in the anonymity proof. For the second condition, the authenticating knowledge soundness of  $\Pi_{wa}$  ensures that for each valid proof  $\pi_{wa}$ , one can extract  $(ID, \mathsf{Sig})$  such that ID authenticates  $\pi_{wa}$ . Given the EU-CMA security of the digital signature scheme, the extracted ID must be one generated by  $\mathsf{GS}.\mathsf{KeyGen}$  and thus be contained in  $\mathsf{grk}$ , which contradicts the second condition.

#### Proof of anonymity.

**Lemma 5.** If the underlying WA-NIZK  $\Pi_{wa}$  satisfies the entropic ZK w.r.t. all k-unpredictable samplers, the proposed VLR group signature scheme  $\Sigma_{gs}$  satisfies the auxiliary-input anonymity w.r.t. the admissible leakge function family  $\mathcal{F}$  (cf. Def.24).

The hybrid event  $\text{Event}_{anon,\mathcal{A}}^{0}$ . This event is identical to that  $\text{Exp}_{\Sigma_{gs},\mathcal{A}}^{\text{Anon},0}(\lambda, n) = 1$ , *i.e.*,(without boxed items)

$$\operatorname{Event}_{\operatorname{anon},\mathcal{A}}^{0} = \begin{bmatrix} j_{0}, j_{1} \leftarrow \$ [n] \\ (gpk, \operatorname{\mathbf{gsk}}, \operatorname{\mathbf{grk}}) \leftarrow \operatorname{\mathsf{GS.KeyGen}}_{1}^{(\lambda)} \\ (i_{0}^{*}, i_{1}^{*}, m^{*}, st) \leftarrow \mathcal{A}_{1}^{\operatorname{Corr}, \operatorname{OSign}_{\operatorname{\mathbf{gsk}}}}(gpk, \overrightarrow{f}(gpk, \operatorname{\mathbf{gsk}})) \\ \vartheta^{*} \leftarrow \operatorname{Sign}(gpk, \operatorname{\mathbf{gsk}}[i_{0}^{*}], m^{*}) : \\ 1 \leftarrow \mathcal{A}_{2}^{\operatorname{Corr}, \operatorname{OSign}_{\operatorname{\mathbf{gsk}}}}(st, \vartheta^{*}) \boxed{\wedge (i_{0}^{*}, i_{1}^{*}) = (j_{0}, j_{1})} \end{bmatrix}$$

The bybrid event  $\text{Event}_{\text{anon},\mathcal{A}}^1$ . This event is almost identical to  $\text{Event}_{\text{anon},\mathcal{A}}^0$ , except the boxed items will be added. I.e., before executing key generation algorithm GS.KeyGen, it randomly samples two indexes  $j_0, j_1 \leftarrow [n]$ , and this event occurs under an extra condition that  $(i_0^*, i_1^*)$  selected by  $\mathcal{A}_1$  is equal to  $(j_0, j_1)$ .

The hybrid event  $\text{Event}_{anon,\mathcal{A}}^2$ . This event is almost identical to  $\text{Event}_{anon,\mathcal{A}}^1$ , except that the key generation algorithm GS.KeyGen is replaced by GS.KeyGen<sup>ev2</sup>, the oracle  $\text{OSign}_{gsk}$  is replaced by  $\text{OSign}_{gsk}^{ev2}$ , and the challenge signature  $\vartheta^*$  is generated in another way.

- GS.KeyGen<sup>ev2</sup>(1<sup> $\lambda$ </sup>). It is almost identical to GS.KeyGen(1<sup> $\lambda$ </sup>), except that the CRS  $\sigma_{wa}$  (which is a part of gpk) is generated by the simulator of  $\Pi_{wa}$ : ( $\sigma_{wa}, \tau_{wa}$ )  $\leftarrow$  SS<sub>wa</sub>(1<sup> $\lambda$ </sup>).
- $\operatorname{OSign}_{gsk}^{ev2}$ . It is almost identical to  $\operatorname{OSign}_{gsk}$ , except that for each query  $(j_0, m)$  or  $(j_1, m)$  where the signer identity is specified to be one of the pre-selected identities, the signature  $\vartheta$  is generated by running the simulator of  $\Pi_{wa}$ :  $\vartheta = \pi_{wa} \leftarrow \operatorname{SP}_{wa}(\sigma_{wa}, \tau_{wa}, (vk_{sig}, m))$ .
- $\text{ The challenge signature } \vartheta^* = \pi^*_{\mathsf{wa}} \leftarrow \mathsf{SP}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, \tau_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^*)).$

The bybrid event  $\text{Event}_{\text{anon},\mathcal{A}}^3$ . This is almost identical to  $\text{Event}_{\text{anon},\mathcal{A}}^1$ , except the challenge signature is generated by using  $\mathbf{gsk}[i_1^*]$ .

The bybrid event  $\text{Event}_{anon,\mathcal{A}}^4$ . This is identical to  $\text{Exp}_{\Sigma_{gs},\mathcal{A}}^{\text{Anon},1}(\lambda,n) = 1$ . We have the following claims.

 $Claim. \text{ For any non-uniform PPT adversary } \mathcal{A}, \text{ if holds that } \Pr[\texttt{Event}^0_{\texttt{anon},\mathcal{A}}] = n(n-1) \Pr[\texttt{Event}^1_{\texttt{anon},\mathcal{A}}].$ 

*Proof.* Note that the selection of  $\{j_0, j_1\}$  is uniform and independent of  $\mathcal{A}$ 's view. Therefore,

$$\frac{1}{n(n-1)} \Pr[\texttt{Event}^0_{\texttt{anon},\mathcal{A}}] = \Pr[\texttt{Event}^1_{\texttt{anon},\mathcal{A}}].$$

Claim. Assume that any adversary cannot break the entropic ZK of  $\Pi_{wa}$  with an advantage greater than  $\operatorname{Adv}_{\Pi_{wa}}^{\mathsf{zk}}$ . It follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$|\Pr[\mathsf{Event}^1_{\mathsf{anon},\mathcal{A}}] - \Pr[\mathsf{Event}^2_{\mathsf{anon},\mathcal{A}}]| \le \mathsf{Adv}^{\mathsf{zk}}_{\varPi_{\mathsf{wa}}}.$$

*Proof.* Assume there is an efficient adversary  $\mathcal{A}$  making  $|\Pr[\mathsf{Event}^1_{\mathsf{anon},\mathcal{A}}] -$ 

 $\Pr[\mathsf{Event}^2_{\mathsf{anon},\mathcal{A}}]| = \epsilon$ , and we proceed this proof by constructing a non-uniform PPT adversary  $\mathcal{B}$  that breaks the entropic ZK of  $\Pi_{wa}$  with the advantage  $\epsilon$ . Recall the definition of entropic ZK (cf. Def.15) where  $\mathcal{B}$  is given  $\sigma_{wa}$  and oracle accesses to either  $(\mathcal{O}_{P1}, \mathcal{O}_{P2})$  or  $(\mathcal{O}_{S1}, \mathcal{O}_{S2})$ , and its goal is to distinguish them.

Before presenting the code of  $\mathcal{B}$ , we specify the oracle queries it will make.

- To generate an uncorrupted user's secret key along with its leakage w.r.t. f, one can query the prover oracle  $\mathcal{O}_{P1}$  with the following sampler  $G(\sigma_{wa}, \cdot)$  (parameterized by  $(vk_{sig}, sk_{sig})$ ):

$$ID \leftarrow \$ \{0,1\}^{\mathsf{id}(\lambda)}; \mathsf{Sig} \leftarrow \mathsf{S}_{\mathsf{sig}}(vk_{\mathsf{sig}}, sk_{\mathsf{sig}}, ID); m \leftarrow 0; \mathbf{return}$$
$$(x = (vk_{\mathsf{sig}}, m), w^{I} = ID, w^{NI} = \mathsf{Sig}, z = f(\sigma_{\mathsf{wa}}, vk_{\mathsf{sig}}, ID, \mathsf{Sig})))$$
(3)

- To obtain a signature on m' from an uncorrputed secret key (ID, Sig), one can query the prover oracle  $\mathcal{O}_{P2}$  with a dual-mode extended sampler  $\mathsf{EG}^{m'} = \{\mathsf{EG}^{m'}_{\mathsf{real}}, \mathsf{EG}^{m'}_{\mathsf{sim}}\}$ , where  $\mathsf{EG}^{m'}_{\mathsf{real}}$  on input  $(\sigma_{\mathsf{wa}}, x = (vk_{\mathsf{sig}}, m), w^I = ID, w^{NI} = \mathsf{Sig})$ , and

$$\mathbf{return} \ (\bar{x} = (vk_{\mathsf{sig}}, m'), \bar{w}^{NI} = w^{NI}), \tag{4}$$

while  $\mathsf{EG}_{sim}^{m'}$  on input  $\sigma_{wa}, x = (vk_{sig}, m)$  outputs  $\bar{x} = (vk_{sig}, m')$ .

It is easy to verify that the above sampler is k-unpredictable (by the definition of admissible leakage function), and that the extended sampler is dual-mode (since ID is never used).

In the code of  $\mathcal{B}$ , we use  $(\mathcal{O}_1, \mathcal{O}_2)$  to denote either  $(\mathcal{O}_{P1}, \mathcal{O}_{P2})$  or  $(\mathcal{O}_{S1}, \mathcal{O}_{S2})$ , depending on which pair of oracles  $\mathcal{B}$  has accesses to.

 $\mathcal{B}^{\mathcal{O}_{\mathsf{P}1},\mathcal{O}_{\mathsf{P}2}}(\sigma_{\mathsf{wa}})$  and  $\mathcal{B}^{\mathcal{O}_{\mathsf{S}1},\mathcal{O}_{\mathsf{S}2}}(\sigma_{\mathsf{wa}})$  $\overline{j_0, j_1 \leftarrow s[n]; (vk_{sig}, sk_{sig})} \leftarrow \mathsf{K}_{sig}(1^{\lambda}); \text{ Set } gpk = (\sigma_{wa}, vk_{sig})$ For  $i \in [n] \setminus \{j_0, j_1\}, ID_i \leftarrow \{0, 1\}^{\mathsf{id}(\lambda)}, \mathsf{Sig}_i \leftarrow \mathsf{S}_{\mathsf{sig}}(vk_{\mathsf{sig}}, sk_{\mathsf{sig}}, ID_i)$ Query  $\mathcal{O}_1$  with the sampler  $G(\sigma_{wa}, \cdot)$  twice;  $Obtain((vk_{sig}, m), f(gpk, ID_{j_0}, Sig_{j_0}))$  and  $((vk_{sig}, m), f(gpk, ID_{j_1}, \mathsf{Sig}_{j_1}))$  $(i_0^*, i_1^*, m^*, st) \leftarrow \mathcal{A}_1^{(\cdot)}(gpk, \overrightarrow{f}(gpk, \mathbf{gsk}))$ **if**  $(j_0, j_1) \neq (i_0^*, i_1^*)$ , Abort Generate  $\vartheta^*$  by querying  $\mathcal{O}_2$  with  $\mathsf{EG}^{m^*}(\sigma_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m), ID_{j_0}, \mathsf{Sig}_{j_0}))$  $b \leftarrow \mathcal{A}_2^{(\cdot)}(st, \vartheta^*)$ Output b, Halt. Oracle queries of  $(A_1, A_2)$  are answered as below: - For a query (i, m') to  $OSign^{ev2}$  do if  $(i \neq j_0, j_1)$ , return  $\vartheta \leftarrow \mathsf{GS.Sign}(gpk, \mathbf{gsk}[i], m')$  to  $\mathcal{A}$ **if**  $(i = j_t, t = 0, 1)$ Query  $\mathcal{O}_{P2}$  with  $\mathsf{EG}^{m'}(\sigma_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m), ID_{j_t}, \mathsf{Sig}_{j_t}))$ ; Obtain $\pi_{\mathsf{wa}}$ ; return  $\vartheta = \pi_{wa}$  to  $\mathcal{A}$ - For a query (i) to Corr : if  $(i \neq j_0, j_1)$ , return  $(\mathbf{gsk}[\mathbf{i}], \mathbf{grk}[i])$ ; otherwise, abort

Notice that conditioned on  $(j_0, j_1) = (i_0^*, i_1^*)$ , and when  $\mathcal{B}$  is given accesses to  $(\mathcal{O}_{P1}, \mathcal{O}_{P2})$ , the environment of  $\mathcal{A}$  provided by  $\mathcal{B}$  is identical to that in  $\mathsf{Event}^1_{\mathsf{anon},\mathcal{A}}$ ; when when  $\mathcal{B}$  is given accesses to  $(\mathcal{O}_{S1}, \mathcal{O}_{S2})$ , the environment of  $\mathcal{A}$  provided by  $\mathcal{B}$  is identical to that in  $\mathsf{Event}^2_{\mathsf{anon},\mathcal{A}}$ . Therefore, it follows that

$$\begin{aligned} &|\Pr[\sigma_{\mathsf{wa}} \leftarrow \mathsf{S}_{\mathsf{wa}}(1^{\lambda}), : 1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma_{\mathsf{wa}})] - \\ &\Pr[(\sigma_{\mathsf{wa}}, \tau_{\mathsf{wa}}) \leftarrow \mathsf{SS}_{\mathsf{wa}}(1^{\lambda}) : 1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{S}1}, \mathcal{O}_{\mathsf{S}2}}(\sigma_{\mathsf{wa}})]| \\ &= |\Pr[\mathsf{Event}_{\mathsf{anon}, \mathcal{A}}^{1}] - \Pr[\mathsf{Event}_{\mathsf{anon}, \mathcal{A}}^{2}]| = \epsilon. \end{aligned}$$

Therefore, it holds that for any non-uniform PPT adversary  $\mathcal{A}$ ,

 $|\Pr[\mathsf{Event}^1_{\mathsf{anon},\mathcal{A}}] - \Pr[\mathsf{Event}^2_{\mathsf{anon},\mathcal{A}}]| \leq \mathsf{Adv}^{\mathsf{zk}}_{\varPi_{\mathsf{wa}}}.$ 

Similarly, we have the following results.

Claim. Assume that any adversary cannot break the entropic ZK of  $\Pi_{wa}$  with an advantage greater than  $\mathsf{Adv}_{\Pi_{wa}}^{\mathsf{zk}}$ . It follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$|\Pr[\mathsf{Event}^2_{\mathsf{anon},\mathcal{A}}] - \Pr[\mathsf{Event}^3_{\mathsf{anon},\mathcal{A}}]| \leq \mathsf{Adv}^{\mathsf{zk}}_{\varPi_{\mathsf{wa}}}.$$

Claim. For any non-uniform PPT adversary  $\mathcal{A}$ , if holds that  $\Pr[\texttt{Event}^3_{\texttt{anon},\mathcal{A}}] = \frac{1}{n(n-1)} \Pr[\texttt{Event}^4_{\texttt{anon},\mathcal{A}}]$ .

Combining all these results, we have that for any non-uniform adversary  $\mathcal{A}$ ,

$$\begin{split} &|\Pr[\mathsf{Ext}^{\mathsf{Anon},0}_{\Sigma_{\mathsf{gs}},\mathcal{A}}(\lambda,n)=1] - \Pr[\mathsf{Ext}^{\mathsf{Anon},1}_{\Sigma_{\mathsf{gs}},\mathcal{A}}(\lambda,n)=1] \\ &= |\Pr[\mathsf{Event}^0_{\mathsf{anon},\mathcal{A}}] - \Pr[\mathsf{Event}^4_{\mathsf{anon},\mathcal{A}}]| \\ &= n(n-1)|\Pr[\mathsf{Event}^1_{\mathsf{anon},\mathcal{A}}] - \Pr[\mathsf{Event}^3_{\mathsf{anon},\mathcal{A}}]| \\ &\leq 2n(n-1)\mathsf{Adv}^{\mathsf{zk}}_{U_{\mathsf{WR}}}, \end{split}$$

which is negligible in  $\lambda$  since we assume the entropic ZK of  $\Pi_{wa}$ .

#### Proof of traceability.

**Lemma 6.** Assume that the underlying WA-NIZK  $\Pi_{wa}$  satisfies the authenticating knowledge soundness and the unforgeability w.r.t. all k-unpredictable samplers, and the digital signature scheme  $\Sigma_{sig}$  satisfies the standard EU-CMA. The proposed VLR group signature scheme  $\Sigma_{gs}$  satisfies the auxiliary-input traceability w.r.t. the admissible leakage function family  $\mathcal{F}$  (cf. Def.24).

*Proof.* For clarity, we define the following events (without boxed items).

$$\begin{split} & \operatorname{Event}_{\operatorname{trace1},\mathcal{A}}^{0} = \begin{bmatrix} j \leftarrow \widehat{[n]} \\ (gpk, \operatorname{\mathbf{gsk}}, \operatorname{\mathbf{grk}}) \leftarrow \operatorname{\mathsf{GS}}.\operatorname{KeyGen}(1^{\lambda}, n) \\ (RL^{*}, \vartheta^{*}, m^{*}, i^{*}) \leftarrow \mathcal{A}^{\operatorname{Corr}, \operatorname{OSign}_{\operatorname{\mathbf{gsk}}}}(gpk, \overrightarrow{f}(gpk, \operatorname{\mathbf{gsk}})) : \\ & \operatorname{\mathsf{GS}}.\operatorname{Verify}(gpk, RL^{*}, \vartheta^{*}, m^{*}) = 1 \wedge \\ & \operatorname{\mathsf{GS}}.\operatorname{Verify}(gpk, \operatorname{\mathbf{grk}}[i^{*}], \vartheta^{*}, m^{*}) = 0, i^{*} \notin U[\wedge i^{*} = j] \end{bmatrix} . \end{split}$$

$$\begin{aligned} & \operatorname{Event}_{\operatorname{trace2,\mathcal{A}}}^{0} = \begin{bmatrix} (gpk, \operatorname{\mathbf{gsk}}, \operatorname{\mathbf{grk}}) \leftarrow \operatorname{\mathsf{GS}}.\operatorname{KeyGen}(1^{\lambda}, n) \\ & (RL^{*}, \vartheta^{*}, m^{*}, i^{*}) \leftarrow \mathcal{A}^{\operatorname{Corr}, \operatorname{OSign}_{\operatorname{\mathbf{gsk}}}}(gpk, \overrightarrow{f}(gpk, \operatorname{\mathbf{gsk}})) : \\ & \operatorname{\mathsf{GS}}.\operatorname{Verify}(gpk, RL^{*}, \vartheta^{*}, m^{*}) = 1 \wedge \\ & \operatorname{\mathsf{GS}}.\operatorname{Verify}(gpk, \operatorname{\mathbf{grk}}, \vartheta^{*}, m^{*}) = 1 \end{aligned}$$

It is easy to see  $\operatorname{Ext}_{\Sigma_{gs},\mathcal{A}}^{\operatorname{trace}}(\lambda,n) = 1$  iff either  $\operatorname{Event}_{\operatorname{trace}1,\mathcal{A}}^{0}$  or  $\operatorname{Event}_{\operatorname{trace}2,\mathcal{A}}^{0}$  happens. Thus, it is sufficient to prove  $\Pr[\operatorname{Event}_{\operatorname{trace}1,\mathcal{A}}^{0}] + \Pr[\operatorname{Event}_{\operatorname{trace}2,\mathcal{A}}^{0}] \leq \operatorname{negl}(\lambda)$ . In the following, we bound the probability of each event separately.

**PART I.**  $\Pr[\texttt{Event}^0_{\texttt{tracel},\mathcal{A}}] \leq \mathsf{negl}(\lambda)$ . We show this via the following hybrid event.

The hybrid event  $\text{Event}_{\text{trace1},\mathcal{A}}^1$ . This event is almost identical to  $\text{Event}_{\text{trace1},\mathcal{A}}^0$  except that the boxed items are included.

Then, we have the following claims.

Claim. For any non-uniform PPT adversary  $\mathcal{A}$ , if holds that  $\Pr[\texttt{Event}^0_{\mathsf{tracel},\mathcal{A}}] = n \Pr[\texttt{Event}^1_{\mathsf{tracel},\mathcal{A}}]$ .

*Proof.* Since the selection of j is uniformly random and independent of  $\mathcal{A}$ 's view,  $\Pr[\texttt{Event}^0_{\texttt{tracel},\mathcal{A}}] = n \Pr[\texttt{Event}^1_{\texttt{tracel},\mathcal{A}}].$ 

Claim. Assume that any adversary cannot break the unforgeability of  $\Pi_{wa}$  with an advantage greater than  $\mathsf{Adv}_{\Pi_{wa}}^{unf}(\lambda)$ . It follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,  $\Pr[\mathsf{Event}_{\mathsf{tracel},\mathcal{A}}^1] \leq \mathsf{Adv}_{\Pi_{wa}}^{unf}(\lambda)$ .

*Proof.* Assuming there is an adversary  $\mathcal{A}$  making  $\Pr[\text{Event}_{\text{tracel},\mathcal{A}}^1] = \epsilon$ , we proceed this proof by constructing an efficient adversary  $\mathcal{B}$  against the unforgeability of  $\Pi_{wa}$  with an advantage  $\epsilon$ . Recall the definition of unforgeability (cf. Def.18), where the adversary  $\mathcal{B}$  is given a CRS  $\sigma_{wa}$ , and oracle accesses to the prover oracles  $\mathcal{O}_{P1}$  and  $\mathcal{O}_{P2}$ . And the goal of  $\mathcal{B}$  is to produce a new statement

 $x^* \in L_{\mathsf{VLR}}$  (with a form of  $(vk_{\mathsf{sig}}, m)$ ) along with a valid proof  $\pi_{\mathsf{wa}}$  that will be authenticated by an honest generated  $w^I$  (which in our case is ID).

We state the code of  $\mathcal{B}$  in the following, by using the adversary  $\mathcal{A}$  in Event<sup>1</sup><sub>trace1, $\mathcal{A}$ </sub>, where  $\mathcal{B}$  queries the prover oracles with the k-unpredictable sampler G and the dual-mode extended sampler EG which are defined in Eq.3 and 4.

$\mathcal{B}^{\mathcal{O}_{P1},\mathcal{O}_{P2}}(\sigma_{wa})$			
$\overline{j \leftarrow s[n]; (vk_{sig}, sk_{sig}) \leftarrow K_{sig}(1^{\lambda}); \text{ Set } gpk = (\sigma_{wa}, vk_{sig})}$			
For $i \in [n] \setminus \{j\}, ID_i \leftarrow \{0, 1\}^{id(\lambda)}, Sig_i \leftarrow S_{sig}(vk_{sig}, sk_{sig}, ID_i)$			
Query $\mathcal{O}_1$ with the sampler $G(\sigma_{wa}, \cdot)$ ,			
$Obtain((vk_{sig}, m), f(gpk, ID_j, Sig_j))$			
$(RL^*, \vartheta^*, m^*, i^*) \leftarrow \mathcal{A}_1^{(\cdot)}(gpk, \overrightarrow{f}(gpk, \mathbf{gsk}))$			
<b>if</b> $j \neq i^*$ , Abort			
Output $x^* = (vk_{sig}, m^*), \pi^*_{wa} = \vartheta^*$ , Halt			
Oracle queries of $\mathcal{A}$ are answered as below:			
– For a query $(i, m')$ to $OSign^{ev2}$ do			
if $(i \neq j)$ , return $\vartheta \leftarrow GS.Sign(gpk, \mathbf{gsk}[i], m')$ to $\mathcal{A}$			
$\mathbf{if} \ (i=j)$			
Query $\mathcal{O}_{P2}$ with $EG^{m'}(\sigma_{wa}, (vk_{sig}, m), ID_j, Sig_j)$ ; Obtain $\pi_{wa}$ ;			
return $\vartheta = \pi_{wa}$ to $\mathcal{A}$			
- For a query (i) to Corr : if $(i \neq j)$ , return (gsk[i], grk[i]); otherwise, abort			

Note that conditioned on  $j = i^*$ , the environment around  $\mathcal{A}$  provided by  $\mathcal{B}$  is identical to that in  $\mathsf{Event}^1_{\mathsf{tracel},\mathcal{A}}$ . Therefore, with the probability  $\epsilon$ ,  $\mathcal{A}$  will output a tuple  $(RL^*, \vartheta^*, m^*, i^*)$  such that  $\mathsf{GS}.\mathsf{Verify}(gpk, RL^*, \vartheta^*, m^*) = 1$  and  $\mathsf{GS}.\mathsf{Verify}(gpk, \mathsf{grk}[i^*], \vartheta^*, m^*) = 0$ . By the description of our construction, it follows that  $\pi^*_{\mathsf{wa}} = \vartheta^*$  is a valid proof for  $x^* = (vk_{\mathsf{sig}}, m^*)$  and  $\mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, x^*, \pi^*_{\mathsf{wa}}, ID_{i^*}) = 1$ . In this case,  $\mathcal{B}$  breaks the unforgeability of  $\Pi_{\mathsf{wa}}$ . Therefore, we have  $\Pr[\mathsf{Event}^1_{\mathsf{tracel},\mathcal{A}}] \leq \mathsf{Adv}^{unf}_{\Pi_{\mathsf{wa}}}(\lambda)$ .

Combining the two claims, we know that for any non-uniform PPT adversary  $\mathcal{A}$ , it follows that

$$\Pr[\texttt{Event}^{0}_{\texttt{trace1},\mathcal{A}}] \leq n\mathsf{Adv}^{\mathsf{unf}}_{\Pi_{\mathsf{wa}}}(\lambda).$$

Since we assume the unforgeability of  $\Pi_{wa}$ , it holds that  $\Pr[\Pr[\texttt{Event}^0_{\texttt{trace1},\mathcal{A}}]] \leq \mathsf{negl}(\lambda)$ .  $\Box$ 

**Part II.**  $\Pr[\text{Event}_{\text{trace}2,\mathcal{A}}^0] \leq \text{negl}(\lambda)$ . We show this via the following hybrid events.

The hybrid event  $\text{Event}_{\text{trace2},\mathcal{A}}^1$ . This is almost identical to  $\text{Event}_{\text{trace2},\mathcal{A}}^0$ , except that the key generation algorithm GS.KeyGen is replaced by GS.KeyGen<sup>ev1</sup>:

- GS.KeyGen<sup>ev1</sup>(1<sup> $\lambda$ </sup>). It is almost identical to GS.KeyGen, except that the CRS  $\sigma_{wa}$  is generated by the extractor of  $\Pi_{wa}$ :  $(\sigma_{wa}, \xi_{wa}) \leftarrow \mathsf{Ext0}_{wa}(1^{\lambda})$ .

The hybrid event  $\text{Event}^2_{\text{trace2},\mathcal{A}}$ . This is almost identical to  $\text{Event}^1_{\text{trace2},\mathcal{A}}$ , except that this event occurs under an additional condition that a valid witness  $(ID^*, \text{Sig}^*)$  can be extracted from  $\vartheta^*$ .

Formally, we define it in the following (excluding the boxed items).

$$\operatorname{Event}_{\operatorname{trace2},\mathcal{A}}^{2} = \begin{cases} (gpk, \operatorname{\mathbf{gsk}}, \operatorname{\mathbf{grk}}; \xi_{\operatorname{wa}}) \leftarrow \operatorname{GS.\operatorname{KeyGen}^{\operatorname{ev1}}(1^{\lambda}, n) \\ (RL^{*}, \vartheta^{*}, m^{*}, i^{*}) \leftarrow \mathcal{A}^{\operatorname{Corr}, \operatorname{OSign}_{\operatorname{\mathbf{gsk}}}}(gpk, \overrightarrow{f}(gpk, \operatorname{\mathbf{gsk}})) \\ (ID^{*}, \operatorname{Sig}^{*}) \leftarrow \operatorname{Ext1}_{\operatorname{wa}}(\sigma_{\operatorname{wa}}, \xi_{\operatorname{wa}}, (vk_{\operatorname{sig}}, m^{*}), \pi_{\operatorname{wa}^{*}} = \vartheta^{*}) : \\ \operatorname{GS.\operatorname{Verify}}(gpk, RL^{*}, \vartheta^{*}, m^{*}) = 1 \land \\ \operatorname{GS.\operatorname{Verify}}(gpk, \operatorname{\mathbf{grk}}, \vartheta^{*}, m^{*}) = 1 \land \\ \operatorname{R_{VLR}}((vk_{\operatorname{sig}}, m^{*}), (ID^{*}, \operatorname{Sig}^{*})) = 1 \land \\ \operatorname{I_{wa}}(\sigma_{\operatorname{wa}}, (vk_{\operatorname{sig}}, m^{*}), \pi_{\operatorname{wa}}^{*}, ID^{*}) = 1 \land \\ \hline \exists i \in [n], ID_{i} = ID^{*} \end{cases}$$

The hybrid event  $\text{Event}_{\text{trace2},\mathcal{A}}^3$ . This is almost identical to  $\text{Event}_{\text{trace2},\mathcal{A}}^2$ , except that the boxed items are included.

We have the following claims.

Claim. Assume that for any non-uniform adversary PPT  $\mathcal{B}$ ,

$$|\Pr[\sigma_{\mathsf{wa}} \leftarrow \mathsf{S}_{\mathsf{wa}}(1^{\lambda}) : 1 \leftarrow \mathcal{B}(\sigma_{\mathsf{wa}})] - \Pr[(\sigma_{\mathsf{wa}}, \xi_{\mathsf{wa}}) \leftarrow \mathsf{Ext0}_{\mathsf{wa}}(1^{\lambda}) : 1 \leftarrow \mathcal{B}(\sigma_{\mathsf{wa}})]| \le \mathsf{Adv}_{\Pi_{\mathsf{wa}}}^{\mathsf{aks1}}(\lambda).$$

It follows that for any non-uniform adversary  $\mathcal{A}$ ,

$$|\Pr[\texttt{Event}^0_{\texttt{trace2},\mathcal{A}}] - \Pr[\texttt{Event}^1_{\texttt{trace2},\mathcal{A}}]| \le \mathsf{Adv}^{\mathsf{aks1}}_{\Pi_{\mathsf{wa}}}(\lambda).$$

*Proof.* This directly comes from the definitions of events.

Claim. Assume that any non-uniform PPT adversary  $\mathcal{B}$  cannot break the authenticating knowledge soundness of  $\Pi_{wa}$  with an advantage greater than  $\mathsf{Adv}_{\Pi_{wa}}^{\mathsf{aks2}}(\lambda)$ , or more precisely,

$$\Pr \begin{bmatrix} (\sigma_{\mathsf{wa}}, \xi_{\mathsf{wa}}) \leftarrow \mathsf{Ext0}_{\mathsf{wa}}(1^{\lambda}), (x^*, \pi^*_{\mathsf{wa}}) \leftarrow \mathcal{B}(\sigma_{\mathsf{wa}}) \\ (w^I = ID^*, w^{NI} = \mathsf{Sig}^*) \leftarrow \mathsf{Ext1}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, \xi_{\mathsf{wa}}, x^*, \pi^*_{\mathsf{wa}}) : \\ \mathsf{V}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, x^*, \pi^*_{\mathsf{wa}}) = 1 \land \\ [(w^I, w^{NI}) \notin R_{\mathsf{VLR}}(x^*) \lor \mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, x^*, \pi^*_{\mathsf{wa}}, w^I) \neq 1] \end{bmatrix} \le \mathsf{Adv}_{II_{\mathsf{wa}}}^{\mathsf{aks2}}(\lambda)$$

It follows that any non-uniform PPT adversary  $\mathcal{A}$ ,

$$|\Pr[\texttt{Event}^1_{\texttt{trace2},\mathcal{A}}] - \Pr[\texttt{Event}^2_{\texttt{trace2},\mathcal{A}}]| \le \mathsf{Adv}^{\mathsf{aks2}}_{\Pi_{\mathsf{wa}}}(\lambda).$$

*Proof.* For clarity, we define the following event.

$$\mathsf{BadE}_{\mathsf{trace2},\mathcal{A}} = \begin{cases} (gpk, \mathbf{gsk}, \mathbf{grk}; \xi_{\mathsf{wa}}) \leftarrow \mathsf{GS}.\mathsf{KeyGen}^{\mathsf{ev1}}(1^{\lambda}, n) \\ (RL^*, \vartheta^*, m^*, i^*) \leftarrow \mathcal{A}^{\mathsf{Corr}, \mathsf{OSign}_{\mathbf{gsk}}}(gpk, \overrightarrow{f}(gpk, \mathbf{gsk})) \\ (ID^*, \mathsf{Sig}^*) \leftarrow \mathsf{Ext1}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, \xi_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^*), \pi_{\mathsf{wa}^* = \vartheta^*}) : \\ \mathsf{GS}.\mathsf{Verify}(gpk, RL^*, \vartheta^*, m^*) = 1 \land \\ \mathsf{GS}.\mathsf{Verify}(gpk, \mathbf{grk}, \vartheta^*, m^*) = 1 \land \\ (R_{\mathsf{VLR}}((vk_{\mathsf{sig}}, m^*), (ID^*, \mathsf{Sig}^*)) \neq 1 \lor \\ \mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^*), \pi_{\mathsf{wa}}^*, ID^*) \neq 1) \end{cases}$$

By standard arguments, it follows

$$\Pr[\texttt{Event}^1_{\texttt{trace2},\mathcal{A}}] = \Pr[\texttt{Event}^2_{\texttt{trace2},\mathcal{A}}] + \Pr[\texttt{BadE}_{\texttt{trace2},\mathcal{A}}].$$

Next, we show  $\Pr[\text{BadE}_{\text{trace2},\mathcal{A}}] \leq \operatorname{Adv}_{\Pi_{wa}}^{aks2}(\lambda)$ . It is easy to construct an efficient adversary  $\mathcal{B}$  which on input  $\sigma_{wa}$  perfectly simulates the environment which is around  $\mathcal{A}$  in  $\operatorname{BadE}_{\operatorname{trace2},\mathcal{A}}$ . It is easy to see  $\mathcal{B}$  can obtain a valid proof  $\pi_{wa}^*$  for  $(vk_{sig}, m^*)$  such that the extracted

$$(ID^*, \mathsf{Sig}^*) \notin R_{\mathsf{VLR}}(vk_{\mathsf{sig}}, m^*) \lor \mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^*), \pi^*_{\mathsf{wa}}, ID^*) \neq 1.$$

Therefore,

$$|\Pr[\texttt{Event}^1_{\texttt{trace2},\mathcal{A}}] - \Pr[\texttt{Event}^2_{\texttt{trace2},\mathcal{A}}]| = \Pr[\texttt{BadE}_{\texttt{trace2},\mathcal{A}}] \leq \mathsf{Adv}^{\texttt{aks2}}_{\varPi_{\texttt{wa}}}(\lambda).$$

Claim. Assume that any non-uniform PPT adversary  $\mathcal{B}$  cannot break the EU-CMA of  $\Sigma_{sig}$  with an advantage greater than  $\mathsf{Adv}^{\mathsf{eucma}}_{\Sigma_{sig}}(\lambda)$ . It follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,  $|\Pr[\mathsf{Event}^2_{\mathsf{trace2},\mathcal{A}}] - \Pr[\mathsf{Event}^3_{\mathsf{trace2},\mathcal{A}}]| \leq \mathsf{Adv}^{\mathsf{eucma}}_{\Sigma_{sig}}(\lambda)$ .

*Proof.* For clarity, we define a "bad" event in the following.

$$\mathsf{BadE}^{2}_{\mathsf{trace2},\mathcal{A}} = \begin{bmatrix} (gpk, \mathbf{gsk}, \mathbf{grk}; \xi_{\mathsf{wa}}) \leftarrow \mathsf{GS}.\mathsf{KeyGen}^{\mathsf{ev1}}(1^{\lambda}, n) \\ (RL^{*}, \vartheta^{*}, m^{*}, i^{*}) \leftarrow \mathcal{A}^{\mathsf{Corr}, \mathsf{OSign}_{\mathsf{gsk}}}(gpk, \overrightarrow{f}(gpk, \mathbf{gsk})) \\ (ID^{*}, \mathsf{Sig}^{*}) \leftarrow \mathsf{Ext1}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, \xi_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^{*}), \pi_{\mathsf{wa}^{*}=\vartheta^{*}}) : \\ \mathsf{GS}.\mathsf{Verify}(gpk, RL^{*}, \vartheta^{*}, m^{*}) = 1 \land \\ \mathsf{GS}.\mathsf{Verify}(gpk, \mathsf{grk}, \vartheta^{*}, m^{*}) = 1 \land \\ R_{\mathsf{VLR}}((vk_{\mathsf{sig}}, m^{*}), (ID^{*}, \mathsf{Sig}^{*})) \neq 1 \land \\ \mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (vk_{\mathsf{sig}}, m^{*}), \pi^{*}_{\mathsf{wa}}, ID^{*}) \neq 1 \land \\ \forall i \in [n], ID_{i} \neq ID^{*} \end{bmatrix}$$

By the definition, we have  $\Pr[\text{Event}_{\text{trace2},\mathcal{A}}^2] = [\text{Event}_{\text{trace2},\mathcal{A}}^3] + \Pr[\text{BadE}_{\text{trace2},\mathcal{A}}^2]$ . Next, we show  $\Pr[\text{BadE}_{\text{trace2},\mathcal{A}}^2] \leq \text{Adv}_{\Sigma_{\text{sig}}}^{\text{eucma}}(\lambda)$ . Assume there exists an efficient adversary  $\mathcal{A}$  making  $\Pr[\text{BadE}_{\text{trace2},\mathcal{A}}^2] = \epsilon$ , and it is easy to construct an efficient adversary  $\mathcal{B}$  that breaks the EU-CMA of  $\Sigma_{\text{wa}}$  with the advantage  $\epsilon$ . More specifically,  $\mathcal{B}$  embeds the given verification key  $vk_{\text{sig}}$  into gpk, generates all  $\{ID_i\}_{i\in[n]}$  by himself, and obtains  $\{Sig_i\}_{i\in[n]}$  by quering the signing oracle w.r.t.  $vk_{\text{sig}}$ , which allows  $\mathcal{B}$  provides the simulation for  $\mathcal{A}$ . If  $ID^* \neq ID_i$  for all  $i \in [n]$ ,  $\mathcal{B}$  obtains a valid signature Sig\* on a new message  $ID^*$ , thus breaking the EU-CMA of  $\Sigma_{\text{sig}}$ .

Claim. It follows that  $\Pr[\texttt{Event}_{\texttt{trace2.4}}^3] = 0$ .

*Proof.* Note that the if  $ID^* = ID_i$  for some  $i \in [n]$ , the proof  $\pi^*_{wa} = \vartheta^*$  must be authenticated by  $ID_i$ . By the description of the verifier algorithm GS.Verify, it contradicts GS.Verify $(gpk, grk, \vartheta^*, m^*) = 1$ . Therefore, the event  $\texttt{Event}^3_{\mathsf{trace2},\mathcal{A}}$  will not happen.

Combing all these claims, it follows that for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$\Pr[\texttt{Event}^0_{\texttt{trace2},\mathcal{A}}] \leq \mathsf{Adv}^{\mathsf{aks1}}_{\varPi_{\mathsf{wa}}}(\lambda) + \mathsf{Adv}^{\mathsf{aks2}}_{\varPi_{\mathsf{wa}}}(\lambda) + \mathsf{Adv}^{\mathsf{eucma}}_{\varSigma_{\mathsf{sig}}}(\lambda).$$

Since we assume the authenticating knowledge soundness of  $\Pi_{wa}$  and the EU-CMA of  $\Sigma_{sig}$ , it holds that

$$\Pr[\texttt{Event}^0_{\texttt{trace2},\mathcal{A}}] \leq \mathsf{negl}(\lambda).$$

#### 7 Plaintext-checkable encryption in the standard model

Plaintext-checkable encryption (PCE) [17] is a public-key encryption primitive that allows us to search encrypted data with plaintext messages but still enables randomized encryption. Compared with deterministic public-key encryption (DPKE) [5], PCE aims to find a more fine-grained definition between the search functionality while preserving best possible security, particularly, it ensures two ciphertexts encrypting the same message are unlinkable (all partial information is still hidden when the plaintext is not known to the attacker). Moreover, it was also shown to be useful for group signatures with verifier-local revocation and backward unlinkability [12].

Existing constructions [17,37,36] are either relying on random oracles or only working for uniform message distributions. <sup>13</sup> In most scenarios, messages are from biased distributions. It is thus a natural question to consider PCE in the standard-model for non-uniform message distributions. <sup>14</sup> In this section, we answer this question and present a generic transformation from a PKE scheme to a PCE scheme, via a simple application of our waNIZK.

# 7.1 Definition

A PCE scheme enables everyone having a public key pk, a ciphertext c and a message m, to check whether m is the plaintext of c under pk. Formally, it consists of four algorithms: KeyGen, Enc, Dec, PCheck. While the first three algorithms describe a standard PKE scheme, the last algorithm is as follows:

-  $\mathsf{PCheck}(pk, c, m)$ . Outputs 1 indicating c is an encryption of m under pk, and 0 otherwise.

Correctness requires that for every  $\lambda$  and m,  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda}), c \leftarrow \mathsf{Enc}(pk, m), \Pr[\mathsf{Dec}(sk, c) = 1 \land \mathsf{PCheck}(pk, c, m) = 1] = 1$ , where the probability is taken over coin tosses of KeyGen and Enc. We follow the definitions from [17]: *Checking completeness:* No efficient adversary can output a ciphertext which decrypts to a message that is refused by PCheck. Formally, for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\Pr[(pk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda}); c \leftarrow \mathcal{A}(pk) : \mathsf{PCheck}(pk, c, \mathsf{Dec}(c, sk)) \neq 1] \le \mathsf{negl}(\lambda).$$
(5)

<sup>&</sup>lt;sup>13</sup> We note that the recent scheme [36] claimed security in the standard model for any high-entropy message distribution. However, their proofs still implicitly assume that the message distribution is uniform. We defer details to Appendix B.

<sup>&</sup>lt;sup>14</sup> The plain-text equality tester, presented in [42], seems close to a PCE. However, it can only check whether a ciphertext encrypts a pre-chosen target value  $m^*$ , while a PCE allows us to test for any plaintext publicly.

Checking soundness: No efficient adversary can generate a ciphertext c and a plaintext m such that c cannot be decrypted to m but  $\mathsf{PCheck}(pk, c, m) = 1$ . Formally, for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$\Pr[(pk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda}); (c, m') \leftarrow \mathcal{A}(pk) :$$
  
$$\mathsf{PCheck}(pk, c, m) = 1 \land \mathsf{Dec}(sk, c) \neq m'] \le \mathsf{negl}(\lambda).$$
(6)

Unlinkability: This new property captures the infeasibility of deciding whether two ciphertexts encrypt the same message, when the message is absent. Formally, we say a PCE scheme  $\Sigma$  satisfies kunlinkability, if for any non-uniform PPT adversary  $\mathcal{A} = (\mathcal{A}_f, \mathcal{A}_g)$ , it holds that  $|\Pr[\mathsf{Exp}_{\Sigma,\mathcal{A}}^{\mathsf{unlink},0}(\mathcal{A}) = 1] - \Pr[\mathsf{Exp}_{\Sigma,\mathcal{A}}^{\mathsf{unlink},1}(\mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda)$ , where the experiment is defined below and the min-entropy of the output of  $\mathcal{A}_f$  should be greater than k.

$Exp^{unlink,b}_{\varSigma,\mathcal{A}}(\mathcal{A})$		
$(pk, sk) \leftarrow KeyGen(1^{\lambda}); m_0 \leftarrow \mathcal{A}_f(pk), m_1 \leftarrow \mathcal{A}_f(pk)$		
$c_0 \leftarrow Enc(pk, m_b); c_1 \leftarrow Enc(pk, m_1); b' \leftarrow \mathcal{A}_g(pk, c_0, c_1)$		
$\mathbf{return} \ b'$		

#### 7.2 Construction

As a PCE scheme is a special PKE scheme that supports the plaintext-checking functionality while preserving the best-possible privacy, the idea behind our transformation is to attach a waNIZK proof that demonstrates the underlying PKE ciphertext is well-formed. More precisely, let  $\Sigma_{pke} = \{K_{pke}, E_{pke}, D_{pke}\}$  be a PKE scheme, and let  $\Pi_{wa} = \{S_{wa}, P_{wa}, V_{wa}, I_{wa}\}$  be a waNIZK for the following language:  $L_{PCE} := \{(c, pk); (m, r) : c = E_{pke}(pk, m; r)\}$  where the message m is the identifier witness. To encrypt a message m, our PCE scheme first encrypts it using  $\Sigma_{pke}$ , and uses  $\Pi_{wa}$  to prove the ciphertext is well-formed, where the CRS for  $\Pi_{wa}$  is a part of the public key. Everyone can check whether a ciphertext ( $c_{pke}, \pi_{wa}$ ) encrypts a particular message m by running the identification algorithm  $I_{wa}$  on  $\pi_{wa}$  and m. The formal construction is presented in  $\Sigma = \{\text{KeyGen}, \text{Enc}, \text{Dec}, \text{PCheck}\}$  in Fig. 6

$KeyGen(1^\lambda)$	Dec(sk,c)
$ \begin{aligned} \overline{\sigma \leftarrow S_{wa}(1^{\lambda})} \\ (pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda}) \\ \mathbf{return} \ pk = (\sigma, pk_{pke}), sk = (pk_{pke}, sk_{pke}) \end{aligned} $	$\label{eq:star} \hline \mathbf{if } V_{wa}(\sigma, (c_{pke}, pk_{pke}), \pi) \mathbf{then} \\ \mathbf{return} \ m \leftarrow D_{pke}(c_{pke}, sk_{pke}) \\ \mathbf{else \ return} \ \bot$
Enc(pk,m)	PCheck(pk,c,m)
$ \begin{aligned} \overline{c_{pke} \leftarrow E_{pke}(pk_{pke}, m; r)} \\ \pi_{wa} \leftarrow P_{wa}(\sigma, (c_{pke}, pk_{pke}), (m, r)) \\ \mathbf{return} \ c = (c_{pke}, \pi_{wa}) \end{aligned} $	$\overline{b \leftarrow I_{wa}(\sigma, (c_{pke}, pk_{pke}), \pi_{wa}, m)}$ return b

Fig. 6. PCE from PKE+WA-NIZK

#### 7.3 Security Analysis

The correctness follows the correctness of the underlying primitives. Regarding the security, we establish the following result.

**Theorem 4.** The PCE scheme  $\Sigma$  satisfies checking completeness, checking soundness, and kunlinkability, if  $\Sigma_{pke}$  is an IND-CPA PKE scheme with perfect correctness, and the waNIZK  $\Pi_{wa}$ satisfies the entropic ZK w.r.t. all k-unpredictable samplers, the authenticating soundness, and the identifier uniqueness.

Before presenting the formal proof, we provide the intuition below. The checking completeness follows the authentiation soundness of  $\Pi_{wa}$ , and the checking soundness is implied by the identifier uniqueness of  $\Pi_{wa}$ . Regarding the k-unlinkability, we argue the distribution  $G = \{(x = (c, pk), w^I = m, \bot) : m \leftarrow M_{\lambda}; c \leftarrow \mathsf{Enc}(pk, m)\}$  for  $L_{\mathsf{PCE}}$  is k-unpredictable w.r.t. an honest key generation  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda})$ , if the min-entropy of  $M_{\lambda}$  is greater than k. We note given (c, pk) finding  $w^I$ is not necessarily  $2^{-k}$ -hard. Indeed, we can define the following distribution  $\overline{G} = \{(x = (c, pk), y, \bot) : m, y \leftarrow M_{\lambda}; c \leftarrow \mathsf{Enc}(pk, m)\}$ . Ensured by the IND-CPA security of the PKE scheme,  $\overline{G}$  is indistinguishable with G. As no side information about y is given, the probability of guessing y should be not greater than  $2^{-k}$ . According to our definition k-unpredictable distributions, G is such a distribution, enabling us to deploy a waNIZK that satisfies the entropic ZK w.r.t. k-unpredictable samplers. The above argument helps us to avoid requiring the sub-exponential hardness of the underlying PKE scheme.

**Proof of checking completeness.** We prove this property by contradiction. Assuming there is a successful adversary  $\mathcal{A}$  breaking the checking completeness, we can construct an efficient adversary  $\mathcal{B}$  breaking the authenticating soundness of  $\Pi_{wa}$  by invoking  $\mathcal{A}$  as follows. Recall that  $\mathcal{B}$  takes as input a CRS  $\sigma_{wa}$  and tries to output  $(x, \pi)$  such that  $\pi$  will not be authenticated by any  $w^{I} \in R_{L}x$ 

 $\begin{aligned} & \frac{\text{Algorithm } \mathcal{B}(\sigma_{\mathsf{wa}})}{(pk_{\mathsf{pke}}, sk_{\mathsf{pke}}) \leftarrow K_{\mathsf{pke}}(1^{\lambda}); \text{ Set } pk = (\sigma, pk_{\mathsf{pke}}); c \leftarrow \mathcal{A}(pk); \text{ Parse } c = (c_{\mathsf{pke}}, \pi_{\mathsf{wa}}) \\ & \text{return } (x = (pk_{\mathsf{pke}}, c_{\mathsf{pke}}), \pi_{\mathsf{wa}}) \end{aligned}$ 

Notice that ensured by the perfect correctness, for a valid  $pk_{pke}$  the message  $m = Dec(sk_{pke}, c_{pke})$ is the only identifier witness of  $(pk_{pke}, c_{pke})$ . Therefore, if  $I_{wa}(\sigma_{wa}, x, \pi_{wa}, m) = 0$ ,  $\mathcal{B}$  breaks the authenticating soundness.

**Proof of checking soundness.** Assuming there is a successful adversary  $\mathcal{A}$  breaking the checking soundness, we can construct an efficient adversary  $\mathcal{B}$  breaking the identifier uniqueness of  $\Pi_{wa}$  by invoking  $\mathcal{A}$  as follows. Recall that  $\mathcal{B}$  takes as input a CRS  $\sigma_{wa}$  and tries to output  $(x, \pi, w_1^I, w_2^I)$  such that  $\pi$  will be authenticated by both  $w_1^I$  and  $w_2^I$ .

$$\begin{split} & \frac{\mathcal{B}(\sigma_{\mathsf{Wa}})}{(pk_{\mathsf{pke}}, sk_{\mathsf{pke}}) \leftarrow K_{\mathsf{pke}}(1^{\lambda}); \text{ Set } pk = (\sigma, pk_{\mathsf{pke}}); (c, m') \leftarrow \mathcal{A}(pk)} \\ & \text{Parse } c = (c_{\mathsf{pke}}, \pi_{\mathsf{wa}}), m = \mathsf{D}_{\mathsf{pke}}(sk_{\mathsf{pke}}, c); \\ & \text{return } (x = (pk_{\mathsf{pke}}, c_{\mathsf{pke}}), \pi_{\mathsf{wa}}, w_1^I = m, w_2^I = m') \end{split}$$

Ensured by the checking completeness m authenticates the proof  $\pi_{wa}$ ; since  $\mathcal{A}$  is successful, m' can also authenticate the proof  $\pi_{wa}$ . Therefore,  $\mathcal{B}$  breaks the identifier uniqueness.

# **Proof of** $\epsilon$ **-unlinkability.** We establish a sequence fo experiments {Exp1, Exp2,

Exp3, Exp4} in Fig.7 where the first one is  $\text{Exp}_{\Sigma,\mathcal{A}}^{\text{unlink},0}$  and the last one is  $\text{Exp}_{\Sigma,\mathcal{A}}^{\text{unlink},1}$ . We prove this property by showing  $|\Pr[\text{Exp}(\lambda) = 1] - \Pr[\text{Exp}(i+1)(\lambda) = 1]| \leq \text{negl}(\lambda)$ . Here we denote the simulator of  $\Pi_{wa}$  by  $(SS_{wa}, SP_{wa})$ .

$Exp1(\lambda)$	$Exp3(\lambda)$
$\boxed{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})}$	$(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})$
$\sigma_{wa} \gets S_{wa}(1^{\lambda})$	$(\sigma_{wa},\tau_{wa}) \leftarrow SS_{wa}(1^{\lambda})$
$m_0 \leftarrow \mathcal{A}_f(pk_{pke},\sigma_{wa})$	$m_0 \leftarrow \mathcal{A}_f(pk_{pke},\sigma_{wa})$
$m_1 \leftarrow \mathcal{A}_f(pk_{pke},\sigma_{wa})$	$m_1 \leftarrow \mathcal{A}_f(pk_{pke},\sigma_{wa})$
$c_{pke,0} \leftarrow E_{pke}(pk_{pke},m_0;r_0)$	$c_{pke,0} \leftarrow E_{pke}(pk_{pke},m_1;r_0)$
$c_{pke,1} \gets E_{pke}(pk_{pke}, m_1; r_1)$	$c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1)$
$\pi_{wa,0} \leftarrow P_{wa}(\sigma_{wa},(pk_{pke},c_{pke,0}),(m_0,r_0))$	$\pi_{wa,0} \leftarrow SP_{wa}(\sigma_{wa},\tau_{wa},(pk_{pke},c_{pke,0}))$
$\pi_{wa,1} \leftarrow P_{wa}(\sigma_{wa},(pk_{pke},c_{pke,1}),(m_1,r_1))$	$\pi_{wa,1} \leftarrow SP_{wa}(\sigma_{wa},\tau_{wa},(pk_{pke},c_{pke,1}))$
$b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}),$	$b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}),$
$(c_{pke,1},\pi_{wa,1}))$	$(c_{pke,1},\pi_{wa,1}))$
$\mathbf{return} \ b'$	$\mathbf{return} \ b'$
$Exp2(\lambda)$	$Exp4(\lambda)$
$\frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})}$	$\frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})}$
$ \begin{aligned} \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \end{aligned} $	$\begin{aligned} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \end{aligned}$
$\begin{aligned} \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \end{aligned}$	$\begin{aligned} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \end{aligned}$
$\begin{aligned} \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \end{aligned}$	$\begin{aligned} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \end{aligned}$
$\begin{aligned} & \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \end{aligned}$	$\begin{aligned} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \end{aligned}$
$\begin{aligned} & \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \end{aligned}$	$\begin{split} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \end{split}$
$\begin{aligned} & \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,0})) \end{aligned}$	$\begin{split} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,0}), (m_1, r_0)) \end{split}$
$\begin{split} & \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,0})) \\ & \pi_{wa,1} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,1})) \end{split}$	$\begin{split} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,0}), (m_1, r_0)) \\ & \pi_{wa,1} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,1}), (m_1, r_1)) \end{split}$
$\begin{split} & \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, m_0; r_0) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,0})) \\ & \pi_{wa,1} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,1})) \\ & b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}), \end{split}$	$\begin{split} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,0}), (m_1, r_0)) \\ & \pi_{wa,1} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,1}), (m_1, r_1)) \\ & b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}), \end{split}$
$\begin{split} \frac{Exp2(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ (\sigma_{wa}, \tau_{wa}) \leftarrow SS_{wa}(1^{\lambda}) \\ m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_0; r_0) \\ c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ \pi_{wa,0} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,0})) \\ \pi_{wa,1} \leftarrow SP_{wa}(\sigma_{wa}, \tau_{wa}, (pk_{pke}, c_{pke,1})) \\ b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}), (c_{pke,1}, \pi_{wa,1})) \end{split}$	$\begin{split} & \frac{Exp4(\lambda)}{(pk_{pke}, sk_{pke}) \leftarrow K_{pke}(1^{\lambda})} \\ & \sigma_{wa} \leftarrow S_{wa}(1^{\lambda}) \\ & m_0 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & m_1 \leftarrow \mathcal{A}_f(pk_{pke}, \sigma_{wa}) \\ & c_{pke,0} \leftarrow E_{pke}(pk_{pke}, m_1; r_0) \\ & c_{pke,1} \leftarrow E_{pke}(pk_{pke}, m_1; r_1) \\ & \pi_{wa,0} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,0}), (m_1, r_0)) \\ & \pi_{wa,1} \leftarrow P_{wa}(\sigma_{wa}, (pk_{pke}, c_{pke,1}), (m_1, r_1)) \\ & b' \leftarrow \mathcal{A}_g((pk_{pke}, \sigma_{wa}), (c_{pke,0}, \pi_{wa,0}), \\ & (c_{pke,1}, \pi_{wa,1})) \end{split}$

Fig. 7. Experiments of unlinkability

We first show

$$\Pr[\mathsf{Exp1}(\lambda) = 1] - \Pr[\mathsf{Exp2}(\lambda) = 1]| \le \mathsf{negl}(\lambda), \tag{7}$$

assuming the entropic ZK of  $\Pi_{wa}$  and the semantic security of  $\Sigma_{pke}$ . Particularly, let  $K_{pke}$  be the trusted parameter generation procedure PG, and let  $G_{\mathcal{A}_f}(pk_{pke}, \sigma_{wa})$  be a sampler that works as follows:

$$\begin{split} m \leftarrow \mathcal{A}_f(pk_{\mathsf{pke}}, \sigma_{\mathsf{wa}}), r_{\mathsf{pke}} & \leftarrow \$ \; R_{\mathsf{pke}}, c_{\mathsf{pke}} = \mathsf{E}_{\mathsf{pke}}(pk_{\mathsf{pke}}, m; r_{\mathsf{pke}}); \\ \mathbf{return} \; (x = (pk_{\mathsf{pke}}, c_{\mathsf{pke}}), w^I = m, w^{NI} = r_{\mathsf{pke}}) \end{split}$$

Since  $\Sigma_{\mathsf{pke}}$  is semantic secure and the min-entropy of  $\mathcal{A}_f$  output is larger than  $-\log(\epsilon)$ , the sampler  $G_{\mathcal{A}_f}$  is  $\epsilon$ -unpredictable w.r.t.  $\mathsf{K}_{\mathsf{pke}}$ . Then, we construct an adversary  $\mathcal{B}_1$  that breaks the entropic ZK property of  $\Pi_{\mathsf{wa}}$  as follows. Recall that  $\mathcal{B}_1$  is a given a CRS  $\sigma_{\mathsf{wa}}$ , a honestly generated public key  $pk_{\mathsf{pke}}$ , and a pair of oracles ( $\mathcal{O}1, \mathcal{O}2$ ), and the goal of  $\mathcal{B}_1$  is to distinguish between the case that  $\sigma_{\mathsf{wa}}$  is a simulated CRS and ( $\mathcal{O}1, \mathcal{O}2$ ) are simulation oracles, and the case that  $\sigma_{\mathsf{wa}}$  is an honest CRS and ( $\mathcal{O}1, \mathcal{O}2$ ) are real prover oracles.

$\mathcal{B}_1^{\mathcal{O}1,\mathcal{O}2}(\sigma_{wa},pk_{pke})$		
Query $\mathcal{O}1$ with $G_{\mathcal{A}}(\sigma_{wa}, pk_{pke})$ twice		
Obtain $((pk_{pke}, c_{pke,0}), \pi_{wa,0})$ and $((pk_{pke}, c_{pke,1}), \pi_{wa,1})$		
$b' \leftarrow \mathcal{A}_g(pk_{pke}, (c_{pke,0}, \pi_{wa,0}), (c_{pke,1}, \pi_{wa,1}));  \mathbf{return} \ b'$		

Notice that when  $\sigma_{wa}$  is an honest CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are real prover oracles, at  $\mathcal{A}_g$ 's view, the experiment established by  $\mathcal{B}$  is identical to Exp1; when  $\sigma_{wa}$  is a simulated CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are simulation oracles, at  $\mathcal{A}_g$ 's view, the experiment established by  $\mathcal{B}$  is identical to Exp2. Therefore, if  $(\mathcal{A}_f, \mathcal{A}_g)$  can distinguish the two experiments,  $\mathcal{B}_1$  can thus distinguish between the case that  $\sigma_{wa}$  is a simulated CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are simulation oracles, and the case that  $\sigma_{wa}$  is an honest CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are real prover oracles.

Next, we show

$$|\Pr[\mathsf{Exp}2(\lambda) = 1] - \Pr[\mathsf{Exp}3(\lambda) = 1]| \le \mathsf{negl}(\lambda), \tag{8}$$

if  $\Sigma_{pke}$  satisfies the semantic security. We can construction an adversary  $\mathcal{B}_2$  that breaks the semantic security of  $\Sigma_{pke}$ . Specifically,  $\mathcal{B}_2$  is given  $pk_{pe}$  and a left-or-right oracle  $\mathsf{OEnc}_b$ , and her goal is to distinguish between  $\mathsf{OEnc}_0$  and  $\mathsf{OEnc}_1$ . The strategy of  $\mathcal{B}_2$  is as follows: generate the simulated CRS  $\sigma_{wa}$  along with its trapdoor  $\tau_{wa}$ ; invoke  $\mathcal{A}_f$  by providing  $(pk_{pke}, \sigma_{wa})$  twice to obtain  $m_0$  as well as  $m_1$ ; encrypt  $m_1$  under  $pk_{pke}$  and obtain the ciphertexts  $c_{pke,1}$ ; submit  $(m_0, m_1)$  to  $\mathsf{OEnc}_b$  and obtain  $c_{pke}, 0$ ; attach simulated proofs  $\pi_{wa,0}$  and  $\pi_{wa,1}$  for  $c_{pke,0}$  and  $c_{pke,1}$  respectively; finally invoke  $\mathcal{A}_g$ by providing  $(\pi_{wa,0}, c_{pke,0})$  and  $(\pi_{wa,1}, c_{pke,1})$  to output the bit b'. When b = 0, at  $\mathcal{A}_g$ 's view, the experiment is identical to  $\mathsf{Exp2}$ ; when b = 1, the experiment is identical to  $\mathsf{Exp3}$ . Therefore,  $\mathcal{B}_2$  can distinguish distinguish between  $\mathsf{OEnc}_0$  and  $\mathsf{OEnc}_1$ .

Lastly, we need to show

$$|\Pr[\mathsf{Exp3}(\lambda) = 1] - \Pr[\mathsf{Exp4}(\lambda) = 1]| \le \mathsf{negl}(\lambda).$$
(9)

To prove this result, we only need to shift from simulated proofs to real proofs. We can construct an adversary  $\mathcal{B}_3$  that breaks the entropic ZK of  $\Pi_{wa}$ . Similar to  $\mathcal{B}_1$ ,  $\mathcal{B}_3$  is also given a given a CRS  $\sigma_{wa}$ , a honestly generated public key  $pk_{pke}$ , and a pair of oracles ( $\mathcal{O}1, \mathcal{O}2$ ). Particularly, to obtain proofs for the ciphertexts with the same ciphertext, we define a reusing query RE which on inputs  $(pk_{pke}, \sigma_{wa}, c_{pke,0}, m_0, r_0)$  outputs  $(c_{pke,1} = \mathsf{P}_{pke}(pk_{pke}, m_0; r_1), m_0, r_1)$  where  $r_1$  is uniformly chosen at random. Ensured by the semantic security of  $\Sigma_{pke}$ , RE is an admissible query.

$\mathcal{B}_1^{\mathcal{O}1,\mathcal{O}2}(\sigma_{wa},pk_{pke})$	
Query $\mathcal{O}1$ with $G_{\mathcal{A}}(\sigma_{wa}, pk_{pke})$ once	
Obtain $((pk_{pke}, c_{pke,0}), \pi_{wa,0})$	
Query $O2$ with RE	
Obtain $((pk_{pke}, c_{pke,1}), \pi_{wa,1})$	
$b' \leftarrow \mathcal{A}_g(pk_{pke}, (c_{pke,0}, \pi_{wa,0}), (c_{pke,1}, \pi_{wa,1}));$	$\mathbf{return} \ b'$

otice that when  $\sigma_{wa}$  is an honest CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are real prover oracles, at  $\mathcal{A}_g$ 's view, the experiment established by  $\mathcal{B}$  is identical to Exp4; when  $\sigma_{wa}$  is a simulated CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are simulation oracles, at  $\mathcal{A}_g$ 's view, the experiment established by  $\mathcal{B}$  is identical to Exp3. Therefore, if  $(\mathcal{A}_f, \mathcal{A}_g)$  can distinguish the two experiments,  $\mathcal{B}_1$  can thus distinguish between the case that  $\sigma_{wa}$  is a simulated CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are simulation oracles, and the case that  $\sigma_{wa}$  is an honest CRS and  $(\mathcal{O}1, \mathcal{O}2)$  are real prover oracles.

From Eq.7 8, 9, it holds that

$$\Pr[\mathsf{Exp1}(\lambda) = 1] - \Pr[\mathsf{Exp4}(\lambda) = 1]| \le \mathsf{negl}(\lambda).$$

# 8 Conclusions and Open Problems

We abstracted a new notion called witness-authenticating NIZK proof system to add an identification functionality to conventional NIZK while preserving the best possible privacy, as well as upgrading soundness properties correspondingly. We gave careful modelings and generic constructions paired with rigorous security analysis. From several examples, we show how our WA-NIZK proof system can be useful in different applications.

We believe there could be many more applications to explore, such as revocation, tracing, repudiation in group (anonymous) authentications, (non) membership proofs, and even realizing random oracles. On the other hand, we may also make the identification functionality more complex to enable a more fine-grained search without leaking extra partial information. We leave them as interesting open questions.

Acknowledgement. This work has been supported in part by research awards from Stellar Development Foundation, Ethereum Foundation, Protocol Labs, as well as SOAR Prize and Digital Science Initiative Pilot Project of The University of Sydney.

#### References

- Abdalla, M., Benhamouda, F., Passelègue, A., Paterson, K.G.: Related-key security for pseudorandom functions beyond the linear barrier. J. Cryptol. 31(4), 917–964 (2018)
- Alamélou, Q., Blazy, O., Cauchie, S., Gaborit, P.: A code-based group signature scheme. Des. Codes Cryptogr. 82(1-2), 469–493 (2017)
- Baecher, P., Fischlin, M., Schröder, D.: Expedient non-malleability notions for hash functions. In: CT-RSA. LNCS, vol. 6558, pp. 268–283. Springer (2011)
- Barak, B., Dodis, Y., Krawczyk, H., Pereira, O., Pietrzak, K., Standaert, F., Yu, Y.: Leftover hash lemma, revisited. In: CRYPTO. LNCS, vol. 6841, pp. 1–20. Springer (2011)
- Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: CRYPTO. LNCS, vol. 4622, pp. 535–552. Springer (2007)
- Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS. pp. 62–73. ACM (1993)
- Bellare, M., Stepanovs, I.: Point-function obfuscation: A framework and generic constructions. In: TCC (A2). LNCS, vol. 9563, pp. 565–594. Springer (2016)
- Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: STOC. pp. 103–112. ACM (1988)
- Boldyreva, A., Cash, D., Fischlin, M., Warinschi, B.: Foundations of non-malleable hash and one-way functions. In: ASIACRYPT. LNCS, vol. 5912, pp. 524–541. Springer (2009)

- Boldyreva, A., Fischlin, M.: On the security of OAEP. In: ASIACRYPT. LNCS, vol. 4284, pp. 210–225. Springer (2006)
- Boneh, D., Eskandarian, S., Fisch, B.: Post-quantum EPID signatures from symmetric primitives. In: CT-RSA. LNCS, vol. 11405, pp. 251–271. Springer (2019)
- 12. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: CCS. pp. 168–177. ACM (2004)
- Brakerski, Z., Segev, G.: Better security for deterministic public-key encryption: The auxiliary-input setting. In: CRYPTO. LNCS, vol. 6841, pp. 543–560. Springer (2011)
- 14. Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: CCS. pp. 132–145. ACM (2004)
- Brickell, E., Li, J.: Enhanced privacy ID from bilinear pairing for hardware authentication and attestation. Int. J. Inf. Priv. Secur. Integr. 1(1), 3–33 (2011)
- Brzuska, C., Mittelbach, A.: Indistinguishability obfuscation versus multi-bit point obfuscation with auxiliary input. In: ASIACRYPT (2). LNCS, vol. 8874, pp. 142–161. Springer (2014)
- Canard, S., Fuchsbauer, G., Gouget, A., Laguillaumie, F.: Plaintext-checkable encryption. In: CT-RSA. LNCS, vol. 7178, pp. 332–348. Springer (2012)
- Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: CRYPTO. LNCS, vol. 1294, pp. 455–469. Springer (1997)
- Canetti, R., Micciancio, D., Reingold, O.: Perfectly one-way probabilistic hash functions (preliminary version). In: STOC. pp. 131–140. ACM (1998)
- Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: CRYPTO. LNCS, vol. 4117, pp. 78–96. Springer (2006)
- Chaum, D., van Heyst, E.: Group signatures. In: EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer (1991)
- Chen, Y., Qin, B., Zhang, J., Deng, Y., Chow, S.S.M.: Non-malleable functions and their applications. In: PKC (2). LNCS, vol. 9615, pp. 386–416. Springer (2016)
- Dachman-Soled, D., Gennaro, R., Krawczyk, H., Malkin, T.: Computational extractors and pseudorandomness. In: TCC. LNCS, vol. 7194, pp. 383–403. Springer (2012)
- 24. Dodis, Y.: On extractors, error-correction and hiding all partial information. In: ICITS. pp. 74–79. IEEE (2005)
- 25. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: STOC. pp. 621–630. ACM (2009)
- Faust, S., Hazay, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A.: Signature schemes secure against hard-to-invert leakage. In: ASIACRYPT. LNCS, vol. 7658, pp. 98–115. Springer (2012)
- Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In: FOCS. pp. 308–317. IEEE Computer Society (1990)
- Feng, H., Tang, Q.: Witness authenticating nizks and applications. In: CRYPTO (4). Lecture Notes in Computer Science, vol. 12828, pp. 3–33. Springer (2021)
- Fischlin, M.: Security of NMAC and HMAC based on non-malleability. In: CT-RSA. LNCS, vol. 4964, pp. 138– 154. Springer (2008)
- 30. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: STOC. pp. 25–32. ACM (1989)
- Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: EUROCRYPT. LNCS, vol. 4965, pp. 415–432. Springer (2008)
- Hsiao, C., Lu, C., Reyzin, L.: Conditional computational entropy, or toward separating pseudoentropy from compressibility. In: EUROCRYPT. LNCS, vol. 4515, pp. 169–186. Springer (2007)
- Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. IACR Cryptol. ePrint Arch. 2020, 1003 (2020)
- Kreuter, B., Lepoint, T., Orrù, M., Raykova, M.: Anonymous tokens with private metadata bit. In: CRYPTO (1). LNCS, vol. 12170, pp. 308–336. Springer (2020)
- 35. Libert, B., Vergnaud, D.: Group signatures with verifier-local revocation and backward unlinkability in the standard model. In: CANS. LNCS, vol. 5888, pp. 498–517. Springer (2009)
- Ma, S., Huang, Q.: Plaintext-checkable encryption with unlink-cca security in the standard model. In: ISPEC. LNCS, vol. 11879, pp. 3–19. Springer (2019)
- Ma, S., Mu, Y., Susilo, W.: A generic scheme of plaintext-checkable database encryption. Inf. Sci. 429, 88–101 (2018)
- Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC. pp. 427–437. ACM (1990)
- Oleksenko, O., Trach, B., Krahn, R., Silberstein, M., Fetzer, C.: Varys: Protecting SGX enclaves from practical side-channel attacks. In: USENIX Annual Technical Conference. pp. 227–240. USENIX Association (2018)

- 40. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: CRYPTO (1). LNCS, vol. 11692, pp. 89–114. Springer (2019)
- 41. Santis, A.D., Crescenzo, G.D., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: CRYPTO. LNCS, vol. 2139, pp. 566–598. Springer (2001)
- 42. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: FOCS. pp. 600–611. IEEE Computer Society (2017)

## A Analysis of Warm-up Constructions

We present a simple WA-NIZK construction  $\Pi_{simple}$  for a language L that admits a pseudorandom witness distribution (or equivalently, a  $|w^{I}|$ -unpredictable sampler where  $|w^{I}|$  denotes the length of  $w^{I}$ ), and a more general WA-NIZK construction  $\Pi_{CI}$  for a language L that admits a CRS-independent k-unpredictable sampler. Then, we analyze the security of  $\Pi_{CI}$ , while all results about  $\Pi_{simple}$  can be obtained by considering it as a special case of  $\Pi_{CI}$  where the randomness extractor is simply an identical function.

#### A.1 Ingredients

We introduce ingredients used in our constructions.

**Computational Strong Randomness Extractor** Computational strong randomness extractors are used to extract pseudorandomness from sources with computational entropy.

**Definition 25 (Strong Randomness Extractor).** Let  $\mathcal{E} = \{E_{\lambda}\}_{\lambda \in \mathbb{N}}$  be an ensemble with the form of  $E_{\lambda} : \{0,1\}^{\eta(\lambda)} \times \{0,1\}^{\ell(\lambda)} \to \{0,1\}^{\zeta(\lambda)}$  where  $\eta(),\ell(),\zeta()$  are polynomials. We say  $\mathcal{E}$  is a  $(k(\lambda),\epsilon(\lambda))$  strong extractor (where  $\epsilon \leq \operatorname{negl}(\lambda)$ ) for sources with unpredictability entropy (resp. Hill entropy), if for (X,Z) satisfying  $\mathbf{H}^{\operatorname{unp}}(X|Z) \geq k(\lambda)$  (resp.  $\mathbf{H}^{\operatorname{hill}}(X|Z) \geq k(\lambda)$ ), and for any non-uniform PPT adversary  $\mathcal{A}$ , it holds that

$$|\Pr[(x,z) \leftarrow (X,Z), r \leftarrow \$ \{0,1\}^{\ell(\lambda)}, rk \leftarrow E_{\lambda}(x,r) : 1 \leftarrow \mathcal{A}(z,r,rk)] - \Pr[(x,z) \leftarrow (X,Z), r \leftarrow \$ \{0,1\}^{\ell(\lambda)}, rk \leftarrow \$ \{0,1\}^{\zeta(\lambda)} : 1 \leftarrow \mathcal{A}(z,r,rk)]| \le \epsilon(\lambda).$$

<u>Parameters in our Construction</u>. Samplers with different unpredictability have different requirements on the randomness extractors.

- For a sampler G with k-Hill entropy, the computational strong randomness extractor can be instantiated with any strong randomness extractor, such as universal hash families [4] and weak pseudorandom function families [23], possibly along with a pseudorandomness generator (PRG). For the conditional entropy k of  $W^I$ , we only require it to be large enough for extracting a pseudorandom string of  $\lambda$  bits, since such a string can always be extended to a pseudorandom string of  $\zeta(\lambda)$  bits by a PRG. More precisely, subjected to the optimal bound on entropy loss of randomness extractors, k should be larger than  $\lambda + 2 \log(1 \setminus \epsilon(\lambda))$ , where  $\epsilon(\lambda)$  is negligible in  $\lambda$ .
- For a sampler G with k-unpredictability entropy, we need a particular type of extractor called a reconstructive extractor [32]. The well-known Goldreich-Levin function GL [30], namely  $GL(\mathbf{M}, \mathbf{x}) = \mathbf{M}\mathbf{x}$ , where  $\mathbf{M}\mathbf{x}$  is the matrix-vector product of randomly-sampled matrix  $\mathbf{M}$  and  $\mathbf{x}$  over GF(2), is such a reconstructive extractor (where  $\mathbf{M}$  is the seed and  $\mathbf{x}$  is the source). For using this extractor to yield a statistically-close-to-uniform string over  $\{0,1\}^{\lambda}$ , the entropy k needs to be larger than  $\lambda + \mathcal{O}(1 \setminus \epsilon)$  where  $\epsilon \leq \mathsf{negl}(\lambda)$ .

In order to ensure the identifier uniqueness, the randomness extractor should be collision-resistant. A collision-resistant randomness extractor has an extra key  $k_{\mathsf{Ext}}$ , such that for  $k_{\mathsf{Ext}} \leftarrow K_{\mathsf{Ext}}$ , it infeasible to find  $(w_1, w_2, r_1, r_2)$  such that  $E_{\lambda}^{k_{\mathsf{Ext}}}(w_1, r_1) = E_{\lambda}^{k_{\mathsf{Ext}}}(w_2, r_2)$ . Collision-resistant extractors [24] can be constructed from pairwise independent permutations, and work for source with k-Hill entropy such that  $k \geq \lambda + 2\log(1 \setminus \epsilon(\lambda))$ .

(Collision Resistant) Pseudorandom Function Family Let  $\mathcal{F} = \{F_{\lambda}\}_{\lambda \in \mathbb{N}}$  be an function ensemble with the form of  $F_{\lambda} : \{0,1\}^{\zeta(\lambda)} \times \{0,1\}^{\kappa} \to \{0,1\}^{\gamma(\lambda)}$ . We say  $\mathcal{F}$  is a pseudorandom function familiy (PRF), if for  $rk \leftarrow \{0,1\}^{\zeta(\lambda)}$ , the function  $F_{\lambda}(rk, \cdot)$  is computationally indistinguishable with a random function f from  $\{0,1\}^{\kappa(\lambda)}$  to  $\{0,1\}^{\gamma(\lambda)}$ .

In order to ensure the identifier uniqueness, the PRF should be collision-resistant that it infeasible to find  $(rk_1, rk_2, t_1, t_2)$  such that  $F_{\lambda}^{k_{prf}}(rk_1, t_1) = F_{\lambda}^{k_{prf}}(rk_2, t_2)$ . Collision-resistant PRF [19] can be constructed from any one-way permutation.

In our construction, for simplicity we take the output space of the randomness extractor  $E_{\lambda}$  as the key space of  $F_{\lambda}$ , and we require  $\gamma(\lambda) \geq 2\zeta(\lambda)$  s.t. for a random  $(t, \alpha)$  the probability that there is a key rk such that  $F_{\lambda}(rk, t) = \alpha$  is negligible.

**NIZK** For different security goals of our construction, the NIZK proof system  $\Pi_{zk}$  should satisfy various properties, including adaptively unbounded zero-knowledge, simulation soundness, simulation extractability, and we will detail it in the security analysis.

#### A.2 The constructions

Construction for pseudorandom distribution:  $\Pi_{\text{simple}}$  Let  $F = \{F_{\lambda}\}_{\lambda \in \mathbb{N}}$  be a PRF (w.l.o.g., with the key length same as  $|w^{I}|$ , and output length  $2|w^{I}|^{15}$ ), and let  $\Pi_{zk} = \{S_{zk}, P_{zk}, V_{zk}\}$  be a NIZK for an NP language

$$L_{\mathsf{simple}} := \{ (x, t, \alpha); (w^{I}, w^{NI}) : (w^{I}, w^{NI}) \in R_{L}(x) \land w^{I} \in R_{L}^{I}(x) \land \alpha = F_{\lambda}(w^{I}, t) \}.$$
(10)

We construct a WA-NIZK  $\Pi_{simple} = \{Setup, Prove, Verify, Identify\}$ , and illustrate the detail in Fig.8 (excluding the boxed items). *Correctness* is straightforward, while *security* follows the intuition we sketched above, and we establish the result below.

**Theorem 5.** Let  $\Pi_{simple}$  be as specified in Fig.8, the following results hold:

- $\Pi_{simple}$  satisfies the authenticating (knowledge) soundness, if  $\Pi_{zk}$  satisfies the (knowledge) soundness;
- $\Pi_{\text{simple}}$  satisfies the unforgeability w.r.t. all  $|w^{I}|$ -unpredictable samplers, if F is a secure PRF, and  $\Pi_{\text{zk}}$  satisfies the simulation soundness;
- $\Pi_{simple}$  satisfies the entropic ZK w.r.t. all  $|w^{I}|$ -unpredictable samplers, if F is a secure PRF and  $\Pi_{zk}$  satisfies the zero knowledgeness.

Observe that, by further requiring the PRF to be collision-resistant,  $\Pi_{simple}$  can also achieve the stronger *identifier uniqueness*. A collision-resistance PRF is known to exist if one-way permutations exist [19]. Since collision resistance against non-uniform adversaries necessarily requires an extra honestly generated public parameter/key, we could slightly modify the construction  $\Pi_{simple}$  such that the CRS contains a  $k_{PRF}$ , and  $F(w^{I}, r)$  then becomes  $F_{k_{PRF}}(w^{I}, r)$ .

<sup>&</sup>lt;sup>15</sup> With such an output length, the probability that for a random pair  $(t, \alpha)$  there exists a key k such that  $F_{\lambda}(k, t) = \alpha$  is negligible, which faciliates the proof for unforgeability.



Fig. 8. CRS-independent Constructions. For the construction  $\Pi_{simple}$  (w.r.t pseudorandom distributions), all boxed items shall be removed.

Construction for CRS-independent distributions:  $\Pi_{\mathsf{Cl}}$  Let  $E = \{E_{\lambda}\}$  be a (computational) strong randomness extractor:  $E_{\lambda} : \{0,1\}^{\eta(\lambda)} \times \{0,1\}^{\ell(\lambda)} \to \{0,1\}^{\zeta(\lambda)}$  (where  $\eta(),\ell(),\zeta()$  are polynomials), let  $F = \{F_{\lambda}\}_{\lambda \in \mathbb{N}}$  be a PRF with the key space  $\{0,1\}^{\zeta}$ , and let  $\Pi_{\mathsf{zk}} = \{\mathsf{S}_{\mathsf{zk}},\mathsf{P}_{\mathsf{zk}},\mathsf{V}_{\mathsf{zk}}\}$  be a NIZK for an NP language

$$L_{\mathsf{CI}} := \{ (x, t, \alpha, r); (w^I, w^{NI}) : \\ (w^I, w^{NI}) \in R_L(x) \land w^I \in R_L^I(x) \land \alpha = F_\lambda(E_\lambda(w^I, r), t) \}.$$

$$(11)$$

We illustrate the construction  $\Pi_{CI}$  in Fig.8 (with boxed items included).

The completeness of this construction is easy to verify, which directly comes from the completeness of the underlying NIZK proof system. We now argue security.

The entropic ZK and the unforgeability are defined w.r.t. an unpredictable sampler. As discussed before, this construction only works for CRS-Independent samplers. Regarding the samplers that our construction can work for, it depends on the computational strong randomness extractor used in our construction. Particularly, for a joint random variable  $(X, (W^I, W^{NI}))$  outputted by G, the randomness extractor should "transform"  $W^I$  to a random variable K over the key space of the PRF *s.t.* K is pseudorandom even conditioned on X. The detailed parameters have been sketched above.

We claim the security of our construction  $\Pi_{CI}$  in the following theorem.

**Theorem 6.** Let  $(X, (W^I, W^{NI}))$  be the output of an arbitrary k-unpredictable sampler  $G(1^{\lambda})$  for an NP language L and an identifier relation  $R_L^I$ . Let  $\mathcal{E} = \{E_{\lambda}\}_{\lambda \in \mathbb{N}}$  be a randomness extractor which extracts a pseudorandom variable over  $\{0, 1\}^{\zeta(\lambda)}$  from  $W^I$  conditioned on X.

- $\Pi_{Cl}$  satisfies the authenticating (knowledge) soundness, if  $\Pi_{zk}$  satisfies the (knowledge) soundness;
- $\Pi_{Cl}$  satisfies the unforgeability w.r.t. all k-unpredictable samplers, if F is a secure PRF, and  $\Pi_{zk}$  satisfies the simulation soundness.
- $\Pi_{Cl}$  satisfies the entropic ZK w.r.t. all k-unpredictable samplers, if F is a secure PRF and  $\Pi_{zk}$  satisfies the zero knowledgeness.

We remark this theorem subsumes Theorem.5 by setting  $\mathsf{E}_{\lambda}$  as an identical function.

Achieving the identifier uniqueness. By requiring both the randomness extractor and the PRF to be collision-resistant,  $\Pi_{CI}$  can also achieve identifier uniqueness. Since the definition of collision resistance (of both PRFs and randomness extractors) against non-uniform adversaries necessarily requires an extra key is generated when the security game starts, we need to modify our construction slightly. Specifically, the two extra keys  $k_{PRF}$  and  $k_{Ext}$  should be generated in the Setup algorithm and be included in the CRS, and whenever the randomness extractor  $E_{\lambda}$  and the PRF  $F_{\lambda}$  are used in other algorithms, they are specified by the keys  $k_{Ext}$  and  $k_{PRF}$  respectively. We conclude the result in the following lemma.

**Lemma 7.** Let  $\mathcal{E}$  be a collision-resistant randomness extractor that satisfies properties required in Theorem.6. Let  $\mathcal{F}$  be a collision-resistant PRF. Then,  $\Pi_{CI}$  (with slight modifications introduced above) has the identifier uniqueness.

The proof for this lemma is given in Appendix.A.4. We note the collision-resistant randomness extractors are only known to exist for sources with enough Hill-entropy. Thus, current instantiations of  $\Pi_{CI}$  with identifier uniqueness can only be waNIZKs for samplers with enough Hill-entropy.

#### A.3 Proof of Theorem.6

We first prove the entropic ZK of our construction  $\Pi_{CI}$  and then prove the soundness definitions.

**Proof of entropic ZK** We start with presenting the simulator (SimSetup, SimProve) of  $\Pi_{CI}$ . Let  $(SS_{zk}, SP_{zk})$  be the simulator of the underlying NIZK proof system  $\Pi_{zk}$ , and then we construct (SimSetup, SimProve) as follows.

1. SimSetup $(1^{\lambda})$ . Firstly invoke the simulator of  $\Pi_{zk}$ :  $(\sigma_{zk}, \tau_{zk}) \leftarrow SS_{zk}(1^{\lambda})$ ; Then, uniformly choose a seed  $r \leftarrow \{0, 1\}^{\ell(\lambda)}$ ; Return the CRS  $\sigma = (\sigma_{zk}, r)$  and the trapdoor  $\tau = \tau_{zk}$ . 2. SimProve $(\sigma, \tau, x)$ . Firstly sample  $t \leftarrow \{0, 1\}^{\kappa(\lambda)}$  and  $\alpha \leftarrow \{0, 1\}^{\gamma(\lambda)}$ ; Then invoke the simulator of  $\Pi_{zk}$  to prove  $(x, \alpha, t, r) \in L_{CI}$ :  $\pi_{zk} \leftarrow SP_{zk}(\sigma_{zk}, \tau_{zk}, (x, \alpha, t, r))$ ; Return the simulated proof  $\pi = (\alpha, t, \pi_{zk})$ .

We denote the event  $[pp \leftarrow \mathsf{PG}(1^{\lambda}); G \leftarrow \mathcal{A}(pp); \sigma \leftarrow \mathsf{Setup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma)]$  by  $\mathsf{Event}_{\mathsf{zk}, \mathcal{A}}^{\mathsf{real}}$ , and denote the event  $[pp \leftarrow \mathsf{PG}(1^{\lambda}); G \leftarrow \mathcal{A}(pp); (\sigma, \tau) \leftarrow \mathsf{Sim}\mathsf{Setup}(1^{\lambda}) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{S}1}, \mathcal{O}_{\mathsf{S}2}}(\sigma)]$  by  $\mathsf{Event}_{\mathsf{zk}, \mathcal{A}}^{\mathsf{sim}}$ . Our goal is to show the difference between  $\Pr[\mathsf{Event}_{\mathsf{zk}, \mathcal{A}}^{\mathsf{real}}]$  and  $\Pr[\mathsf{Event}_{\mathsf{zk}, \mathcal{A}}^{\mathsf{sim}}]$  is negligible in  $\lambda$ , for any non-uniform PPT adversary  $\mathcal{A}$ . To do this, we define a sequence of events  $(\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^i)_{i\in[0,4]}$ , where the  $\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^0$  is  $\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{\mathsf{real}}$  and the  $\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^4$  is  $\text{Event}_{\mathsf{z}\mathsf{k},\mathcal{A}}^{\mathsf{sim}}$ , and show the differences between probabilities of adjacent events are negligible (see Lemma.8, 9, 10).

The hybrid event  $\text{Event}_{\mathsf{zk},\mathcal{A}}^1$ . This event is defined as follows:  $[pp \leftarrow \mathsf{PG}(1^{\lambda}); G \leftarrow \mathcal{A}(pp); (\sigma, \tau) \leftarrow \overline{\mathsf{SimSetup}(1^{\lambda}): 1 \leftarrow \mathcal{A}^{\mathsf{OP1}_1,\mathsf{OP1}_2}(\sigma)]}$ , where the oracle  $\mathsf{OP1}_1, \mathsf{OP1}_2$  are defined by modifying  $\mathcal{O}_{\mathsf{P1}}, \mathcal{O}_{\mathsf{P2}}$ . Precisely,

$OP1_1(\sigma, \tau, pp)$	$OP1_2(\sigma, \tau, pp, x_i, EG, st)$
$\overline{i++;}$	$\overline{\mathrm{Find}(i, x_i, (w_i^I, w_i^{NI}))} \in st$
$(x_i, (w_i^I, w_i^{NI})) \leftarrow G(pp);$	$(\bar{x}, \bar{w}^{NI}) \leftarrow EG_{\mathtt{real}}(pp, \sigma, x_i, (w_i^I, w_i^{NI}))$
$st \leftarrow st \cup (i, x_i, (w_i^I, w_i^{NI}));$	$\bar{\pi} \leftarrow \boxed{Prove^{(1)}(\sigma, \tau, \bar{x}, w_i^I, \bar{w}^{NI})}$
$\pi_i \leftarrow \boxed{Prove^{(1)}(\sigma, \tau, x_i, w_i^I, w_i^{NI})}$	$\mathbf{return} \ (\bar{x}, \bar{\pi})$
$\mathbf{return}\ (x_i,\pi_i)$	

The boxed items are different from the prover oracles  $\mathcal{O}_{P1}$ ,  $\mathcal{O}_{P2}$ . Here, the hybrid prover algorithm  $\mathsf{Prove}^{(1)}$  is executed as follows, where the boxed items are different from the prover algorithm  $\mathsf{Prove}$ .

 $\mathsf{Prove}^{(1)}(\sigma, \tau, x, (w^I, w^{NI})).$ 1. Parse  $\sigma = (\sigma_{\mathsf{zk}}, r)$ , and  $\tau = \tau_{\mathsf{zk}}$ ; 2. Extract a PRF key from  $w^I$  using the CRS  $r: rk \leftarrow E_{\lambda}(w^I, r);$ 3. Evaluate the PRF on a random input:  $t \leftarrow \{0, 1\}^{\kappa(\lambda)}$  and  $\alpha \leftarrow F_{\lambda}(rk, t)$ ; and Prove  $\in$ L $(\alpha, t)$ is honestly generated 4. xby using the simulation prover algorithm of  $\Pi_{\mathsf{zk}}$ :  $\pi_{\mathsf{zk}} \leftarrow \mathsf{SP}_{\mathsf{zk}}(\sigma_{\mathsf{zk}}, \tau_{\mathsf{zk}}, (x, \alpha, t, r))$ . 5. Output the final proof  $\pi = (\alpha, t, \pi_{zk})$ .

The hybrid event  $\text{Event}_{zk,\mathcal{A}}^2$ . Then, we build the next hybrid event  $\text{Event}_{zk,\mathcal{A}}^2$ . This event is almost identical to  $\text{Event}_{zk,\mathcal{A}}^1$ , with the following exceptions: it replaces the oracles  $OP1_1, OP1_2$  with oracles  $OP2_1, OP2_2$ .

$OP2_1(\sigma, \tau, pp)$	$OP2_2(\sigma, \tau, pp, x_i, EG, st)$
$\overline{i++;(x_i,(w_i^I,w_i^{NI}))} \leftarrow G(pp);$	$\operatorname{Find}(i, \boxed{rk_i}, x_i, (w_i^I, w_i^{NI})) \in st$
$rk_i \leftarrow \{0,1\}^{\zeta(\lambda)}$	$(\bar{x}, \bar{w}^{NI}) \leftarrow EG_{\mathtt{real}}(pp, \sigma, x_i, (w_i^I, w_i^{NI}))$
$st \leftarrow st \cup (i, \boxed{k_i}, x_i, (w_i^I, w_i^{NI}));$	$\bar{\pi} \leftarrow \boxed{Prove^{(2)}(\sigma, \tau, \bar{x}, rk_i)}$
$\pi_i \leftarrow \boxed{Prove^{(2)}(\sigma, \tau, x_i, rk_i)}$	$\mathbf{return}\ (\bar{x},\bar{\pi})$
<b>return</b> $(x_i, \pi_i)$	

Here the hybrid prover algorithm  $Prove^{(2)}$  only differs from  $Prove^{(1)}$  at step 2: it just set the key of PRF as the input rk.

The hybrid event  $\text{Event}_{zk,\mathcal{A}}^3$ . Next, we present another hybrid event  $\text{Event}_{zk,\mathcal{A}}^3$ . This is event is almost identical to  $\text{Event}_{zk,\mathcal{A}}^2$ , with the following exceptions: it replaces the oracles  $OP2_1, OP2_2$  with  $OP3_1, OP3_2$ . The only difference between the them is that  $OP3_1, OP3_2$  execute the algorithm  $Prove^{(3)}$  rather than  $Prove^{(2)}$ .

Prove<sup>(3)</sup> $(\sigma, \tau, x, rk)$ . It only differs from Prove<sup>(2)</sup> at the step 3. 3. Randomly sample  $t \leftarrow \{0, 1\}^{\kappa(\lambda)}$  and  $\alpha \leftarrow \{0, 1\}^{\gamma(\lambda)}$ .

We prove the differences between  $\text{Event}_{zk,\mathcal{A}}^{0}$ ,  $\text{Event}_{zk,\mathcal{A}}^{1}$ ,  $\text{Event}_{zk,\mathcal{A}}^{2}$ ,  $\text{Event}_{zk,\mathcal{A}}^{3}$  are negligible in the following three lemmas. And We note in  $\text{Event}_{zk,\mathcal{A}}^{3}$ , the output of the hybrid algorithm  $\text{Prove}^{(3)}$  is identical to the simulation prover algorithm SimProve. Although in context  $\text{Prove}^{(3)}$  differs from SimProve at the step 2, the key rk generated in step 2 is not used in the latter steps. Therefore,

$$|\Pr[\texttt{Event}^{3}_{\mathsf{zk},\mathcal{A}}] - \Pr[\texttt{Event}^{\mathsf{sim}}_{\mathsf{zk},\mathcal{A}}]| \le \mathsf{negl}(\lambda).$$
(12)

Combined the following lemmas together, we have proved the entropic ZK of  $\Pi_{CI}$ .

**Lemma 8.** if  $\Pi_{zk}$  is adaptively unbounded zero-knowledge, then for any non-uniform PPT adversary  $\mathcal{A}$ ,

$$|\Pr[\text{Event}_{\mathsf{zk},\mathcal{A}}^{\mathsf{real}}] - \Pr[\text{Event}_{\mathsf{zk},\mathcal{A}}^{1}]| \le \mathsf{negl}(\lambda).$$

Proof. We can consider an adversary  $C_1$  for  $\Pi_{zk}$ .  $C_1$  is given a CRS  $\sigma_{zk}$  and a prover oracle  $\mathcal{O}_{zk}$ , which are either an honest CRS and a real prover oracle or a simulated CRS and a simulation prover oracle. Then,  $C_1$  mimics all steps in  $\text{Event}_{zk,\mathcal{A}}^1$  for  $\mathcal{A}$ :  $C_1$  provides  $\sigma_{zk}$  to  $\mathcal{A}$ , and answers queries to  $\text{OP1}_1, \text{OP1}_2$  by querying  $\mathcal{O}_{zk}$  whenever it needs to execute the algorithm  $\text{SP}_{zk}$ . It is easy to see that, when  $\sigma_{zk}$  is a simulated CRS and  $\mathcal{O}_{zk}$  is a simulation oracle, at the point of  $\mathcal{A}$ 's view, all steps are identical to that in  $\text{Event}_{zk,\mathcal{A}}^1$ ; when  $\sigma_{zk}$  and  $\mathcal{O}_{zk}$  are simulated, these steps are identical to that in  $\text{Event}_{zk,\mathcal{A}}^{\text{real}}$ . Therefore, if the  $|\Pr[\text{Event}_{zk,\mathcal{A}}^{\text{real}}] - \Pr[\text{Event}_{zk,\mathcal{A}}^1]| > \text{negl}(\lambda)$ ,  $C_1$  can break the zero-knowledge property of  $\Pi_{zk}$ .

**Lemma 9.** if the computational randomness extractor  $\mathcal{E}$  is secure, then

$$|\Pr[\text{Event}_{\mathsf{zk},\mathcal{A}}^1] - \Pr[\text{Event}_{\mathsf{zk},\mathcal{A}}^2]| \le \mathsf{negl}(\lambda).$$

*Proof.* To facilitate our proof, we design a special form of security game for computational randomness extractors as follows.

1. Generate  $pp \leftarrow \mathsf{PG}(1^{\lambda})$ .  $\mathcal{C}_2$  on input pp specifies a sampler  $G(pp) = (X, W^I, W^{NI})$  where  $W^I$ (conditioned on X, PP) satisfies the entropy requirement of  $\mathcal{E}$ . 2. Sample M instances:  $(x_i, (w_i^I, w_i^{NI}))$  for  $i \in [M]$ . 3. Sample a seed  $r \leftarrow \{0, 1\}^{\ell(\lambda)}$ , and a bit  $b \leftarrow \{0, 1\}$ . 4. For  $i \in [M]$ , if b = 0, sample  $rk_i \leftarrow \{0, 1\}^{\zeta(\lambda)}$ ; if b = 1, generate  $rk_i \leftarrow E_{\lambda}(w_i^I, r)$ . 5. Return  $\mathcal{C}_2$  a table consisting of tuples  $\langle x_i, rk_i \rangle$ . 6.  $\mathcal{C}_2$  outputs b'. Since the tuples  $(x_i, (w_i^I, w_i^{NI}))_{i \in [M]}$  are independent samples, by standard arguments it is easy to see for every non-uniform PPT adversary  $C_2$ ,  $\Pr[b' = b] \leq \operatorname{negl}(\lambda)$ , when  $\mathcal{E}$  is a computational randomness extractor working for  $(G, \mathsf{PG})$  sources. However, assume there is an adversary  $\mathcal{A}$  s.t.  $|\Pr[\operatorname{Event}_{\mathsf{zk},\mathcal{A}}^1] - \Pr[\operatorname{Event}_{\mathsf{zk},\mathcal{A}}^2]| > \operatorname{negl}(\lambda)$ . Then, we can construct an adversary  $C_2$  which breaks the computational randomness extractor by leveraging  $\mathcal{A}$ .

We now state the strategy of  $C_2$ .  $C_2$  mimics all steps in Event<sup>2</sup><sub>zk,A</sub> with A.

1. After  $\mathcal{A}$  specifies the unpredictable sampler G,  $\mathcal{C}_2$  sets G in Step 1.

2. Then,  $C_2$  is given a seed r and a table of tuples  $\langle (x_i, rk_i \rangle)$ , and generates a simulated CRS  $\sigma_{\mathsf{zk}}$  and its trapdoor  $\tau_{\mathsf{zk}}$ .

3. When  $\mathcal{A}$  queries  $\mathsf{OP1}_1$  with the sampler G,  $\mathcal{C}_2$  picks a tuple  $\langle (x_i, rk_i \rangle$  which was not used previously, runs the PRF  $\alpha \leftarrow F_{\lambda}(rk_i, t)$  on a uniformly chosen input t, generates the simulated proof  $\pi_{\mathsf{zk}}$  for  $(x_i, \alpha, t, r)$ , and returns  $(x_i, (\alpha, t, \pi_{\mathsf{zk}}))$  to  $\mathcal{A}$ .

4. When  $\mathcal{A}$  queries  $\mathsf{OP1}_2$  with a dual-mode extended sampler  $\mathsf{EG} = \{\mathsf{EG}_{\mathsf{real}}, \mathsf{EG}_{\mathsf{sim}}\}$  and a statement  $x_i$ ,  $\mathcal{C}_2$  firstly checks whether  $x_i$  was previously return to  $\mathcal{A}$ , and aborts if not; otherwise,  $\mathcal{C}_2$  runs  $\mathsf{EG}_{\mathsf{sim}}$  (see Def.14), to generate the statement  $\bar{x}$ . Then,  $\mathcal{C}_2$  runs the PRF  $\alpha \leftarrow F_{\lambda}(rk_i, t)$  on a uniformly chosen input t where  $rk_i$  corresponds to  $(x_i)$ , generates the simulated proof  $\pi_{\mathsf{zk}}$  for  $(\bar{x}, \alpha, t, r)$ , and returns  $(\bar{x}, (\alpha, t, \pi_{\mathsf{zk}}))$  to  $\mathcal{A}$ .

We assume that  $\mathcal{A}$  makes at most M queries to  $\mathsf{OP1}_1$ . It is easy to see that when b = 0, *i.e.*, each  $rk_i$  is uniformly chosen, the above steps simulated by  $\mathcal{C}_2$  are computationally indistinguishable with that in  $\mathsf{Event}^2_{\mathsf{zk},\mathcal{A}}$ ; when b = 1, these steps are computationally indistinguishable with that in  $\mathsf{Event}^2_{\mathsf{zk},\mathcal{A}}$ . Therefore, if there is an  $\mathcal{A}$  s.t.  $|\Pr[\mathsf{Event}^2_{\mathsf{zk},\mathcal{A}}] - \Pr[\mathsf{Event}^2_{\mathsf{zk},\mathcal{A}}]| > \mathsf{negl}(\lambda)$ ,  $\mathcal{C}_2$  can guess the value of b with non-negligible probability.

**Lemma 10.** If  $\mathcal{F}$  is a PRF, then

$$|\Pr[\textit{Event}^2_{\mathsf{zk},\mathcal{A}}] - \Pr[\textit{Event}^3_{\mathsf{zk},\mathcal{A}}]| \le \mathsf{negl}(\lambda).$$

*Proof.* Note in  $\text{Event}_{\mathsf{zk},\mathcal{A}}^2$ ,  $\alpha \leftarrow F_{\lambda}(rk,t)$  where rk is uniformly chosen at random. From the pseudorandomness of  $\mathcal{F}$ ,  $(t,\alpha)$  is computationally indistinguishable with  $(t \leftarrow \{0,1\}^{\kappa(\lambda)}, \alpha \leftarrow \{0,1\}^{\gamma(\lambda)})$ . Therefore the difference between the probabilities of the two events is negligible.

**Proof of authenticating soundness.** Now we prove the authenticating soundness of  $\Pi_{CI}$ . Assume there is a non-uniform PPT adversary  $\mathcal{A}$  which breaks the authenticating soundness, *i.e.*,  $\mathcal{A}$ outputs a valid proof  $\pi$  for a statement x such that either x is a false statement or no identifier witness of x can authenticate  $\pi$ . We first parse  $\pi = (\alpha, t, \pi_{zk})$ . From the definition of Cl.Verify,  $\pi_{zk}$  is a valid proof for  $(x, \alpha, t, r)$  w.r.t. the proof system  $\Pi_{zk}$ . Then, from the soundness of  $\Pi_{zk}$ , the statement  $(x, \alpha, t, r) \in L_{CI}$  must be true, *i.e.*, there must be a witness  $(w^I, w^{NI}) \in R_L(x)$ and  $F_{\lambda}(E_{\lambda}(w^I, r), t) = \alpha$ . According the description of Cl.Identify, this identifier witness  $w^I$  authenticates  $\pi$ . The above arguments contradicts our assumption that  $\mathcal{A}$  breaks the authenticating soundness. Therefore, we complete our proof.

**Proof of authenticating knowledge soundness.** We firstly establish the knowledge extractor for our construction. Let  $(\mathsf{Ext}_{\mathsf{zk},0},\mathsf{Ext}_{\mathsf{zk},1})$  be knowledge extractor of  $\Pi_{\mathsf{zk}}$ . The knowledge extractor  $(\mathsf{Ext}_0,\mathsf{Ext}_1)$  of  $\Pi_{CI}$  can be constructed as follows.

1.  $\mathsf{Ext}_0(1^{\lambda})$ . Firstly invoke the knowledge extractor of  $\Pi_{\mathsf{zk}}$ :  $(\sigma_{\mathsf{zk}}, \xi_{\mathsf{zk}}) \leftarrow \mathsf{Ext}_{\mathsf{zk},0}(1^{\lambda})$ ; Then, uniformly choose a seed  $r \leftarrow \{0, 1\}^{\ell(\lambda)}$ ; Return the CRS  $\sigma = (\sigma_{\mathsf{zk}}, r)$  and the trapdoor  $\xi = \xi_{\mathsf{zk}}$ . 2.  $\mathsf{Ext}_1(\sigma, \xi, \pi)$ . Firstly parse the proof  $\pi = (\alpha, t, \pi_{\mathsf{zk}})$ ; Then, use the knowledge extractor of  $\Pi_{\mathsf{zk}}$ :  $(w^I, w^{NI}) \leftarrow \mathsf{Ext}_{\mathsf{zk},1}(\sigma_{\mathsf{zk}}, \xi_{\mathsf{zk}}, \pi_{zk}, (x, \alpha, t, r))$ .

We show the extracted witness  $(w^I, w^{NI}) \in R_L(x)$  and  $w^I$  authenticates  $\pi$ . By the definition of knowledge soundness, since  $\pi_{zk}$  is a valid proof for  $(x, \alpha, t, r)$ , we have  $(w^I, w^{NI}) \in R_L(x)$  and  $\alpha = F_{\lambda}(E_{\lambda}(w^I, r), t)$ . By the definition of the algorithm Identify,  $w^I$  authenticates  $\pi$ . Thus, we complete the proof for the authenticating knowledge soundness.

**Proof of unforgeability** Recall the definition of unforgeability in Def.18, and denote the event that the adversary  $\mathcal{A}$  succeeds by

$$\mathtt{Event}_{\mathtt{ex},\mathcal{A}} := \begin{bmatrix} pp \leftarrow \mathsf{PG}(1^{\lambda}); \sigma \leftarrow \mathsf{Setup}(1^{\lambda}); G \leftarrow \mathcal{A}(pp); \\ (x^*, \pi^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}1}, \mathcal{O}_{\mathsf{P}2}}(\sigma, pp) : (x^*, \pi^*) \notin \mathsf{Hist} \\ \land \mathsf{Verify}(\sigma, x^*, \pi^*) = 1 \land \exists w^I \in st, \mathsf{Identify}(\sigma, x^*, \pi^*, w^I) = 1 \end{bmatrix}$$

We prove  $\Pr[\text{Event}_{ex,\mathcal{A}}] \leq \operatorname{negl}(\lambda)$  by considering the following hybrid event.

The hybrid event  $\text{Event}_{e_{x,\mathcal{A}}}^{0}$ . This event is almost identical to  $\text{Event}_{e_{x,\mathcal{A}}}$ , except that Setup is replaced by SimSetup, and the oracles  $\mathcal{O}_{P1}, \mathcal{O}_{P2}$  are replaced with the oracles  $\text{OP1}_1, \text{OP1}_2$  that we introduced in  $\text{Event}_{z_{k,\mathcal{A}}}^2$ . It is easy to see if  $\Pi_{z_k}$  is zero-knowledge, then  $|\Pr[\text{Event}_{e_{x,\mathcal{A}}}] - \Pr[\text{Event}_{e_{x,\mathcal{A}}}^0]| \leq \operatorname{negl}(\lambda)$ .

The hybrid event  $\text{Event}_{e_{x,\mathcal{A}}}^1$ . This event is almost identical to  $\text{Event}_{e_{x,\mathcal{A}}}^0$ , except that the oracles are replaced with the oracles  $OP2_1, OP2_2$  that we introduced in  $\text{Event}_{z_{k,\mathcal{A}}}^2$ . And the last success condition is changed to " $\exists rk \in st, F_{\lambda}(rk, t) = \alpha$ , where  $\pi = (\alpha, t, \pi_{z_k})$ ".

**Lemma 11.** If  $\mathcal{E}$  is computational randomness extractor, then  $|\Pr[\text{Event}_{ex,\mathcal{A}}] - \Pr[\text{Event}_{ex,\mathcal{A}}^1]| \le \operatorname{negl}(\lambda)$ , for any non-uniform PPT adversary  $\mathcal{A}$ .

Proof (sketch). This proof is very similar to the proof for Lemma.9. Informally, we consider an adversary  $\mathcal{C}$ , which is given a table of tuples  $\langle x_i, rk_i \rangle$ . Here,  $rk_i$  is either uniformly sampled at random, or is extracted from  $w_i^I$  by using a public seed r.  $\mathcal{C}$  can mimic all steps in  $\texttt{Event}_{\mathsf{ex},\mathcal{A}}^1$ , using the information contained in the table. The strategy of  $\mathcal{C}$  is the same as that of  $\mathcal{C}_2$  given in the Lemma.9. When  $k_i$  is uniformly sampled, these steps are computationally indistinguishable with that in  $\texttt{Event}_{\mathsf{ex},\mathcal{A}}^1$ ; otherwise, the steps are computationally indistinguishable with that in  $\texttt{Event}_{\mathsf{ex},\mathcal{A}}^1$ . Therefore, if there is such an adversary  $\mathcal{A}$ ,  $\mathcal{C}$  can decide whether  $rk_i$  is uniformly sampled.

Then, we prove the event  $Event_{ex,A}^1$  happens with negligible probability.

**Lemma 12.** If the function  $\mathcal{F}$  is a secure PRF whose output length  $\gamma(\lambda) \geq 2\eta(\lambda)$  and  $\Pi_{\mathsf{zk}}$  has simulation soundness, we have  $\Pr[\mathsf{Event}_{\mathsf{ex},\mathcal{A}}^1] \leq \mathsf{negl}(\lambda)$ , for any non-uniform PPT adversary  $\mathcal{A}$ .

Proof (sketch). We consider an adversary  $\mathcal{C}'$ , which has access to polynomial many oracles  $\mathcal{OF}_i(\cdot) := F_{\lambda}(rk_i, \cdot)$  indexed by *i* where each  $rk_i$  is independently sampled form the uniform distribution.  $\mathcal{C}$ 's goal is to produce a fresh pair  $(\alpha, t)$  s.t.  $F_{\lambda}(rk_i, t) = \alpha$ . It is easy to see if  $\mathcal{F}$  has unpredictability, the advantage of  $\mathcal{C}'$  is negligible. However, we can construct an adversary  $\mathcal{C}'$  with non-negligible advantage, by leveraging an adversary  $\mathcal{A}$  s.t.  $\Pr[\text{Event}_{e_{\mathbf{x}},\mathcal{A}}^1] > \operatorname{negl}(\lambda)$ . This contradiction implies for any non-uniform PPT adversary  $\mathcal{A}$ ,  $\Pr[\text{Event}_{e_{\mathbf{x}},\mathcal{A}}^1] \leq \operatorname{negl}(\lambda)$ .

We now describe the strategy of  $\mathcal{C}'$ . It just mimics all steps in  $\mathtt{Event}^1_{\mathsf{ex},\mathcal{A}}$  with the following exceptions:

- 1. Whenever it needs to sample a new key  $rk_i$ , it adds an index *i* of an oracle  $\mathcal{OF}_i(\cdot)$  which was not added before;
- 2. For answering queries to the oracles  $\mathsf{OP2}_1, \mathsf{OP2}_2$ , it maintains a table  $Tab_i$  for each index *i*. When it needs to evaluate the function  $F_{\lambda}(rk_i, \cdot)$  on an input *t*, it first looks up the table  $Tab_i$ and returns  $\alpha$  if  $(\alpha, t) \in Tab_i$ ; Otherwise, it samples  $\alpha \leftarrow \{0, 1\}^{\gamma(\lambda)}$ , returns  $(\alpha, t)$  and stores it in  $Tab_i$ ;
- 3. For the statement-proof pair  $(x^*, (\alpha^*, t^*, \pi_{\mathsf{zk}}^*))$  submitted by  $\mathcal{A}$ , the last success condition is changed to " $\exists (i), \mathcal{OF}_i(t^*) = \alpha^*$  or  $(\alpha^*, t^*) \in Tab_i$ ".

It is easy to see from the pseudorandomness of  $\mathcal{F}$  the steps above are computationally indistinguishable with that in  $\mathtt{Event}^1_{\mathsf{ex},\mathcal{A}}$ . Therefore, if  $\Pr[\mathtt{Event}^1_{\mathsf{ex},\mathcal{A}}] > \mathsf{negl}(\lambda)$ , after the steps above  $\mathcal{A}$ will also succeed with non-negligible probability. Let us denote the event that  $\mathcal{A}$  succeeds by Succ, then we have

$$\Pr[\operatorname{Succ}|\forall i, (\alpha^*, t^*) \notin Tab_i] + \Pr[\operatorname{Succ}|\exists i, (\alpha^*, t^*) \in Tab_i] > \operatorname{negl}(\lambda).$$

According to the successful conditions, if  $\forall i, (\alpha^*, t^*) \notin Tab_i$ , we have  $\exists i, \mathcal{OF}_i(t^*) = \alpha^*$ . Then, the adversary  $\mathcal{C}'$  can directly takes  $(\alpha^*, t^*)$  to break the unpredictability of  $\mathcal{F}$ .

Now to complete this proof, we show  $\Pr[\operatorname{Succ}|\exists (i, j), (\alpha^*, t^*) \in Tab_i] \leq \operatorname{negl}(\lambda)$ , and thus  $\Pr[\operatorname{Succ}|\forall i, (\alpha^*, t^*) \notin Tab_i] > \operatorname{negl}(\lambda)$ . This result comes from the simulation soundness of the underlying proof system  $\Pi_{zk}$ . Specifically, since every pair  $(\alpha, t)$  in  $Tab_i$  are uniformly sampled and  $\gamma(\lambda) \geq 2\eta(\lambda)$ , the probability that there exists a rk such that  $\alpha = F_{\lambda}(rk, t)$  is negligible. Therefore, the statement  $(x^*, \alpha^*, t^*, r)$  is a false statement. Since we also require that the pair  $(x^*, (\alpha^*, t^*, \pi_{zk}^*))$  is not returned by the oracles  $\operatorname{OP2}_1, \operatorname{OP2}_2$  before,  $\pi_{zk}^*$  is not produced by  $\mathcal{C}'$ . Therefore, the adversary  $\mathcal{A}$  can produce a valid proof for  $(x^*, \alpha^*, t^*, r)$  with probability which is negligible in  $\lambda$ . Thus,  $\Pr[\operatorname{Succ}|\exists i, (\alpha^*, t^*) \in Tab_i] \leq \operatorname{negl}(\lambda)$ .

#### A.4 Proof of Lemma.7

Assume there is a non-uniform PPT adversary  $\mathcal{A}$  which breaks the identifier uniqueness, *i.e.*,  $\mathcal{A}$  outputs a tuple  $(x, \pi = (\alpha, t, \pi_{zk}), w_1^I, w_2^I)$  s.t.  $\pi$  is authenticated by the two identifier witnesses. Then, it contradicts either the collision resistance of the randomness extractor  $\mathcal{E}$ , or the collision resistance of the PRF  $\mathcal{F}$ .

We note when instantiating with collision-resistant PRFs and randomness extactors, besides the extractor seed r and the CRS  $\sigma_{zk}$  of  $\Pi_{zk}$ , the CRS should contains two extra keys  $k_{PRF}$  and  $k_{Ext}$ . By the definition of the identification algorithm, we have  $F_{\lambda,k_{PRF}}(E_{\lambda,k_{Ext}}(w_1^I,r),t) = \alpha$  and  $F_{\lambda,k_{PRF}}(E_{\lambda,k_{Ext}}(w_2^I,r),t) = \alpha$ . If  $E_{\lambda,k_{Ext}}(w_1^I,r) = E_{\lambda,k_{Ext}}(w_2^I,r)$ , it breaks the collision resistance of  $\mathcal{E}$ ; If  $E_{\lambda,k_{Ext}}(w_1^I,r) \neq E_{\lambda,k_{Ext}}(w_2^I,r)$ , it breaks the collision resistance of  $\mathcal{F}$ .'

# B Analysis of PCE scheme in [36]

The recent work [36] presented a framework for plaintext-checkable encryption schemes and gave an instantiation based on the symmetric external Diffie-Hellman assumption (SDXH) assumption.

We that show the SDXH-based construction actually cannot be proven secure under the SDXH assumption, which suggests the framework is also not secure for non-uniform message distributions.

For bilinear pairing groups  $(q, G_1, G_2, G_T, g_1, g_2, e)$ , the SDXH assumption says the DDH problem is intractable in both  $G_1$  and  $G_2$ . Namely, for  $G_\beta$  where  $\beta = 1, 2, a, b, c \leftarrow \mathbb{Z}_q$ ,  $(g^a_\beta, g^b_\beta, g^a_\beta)$  and  $(g^a_\beta, g^b_\beta, g^c_\beta)$  are computationally indistinguishable.

In the SXDH-based PCE scheme, the public parameter consists of  $(q, G_1, G_2, G_3)$ 

 $G_T, g_{1,1}, g_{1,2}, g_{2,1}, g_{2,2}, e)$ , where  $(g_{\beta,1}, g_{\beta,2})$  are random generators of  $G_\beta$  for  $\beta = 1, 2$ . The public key pk is formed by  $(g_{1,1}^{s_1}g_{1,2}^{s_2}, g_{1,1}^{t_1}g_{1,2}^{t_2}, g_{2,1}^{s_1}g_{2,2}^{s_2})$ , while the secret key is  $(s_1, s_2, t_1, t_2, a_1, a_2)$  that are uniformly chosen from  $\mathbb{Z}_q$  at random. To encrypt a message M, it randomly chosen  $r \leftrightarrow \mathbb{Z}_q$ , compute  $W = (g_{2,1}^r, g_{2,2}^r), X = (g_{2,1}^{a_1}g_{2,2}^{a_2})^r M$ , and  $Y = e((g_{1,1}^{s_1}g_{1,2}^{s_2})^r(g_{1,1}^{t_1}g_{1,2}^{t_2})^{\theta}, g_{2,1}^{a_1}g_{2,2}^{a_2})$  where  $\theta = H(W, X)$  for a hash function H. The ciphertext C = (W, X, Y).

Let  $C_0$  be an encryption for  $M_0$  under a randomness  $r_0$ , and  $C_1$  be an encryption for  $M_b$  under a randomness  $r_1$  where b = 0 or b = 1. For each  $C_{\beta} = (W_{\beta}, X_{\beta}, Y_{\beta})$ , we can compute a value in  $G_T$ :  $e((g_{1,1}^{s_1}g_{1,2}^{s_2}(g_{1,1}^{t_1}g_{1,2}^{t_2})^{\theta_{\beta}}, M_{\beta}) = e((g_{1,1}^{s_1}g_{1,2}^{s_2}(g_{1,1}^{t_1}g_{1,2}^{t_2})^{\theta_{\beta}}, X_{\beta})/Y_{\beta}$ , where  $\theta_{\beta} = H(W_{\beta}, X_{\beta})$ . We assume w.l.o.g. that  $M_{\beta} = g_{2,1}^{m_{\beta}}$  for some  $m_{\beta}$ , and  $(g_{1,1}^{s_1}g_{1,2}^{s_2}(g_{1,1}^{t_1}g_{1,2}^{t_2})^{\theta_{\beta}} = g_{1,1}^{f_{\beta}}$ , for some  $f_{\beta}$ . Then, the unlinkability of the PCE scheme implies that the distribution  $(e(g_{1,1},g_{2,1})^{f_0})$ ,

 $e(g_{1,1}, g_{2,1})^{f_0 m_0}, e(g_{1,1}, g_{2,1})^{f_1}, e(g_{1,1}, g_{2,1})^{f_1 m_0})$  and  $(e(g_{1,1}, g_{2,1})^{f_0}, e(g_{1,1}, g_{2,1})^{f_0 m_0})$ 

 $e(g_{1,1}, g_{2,1})^{f_1}$ ,  $e(g_{1,1}, g_{2,1})^{f_1m_1}$ ) are indistinguishable. However, since  $m_0$  and  $m_1$  are not uniformly distributed, we cannot have the result from the SXDH assumption.