Witness Authenticating NIZKs and Applications

Hanwen Feng[1] and Qiang Tang[2(✉)]

[1] Beihang University, Beijing, China
feng_hanwen@buaa.edu.cn
[2] The University of Sydney, Sydney, Australia
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 her 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 [8,26] 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 [35,38], anonymous authentication tools such as group and ring signatures [20,21], 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 [5,11,14,17,32].

Part of the work was done while both authors were at New Jersey Institute of Technology.

T. Malkin and C. Peikert (Eds.): CRYPTO 2021, LNCS 12828, pp. 3–33, 2021.
https://doi.org/10.1007/978-3-030-84259-8_1

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 π showing $x \in L$ was generated by the witness w, i.e., $\mathsf{Identify}(x, w, \pi) \stackrel{?}{=} 1$. It means that each witness w is "committed" to the proof π 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 [31] 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, only the administrators obtain this extra information which remains hidden to everyone else in the system. As pointed out in the recent work of [31], 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 π 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), any of the proofs generated by the attacker will not be identified by w, i.e., $\mathsf{Identify}(x, w, \pi) = 0$.

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 witness-authenticating 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 the 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 application, we introduce a generalization that we only require the $\mathsf{Identify}$ algorithm to take into *a part* of the witness. A bit more formally, we introduce a notion called *identifier witness*, which splits each witness w into an identifier witness w^I and a non-identifier witness w^{NI}. 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 π 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).

Entropic zero-knowledgeness. As a witness-authenticating proof has to convey at least a bit about the identifier witness that makes the identification functionality possible, the conventional zero-knowledgeness 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 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 applied in a larger system, which requires an honestly generated public parameter *pp* (and the witness could 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. 2.1 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 π 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 zero-knowledge definition would restrict adversary from learning side information about the witness via other or directly related proofs. We define another proof oracle to enable an adversary to obtain proofs on related statements. See Sect. 2.2 for details.

Soundness definitions. 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, in the sense that 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,

- For the former property, we formalize it by augmenting 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 defends against a malicious prover trying to "frame" 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 discussion in Sect. 2.3.[1]

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

Basic ideas. 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.

Warm-up constructions. Let us start with a special case where the identifier witness is uniform (or pseudorandom). For example, in group/ring signatures, the identifier witness is each user's secret key. We notice that simulatability can

[1] 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. 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.

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 simply follow the pseudorandomness, which in turn ensures the entropic zero-knowledgeness and unforgeability (the underlying NIZK should 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 already be useful.

A more general solution needs to deal with a general unpredictable sampler. We may apply a strong extractor [29] 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 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. However, as a consequence, this technique can only be applied to the setting that the statements are from a *CRS-independent* sampler.

Full-fledged solution for CRS-dependent samplers. 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. But 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: it is not clear how to force the same random seed to be used for every prover (if we do not want to get into the difficulty of reusable extractor or related-key secure PRF); 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 or certain functionality so that we can prove and further 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. 3.2.

Applications. Our new abstraction of waNIZK can provide a tool for many interesting applications. Here we will showcase three non-trivial applications in hash functions, anonymous authentication revocation, 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 [27], 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. 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 contradicts 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 [30] 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. 4.1.

Auxiliary-input group signatures with verifier-local revocation. 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. One approach to mitigate the threat is to employ leakage-resilient cryptograohic schemes. However, existing VLR group signature schemes [11,12,14,15,32] do not provide any security guarantee when auxiliary information about secret key is leaked. We therefore study the problem of constructing 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 security). Existing constructions leverage either algebraic approaches [12,14,15,32] or generic approaches such as PRFs [11] to realize the mechanism 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.

Plaintext-checkable encryption in the standard model. 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 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 [17,34] 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 [35], we need to prove properties about the plaintext of a PCE ciphertext via NIZKs. Random oracles clearly become unfavorable. Attempts exist [17,33,34] 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 biased message distributions is a natural 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 on uniform messages.

Notations. Throughout the paper, we use λ 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 illustrate other notations and recall definitions of NIZKs and computational entropy in the full version.

2 Syntax and Security Models

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 $\mathsf{Identify}$, such that anyone having a witness $w \in R_L(x)$ can efficiently check whether a proof π 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 $\mathsf{Identify}$ 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 $\mathsf{Identify}$. Now other users who may know the actual *secret* cannot figure out the randomness freshly generated; thus, they will not be able to run $\mathsf{Identify}$. 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^I_L (called an identifier relation of L), $R^I_L((x, w^{NI}), w^I) = 1$, or $w^I \in R^I_L(x)$ for short. Now we only need the identifier witness for the identification algorithm.[2] Formally,

Definition 1 (waNIPS). *Let L be an NP language, and R^I_L be an identifier relation of L. A waNIPS on (L, R^I_L) is defined by four efficient algorithms:*

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

[2] 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.

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

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 2 (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 $\sigma \leftarrow \mathsf{Setup}(1^\lambda), \pi \leftarrow \mathsf{Prove}(\sigma, x, (w^I, w^{NI}))$, the following holds:*

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

2.1 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? The question arises given that the identification functionality is clearly incompatible with the standard zero-knowledgeness.

As a concrete example, consider the range proof system where we use a NIZK to prove a committed integer value m w.r.t. a commitment *com* belongs to the 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 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", and a waNIZK proof system is expected to work for languages admitting such "hard" distributions.

A natural way to describe a distribution is to specify an (adversarial) sampler G which is a non-uniform PPT algorithm and on input a security parameter outputs an element $x \in L_\lambda$ and its witness $(w^I, w^{NI}) \in R_L(x)$[3]. The unpredictability of this sampler can then be quantified by unpredictability entropy [29] of the identifier witness w^I. More precisely, G is k-unpredictable when $\mathbf{H}^{\mathsf{unp}}(W^I|X) \geq k(\lambda)$, 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.

[3] 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.

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:

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

where M_λ 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. Sect. 4.3), 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 $\mathsf{PG} = pp$ 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 this CRS-dependent sampler: the statement itself could be containing auxiliary input of the identifier witness. An extreme example of this auxiliary input could be a valid proof; though the witness is still unpredictable to the adversary, such kind of auxiliary input 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 require the sampler to be unpredictable *for every* CRS. In this way, the hardcoded proof would be automatically ruled out, as there always exists one particular CRS such that an extractor knowing the corresponding trapdoor can recover the witness from the proof, which violates the unpredictability requirement.

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

Definition 3 (Unpredictable sampler). *Let G be a sampler for (L, R_L^I). We say G is k-unpredictable w.r.t. a trusted parameter generation procedure* PG*, if for every CRS σ in the range of* $\mathsf{Setup}(1^\lambda)$*, it holds that*

$$\mathbf{H}^{\mathsf{unp}}(W_\sigma^I | X_\sigma, PP) \geq k(\lambda),$$

where $PP = \mathsf{PG}(1^\lambda)$ *and* $(X_\sigma, W_\sigma^I, W_\sigma^{NI}) \leftarrow G(PP, \sigma)$.

Clearly, the basic requirement is $k = \omega(\log \lambda)$.[4] If we are considering CRS-independent samplers, G simply does not take as input the CRS σ.

2.2 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 zero-knowledge property w.r.t unpredictable samplers. Since now the attacker does not know the witness, 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 σ and a valid proof π using (x, w, σ), can be emulated by a simulator without using the witness. The adaptive counterpart 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 $\mathcal{O}_{\mathsf{P1}}$) takes the tuple (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 $\mathcal{O}_{\mathsf{S1}}$)) 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

[4] We can also measure the unpredictability by HILL entropy [29]. On the one hand, it brings more restrictions on the languages to be proved; On the other hand, for samplers with sufficient HILL entropy we can give more efficient constructions which we explain in details in the full paper.

adaptively obtain multiple proofs, which could be generated from independently sampled statements. Also, seeing a statement x (whose identifier witness is w^I), the adversary can ask the prover to prove another related statement $\bar{x}$, which has the same identifier witness w^I.

Formally, we let the prover oracle $\mathcal{O}_{\mathsf{P1}}$ (or $\mathcal{O}_{\mathsf{S1}}$) be stateful, and augment a pair of new oracles $\mathcal{O}_{\mathsf{P2}}$ and $\mathcal{O}_{\mathsf{S2}}$, which, with access to the states of $\mathcal{O}_{\mathsf{P1}}$ and $\mathcal{O}_{\mathsf{S1}}$, take as inputs an index (that specifies a previously sampled tuple) and an extended sampler EG. EG generates an extended statement $\bar{x}$ (and corresponding non-identifier witness) seeing x, which is associated with the same w^I. However, an arbitrarily extended statement may leak the entire w^I although 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.

Definition 4. *We say* EG *is an admissible extended sampler w.r.t.* G *and* PG*, if there exists a PPT algorithm* $\widetilde{\mathsf{EG}}$*, such that for every* σ*, and any non-uniform PPT* $\mathcal{A}$*, the following holds that* $pp \leftarrow \mathsf{PG}$*,* $(x, w^I, w^{NI}) \leftarrow G(\sigma, pp)$*,* $(\bar{x}, \bar{w}^{NI}) \leftarrow \mathsf{EG}(pp, \sigma, x, w^I, w^{NI})$*,* $\widetilde{x} \leftarrow \widetilde{\mathsf{EG}}(pp, x)$*,* $\Pr[(w^I, \bar{w}^{NI}) \in R_L(\bar{x})] = 1$ *and* $|\Pr[\mathcal{A}(\sigma, pp, \bar{x}, w^I) = 1] - \Pr[\mathcal{A}(\sigma, pp, \widetilde{x}, w^I) = 1]| \leq \mathsf{negl}(\lambda)$*, where the probability is taken over the coin tosses of* PG*,* G*,* EG*,* $\widetilde{\mathsf{EG}}$ *and* $\mathcal{A}$*.*

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

Definition 5 (Entropic ZK). *A waNIPS* Π *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* $\{\mathsf{SimSetup}, \mathsf{SimProve}\}$*, such that for every non-uniform PPT adversary* $\mathcal{A}$*, it holds that*

$$\left| \Pr\left[\begin{array}{l} \sigma \leftarrow \mathsf{Setup}(1^\lambda) \\ pp \leftarrow \mathsf{PG}(1^\lambda); G \leftarrow \mathcal{A}(pp, \boxed{\sigma}) : \\ 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P1}}, \mathcal{O}_{\mathsf{P2}}}(\sigma, pp) \end{array}\right] - \Pr\left[\begin{array}{l} (\sigma, \tau) \leftarrow \mathsf{SimSetup}(1^\lambda) \\ pp \leftarrow \mathsf{PG}(1^\lambda); G \leftarrow \mathcal{A}(pp, \boxed{\sigma}) : \\ 1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{S1}}, \mathcal{O}_{\mathsf{S2}}}(\sigma, pp) \end{array}\right] \right| \leq \mathsf{negl}(\lambda),$$

where the real prover oracles $\mathcal{O}_{\mathsf{P1}}, \mathcal{O}_{\mathsf{P2}}$ *and the simulator oracles* $\mathcal{O}_{\mathsf{S1}}, \mathcal{O}_{\mathsf{S2}}$ *are defined in Fig. 1. The sampler* G *should belong to* $\mathcal{G}$*.* EG *shall be an admissible extended sampler w.r.t.* PG *and* G *(cf. Definition 4).*

Remark 1. Entropic ZK for CRS-independent samplers can be easily obtained by removing the CRS (the boxed σ in Fig. 1) from the input of $\mathcal{A}$ and G; it also suffices in interesting applications and admits more efficient constructions. For detailed elaborations, we defer to the full version.

2.3 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

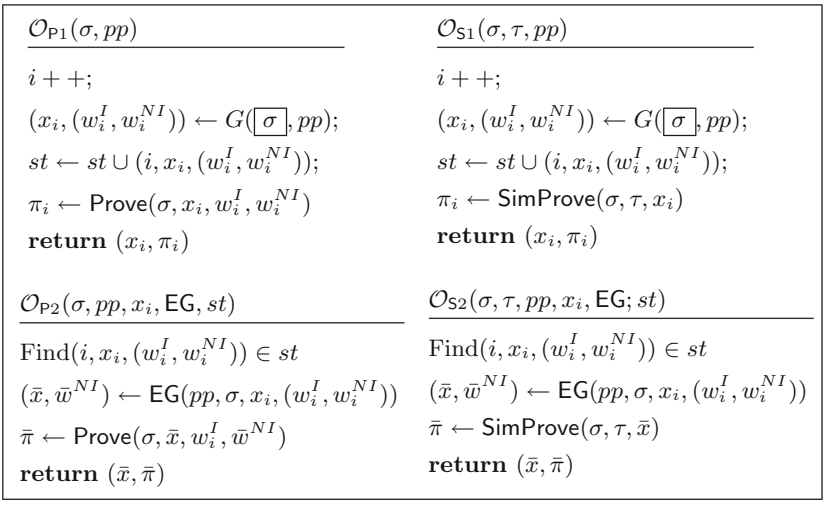

Fig. 1. The oracles. $\mathcal{O}_{\mathsf{P1}}$ (resp. $\mathcal{O}_{\mathsf{S1}}$)and $\mathcal{O}_{\mathsf{P2}}$ (resp. $\mathcal{O}_{\mathsf{S2}}$) share the state st which is initialized to be $\emptyset$. The counter i is initialized to be 0.

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 mimics the binding property and 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. Formally,

Definition 6 (Authenticating knowledge soundness). *We say a waNIPS Π 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 Ext_0 is computationally indistinguishable with the real CRS:*

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

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

$$\Pr\left[\begin{array}{l}(\sigma, \xi) \leftarrow \mathsf{Ext}_0(1^\lambda), (x, \pi) \leftarrow \mathcal{A}(\sigma), (w^I, w^{NI}) \leftarrow \mathsf{Ext}_1(\sigma, \xi, x, \pi) : \\ \mathsf{Verify}(\sigma, x, \pi) = 1 \wedge \left[(w^I, w^{NI}) \notin R_L(x) \vee \mathsf{Identify}(\sigma, x, \pi, w^I) \neq 1\right]\end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

Remark 2. A weaker definition, which we call *authenticating soundness*, only requires the existence of such (w^I, w^{NI}) instead of that $\mathcal{A}$ must know the witness. In some concrete applications of NIZKs such as signatures of knowledge [20],

the knowledge extraction procedure can be done by external primitives such as PKE. Thus, the NIZK does not have to be knowledge sound. The authenticating soundness will suffice for replacing authenticating knowledge soundness in similar cases. This notions will be formalized in the full paper.

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.[5]

Definition 7 (Unforgeability). *Let Π be a waNIPS for (L, R_L^I). We say Π satisfies unforgeability w.r.t.* PG *and a collection of unpredictable samplers $\mathcal{G}$ (cf. Definition 3), if for any non-uniform PPT adversary $\mathcal{A}$, it holds that*

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

where $G \in \mathcal{G}$, and $\mathcal{O}_{\mathsf{P}1}$, $\mathcal{O}_{\mathsf{P}2}$ are prover oracles specified in Fig. 1. Hist *denotes the query-response history of $\mathcal{O}_{\mathsf{P}1}$ and $\mathcal{O}_{\mathsf{P}2}$, and st denotes the set of identifier witnesses generated by all calls (made by $\mathcal{A}$) to $\mathcal{O}_{\mathsf{P}1}$.*

Remark 3. The unforgeability could be weakened and is still useful; for example, for CRS-independent statements can be obtained by removing all boxed items above. On the other hand, in certain applications, we also need to strengthen the unforgeability: it is required that an adversary can neither frame the target identifier witness nor any identifier witness *related to* it. We term the strengthened definition by *related-witness unforgeability*, and show its application to non-malleable hash functions. Details will be given in the full version.

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 σ, directly outputs (x, π), where $(x, w^I, w^{NI}) \leftarrow G_L(1^\lambda)$ and $\pi \leftarrow \mathsf{Prove}(\sigma, x, w^I, w^{NI})$. Given the entropic ZK of π, the identifier witness output by $G_{L'}$ is unpredictable. However, $\mathcal{A}$ can directly output π to break the unforgeability.

[5] Different from the conventional simulation soundness, where the adversary is given simulated proofs, here we provide real proofs, which will be needed in applications.

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

$$\Pr\left[\begin{array}{c}\sigma \leftarrow \mathsf{Setup}(1^\lambda); (x, \pi, w_1^I, w_2^I) \leftarrow \mathcal{A}(\sigma) : \mathsf{Verify}(\sigma, x, \pi) = 1 \wedge \\ \mathsf{Identify}(\sigma, x, \pi, w_1^I) = 1 \wedge \mathsf{Identify}(\sigma, x, (\pi, w_2^I)) = 1\end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

2.4 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. Accordingly, the output of an admissible extended sampler EG shall be computationally indistinguishable with that of the associated $\widetilde{\mathsf{EG}}$, even when the auxiliary information z is given to the distinguisher. In the strengthened definitions, the prover oracle $\mathcal{O}_{\mathsf{P1}}$ and the simulation oracle $\mathcal{O}_{\mathsf{S1}}$ will also return the auxiliary input z for the sampled w^I. Formal definitions appear in the full version.

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 auxiliary-input counterparts.

3 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. For example, when transforming a NIZK to a knowledge-sound NIZK [38], 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.

- *For "zero-knowledgeness".* The tag should not leak any information about the identifier witness except the bit to a verifier knowing the corresponding identifier witness. 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 just for unforgeability, 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 want the 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).

3.1 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 witness is usually a secret key. In this case, the identifier witness could be pseudorandom 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 will be presented in the full version.

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 [4,29] 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 [29] 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 [23,24] (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 [24]. 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, we could simply let the CRS include one single piece of uniform seed. 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 will be presented in the full version.

Unfortunately, once we do not have the luxury that the sampler is *independent* of the CRS, we will need new ideas for the challenges.

3.2 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 depend on 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 $f(\mathtt{params}, w) = \tau$, if $\mathtt{params}$ is from CRS, the adversary can

always find a witness distribution that depends on `params` such that the output τ 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.[6] Regarding the unforgeability, we note it can be realized by adding a simulation-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.

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

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 d-linear based construction [13] can satisfy the auxiliary-input security and the 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 "goodness" 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, $\mathsf{Dec}(sk, \mathsf{DEnc}(pk, m)) = m$. This automatically implies that the encryption

[6] 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 [24]; 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.

function DEnc for each valid pk defines an injective function, i.e., for any $w_1 \neq w_2, \mathsf{DEnc}(pk, w_1) \neq \mathsf{DEnc}(pk, w_2)$. 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 $\Sigma_{\mathsf{de}} = \{\mathsf{K}_{\mathsf{de}}, \mathsf{E}_{\mathsf{de}}, \mathsf{D}_{\mathsf{de}}\}$ (whose formal definition is recalled in the full paper). 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 $R_{L_{\mathsf{de}}}$ s.t. a key pair (pk, sk) is valid iff $R_{L_{\mathsf{de}}}(pk, sk) = 1$.
- A NIZK proof system $\Pi_{\mathsf{zk}} = \{\mathsf{S}_{\mathsf{zk}}, \mathsf{P}_{\mathsf{zk}}, \mathsf{V}_{\mathsf{zk}}\}$ for an NP language

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

The full-fledged construction $\Pi_{\mathsf{CD}} = \{\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Identify}\}$ 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 Π_{zk} and of the DKPE scheme Σ_{de}. Particularly, under an honest pk, $c = \mathsf{E}_{\mathsf{de}}(pk, w^I)$ uniquely determines w^I and thus the proof will not be mis-identified by another identifier witness.

We claim the security of Π_{CD} in the following theorem and present here only a security sketch: the statement being proved by Π_{zk} is formed by (x, pk, c). (1) the knowledge soundness of Π_{zk} ensures one can extract a w^I and $c = \mathsf{E}_{\mathsf{de}}(pk, w^I)$. By the description of $\mathsf{Identify}$, these together imply the authenticating knowledge soundness. (2) When Π_{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-MU-secure with respect to 2^{-k}-hard-to-invert auxiliary inputs, the proof can achieve the entropic ZK.

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 Π_{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 $(pk, c, \pi_{\mathsf{zk}})$ where (pk, c) is an honest encryption of the identifier witness while π_{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 $(pk^*, c^*, \pi_{\mathsf{zk}}^*)$, for a challenge statement x^*, satisfying $c^* = \mathsf{E}_{\mathsf{de}}(pk^*, w^I)$, and $\mathcal{B}$ can further leverage the knowledge extractor of Π_{zk} to extract w^I, which is the plaintext of these deterministic encryptions, from π_{zk}.

$\mathsf{Setup}(1^\lambda)$

$\sigma_{\mathsf{zk}} \leftarrow \mathsf{S}_{\mathsf{zk}}(1^\lambda)$ // generate a CRS of the underlying NIZK
return $\sigma = \sigma_{\mathsf{zk}}$

$\mathsf{Prove}(\sigma, x, (w^I, w^{NI}))$

$(pk, sk) \leftarrow \mathsf{K}_{\mathsf{de}}(1^\lambda)$ // generate the public key and the secret key
$c \leftarrow \mathsf{E}_{\mathsf{de}}(pk, w^I)$ // encrypt the identifier witness under pk
$\pi_{\mathsf{zk}} \leftarrow \mathsf{P}_{\mathsf{zk}}(\sigma_{\mathsf{zk}}, (x, pk, c), (w^I, w^{NI}, sk))$
// prove $x \in L \wedge (pk, c)$ are well-formed
return $\pi = (pk, c, \pi_{\mathsf{zk}})$

$\mathsf{Verify}(\sigma, x, \pi)$

$b \leftarrow \mathsf{V}_{\mathsf{zk}}(\sigma_{\mathsf{zk}}, (x, pk, c), \pi_{\mathsf{zk}})$ // check the validity of the proof π_{zk}
return b

$\mathsf{Identify}(\sigma, x, \pi, w^I)$

$c' \leftarrow \mathsf{E}_{\mathsf{de}}(pk, w^I)$ // encrypt the identifier witness under the public key
if $(c = c')$ **then return** 1 **else return** 0

Fig. 2. The full-fledged construction.

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

Due to the lack of space, we defer detailed proofs in the full version.

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

- *Π_{CD} satisfies the authenticating (knowledge) soundness, if Π_{zk} satisfies the (knowledge) soundness;*
- *Π_{CD} satisfies the identifier uniqueness, if Π_{zk} is sound, and the DPKE satisfies perfect correctness;*
- *Π_{CD} satisfies the entropic ZK w.r.t. all k-unpredictable samplers, if Π_{zk} is zero-knowledge, and Σ_{de} is PRIV- IND-MU-secure with respect to 2^{-k}-hard-to-invert auxiliary inputs.*[7]

[7] A basic requirement is $k = \omega(\log \lambda)$ s.t. it is possible to have such a DPKE scheme.

– *Π_{CD} satisfies the (related-witness) unforgeability w.r.t. all k-unpredictable samplers, if Π_{zk} is a simulation-extractable NIZK, and Σ_{de} is PRIV- IND-MU-secure with respect to 2^{-k}-hard-to-invert auxiliary inputs.*

Sketch of instantiation. Since the underlying DPKE scheme Σ_{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, ν is a negligible function in λ, 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 Π_{CD} should be k-unpredictable for some $k \geq 2\log q + \omega(\log \lambda)$.[8] Regarding the underlying simulation-extractable NIZK Π_{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 [26] or the LWE assumption [37], and we can add simulation extractability to them using standard tools including one-way functions and public-key encryptions as noted in [38]. In addition, since the tag generation procedure is algebraic (and Groth-Sahai-friendly), we can leverage the (simulation-extractable) Groth-Sahai proof system [28] to instantiate Π_{zk}, if the statement $x \in L$ that we wish to prove is also Groth-Sahai-friendly.

4 Applications

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

4.1 Non-malleable (Perfectly One-Way) Hash Functions from Standard Assumptions

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 [27], 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 "maul" 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

[8] In certain applications, we may be interested 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.

for many of their interesting applications, such as instantiating random oracles in Bellare-Rogaway encryption [6,9].

Unfortunately, both perfectly one-way hash and non-malleable hash (with general auxiliary inputs) have no construction from any efficiently falsifiable assumption [3,9,18,24].[9] In particular, Boldyreva *et al.* [9] presented constructions of non-malleable hash from perfectly one-way hash functions [18] and simulation-extractable NIZKs [38], thus directly inherits the non-efficiently falsifiable assumption from [18]; Baecher *et al.* [3] showed another construction for non-malleable hash, but it requires a random oracle. Recall that the main motivation of non-malleable 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 [30] 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 functions 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.

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.
- $\mathsf{H}(hk, s)$. On inputs a key hk and an input s output a hash value y.
- $\mathsf{HVf}(hk, s, y)$. 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. ϵ-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 ϵ-hard-to-invert w.r.t. S, and for any non-uniform PPT adversary $\mathcal{A}$, it holds that

$$\Pr\left[\begin{array}{c} hk \leftarrow \mathsf{HK}(1^\lambda), s_0 \leftarrow S_\lambda, s_1 \leftarrow \{0,1\}^{|s_0|}, b \leftarrow_\$ \{0,1\}, \\ y \leftarrow \mathsf{H}(hk, s_b), b' \leftarrow \mathcal{A}(hk, y, \mathsf{hint}(hk, s_0)) : b = b' \end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

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

$$\Pr\left[\begin{array}{c} hk \leftarrow \mathsf{HK}(1^\lambda), (s, s', y) \leftarrow \mathcal{A}(hk) : \\ s \neq s' \wedge \mathsf{HVf}(hk, s, y) = \mathsf{HVf}(hk, s', y) = 1 \end{array}\right] \leq \mathsf{negl}(\lambda)\,.$$

[9] Under standard assumptions, the only existing perfectly one-way functions with auxiliary inputs [7] does not enjoy the collision resistance; and the only non-malleable hash (also given in [9]) is only secure against a very special class of auxiliary input.

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 [27], and OAEP [10]. Informally, non-malleability requires that an adversary, seeing a hash value $y = \mathsf{H}(hk, s)$ and an auxiliary input $\mathsf{hint}(hk, s)$, cannot find another y^* whose pre-image is meaningfully *related to* s. The formal definition appears in the full paper. We note the "relation" between the pre-images is described a transformation set Φ, namely, s' is Φ-related to s if $s' = \phi(s)$ for some $\phi \in \Phi$. The non-malleability is defined w.r.t. a transformation set Φ rather than any transformation ϕ, 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 $\Phi^{\mathsf{srs}}_{\mathsf{brs}}$) developed in [22], which are the currently most general yet achievable class (see the full paper).

Construction. Observe that non-malleable (perfectly one-way) hash has three security requirements and a verifier algorithm on each input-output pair. 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 $\mathsf{COM} = \{\mathsf{K}_{\mathsf{com}}, \mathsf{C}_{\mathsf{com}}\}$ be a commitment scheme, and let $\Pi_{\mathsf{wa}} = \{\mathsf{S}_{\mathsf{wa}}, \mathsf{P}_{\mathsf{wa}}, \mathsf{V}_{\mathsf{wa}}, \mathsf{I}_{\mathsf{wa}}\}$ be a WA-NIZK for an NP language $L_{\mathsf{nm}} := \{(c, k_{\mathsf{com}}); (s, r) : c = \mathsf{C}_{\mathsf{com}}(k_{\mathsf{com}}, s; r)\}$, in which s *is the identifier witness*. Here we require Π_{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 $\Phi^{\mathsf{srs}}_{\mathsf{brs}}$. We present the detailed description in Fig. 3.

$\mathsf{HK}(1^\lambda)$

$\sigma_{\mathsf{wa}} \leftarrow \mathsf{S}_{\mathsf{wa}}(1^\lambda)$ and $k_{\mathsf{com}} \leftarrow \mathsf{K}_{\mathsf{com}}(1^\lambda)$; **return** $hk = (\sigma_{\mathsf{wa}}, k_{\mathsf{com}})$

$\mathsf{H}(hk, s)$

$c_{\mathsf{com}} \leftarrow \mathsf{C}_{\mathsf{com}}(k_{\mathsf{com}}, s; r)$; $\pi_{\mathsf{wa}} \leftarrow \Pi_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (c_{\mathsf{com}}, k_{\mathsf{com}}), (s, r))$; **return** $y = (c_{\mathsf{com}}, \pi_{\mathsf{wa}})$

$\mathsf{HVf}(hk, s, y)$

return 1 **if** $\mathsf{V}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (c_{\mathsf{com}}, k_{\mathsf{com}}), \pi_{\mathsf{wa}}) = 1 \wedge \mathsf{I}_{\mathsf{wa}}(\sigma_{\mathsf{wa}}, (c_{\mathsf{com}}, k_{\mathsf{com}}), \pi_{\mathsf{wa}}, s) = 1$

Fig. 3. Non-malleable Hash from commitment+ waNIZKs

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. For detailed proofs, we defer them to the full version.

Theorem 2. *$\mathcal{H}$ satisfies the perfect one-wayness w.r.t. ϵ-hard-to-invert auxiliary inputs, collision resistance, and non-malleability w.r.t. the transformation set $\Phi^{\mathsf{srs}}_{\mathsf{brs}}$ and ϵ-hard-to-invert auxiliary inputs, if the commitment scheme COM satisfies computationally hiding, and Π_{wa} satisfies the identifier uniqueness, the entropic ZK and the related-witness unforgeability w.r.t. the transformation set $\Phi^{\mathsf{srs}}_{\mathsf{brs}}$ and all $(-\log \epsilon)$-unpredictable samplers.*

4.2 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 [25]), 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 [36]. 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 [12, 14, 15, 32]) 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.

The definitions. A VLR group signature scheme Σ_{gs} is defined by a tuple of three PPT algorithms.

- $\mathsf{GS.KeyGen}(1^\lambda, n)$. It outputs a group public key gpk, and for each user $i \in [n]$, outputs the secret key $\mathbf{gsk}[i]$ along with the revocation token $\mathbf{grk}[i]$.
- $\mathsf{GS.Sign}(gpk, \mathbf{gsk}[i], m)$. It outputs a valid signature ϑ for m under gpk.
- $\mathsf{GS.Verify}(gpk, RL, \vartheta, m)$. It returns either 1 indicating that ϑ 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 Σ_{gs} is correct, if for $(gpk, \mathbf{gsk}, \mathbf{grk}) \leftarrow \mathsf{GS.KeyGen}(1^\lambda)$, 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 $\mathsf{GS.Verify}(gpk, \emptyset, \vartheta, m) = 1$ and $\mathsf{GS.Verify}(gpk, \mathbf{grk}[i^*], \vartheta, m) = 0$, the signer of ϑ will be traced to user i^*.

A VLR group signature scheme should satisfy the *anonymity* and the *traceability*. In the following, we briefly introduce the auxiliary-input counterparts of them, and the formal definitions are presented in the full paper. 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.

- *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 $\mathbf{grk}[i]$. When considering the auxiliary-input anonymity, the adversary is further allowed to see a hard-to-invert function on $(gpk, \mathbf{gsk})$.
- *Traceability* captures that any non-uniform PPT adversary $\mathcal{A}$ can neither produce a valid signature-message pair (ϑ, 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 $\mathbf{grk}[i]$. When considering the auxiliary-input anonymity, the adversary is further allowed to see a hard-to-invert function on $(gpk, \mathbf{gsk})$.

The 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 ϑ was created by $\mathbf{gsk}[i]$. If ϑ 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.

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[10] is leaked to adversaries. More formally, let $\Sigma_{\mathsf{sig}} = \{\mathsf{K}_{\mathsf{sig}}, \mathsf{S}_{\mathsf{sig}}, \mathsf{V}_{\mathsf{sig}}\}$ be a standard digital signature scheme. Let $\Pi_{\mathsf{wa}} = \{\mathsf{S}_{\mathsf{wa}}, \mathsf{P}_{\mathsf{wa}}, \mathsf{V}_{\mathsf{wa}}, \mathsf{I}_{\mathsf{wa}}\}$ be a waNIZK for the following language: $\mathsf{L}_{\mathsf{VLR}} : \{(vk_{\mathsf{sig}}, m); (ID, \mathsf{Sig}) : \mathsf{V}_{\mathsf{sig}}(vk_{\mathsf{sig}}, \mathsf{Sig}, ID)\}$, where ID is the identifier witness. The formal description of the VLR group signature Σ_{gs} will be presented in the full paper.

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

Definition 9. *We say $\mathcal{F}$ is admissible w.r.t. Σ_{gs}, if for every σ_{wa} in the range of $\mathsf{S}_{\mathsf{wa}}(1^\lambda)$, and every $(vk_{\mathsf{sig}}, sk_{\mathsf{sig}})$ in the range of $\mathsf{K}_{\mathsf{sig}}(1^\lambda)$, it holds that*

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

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

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 Σ_{sig} be a standard-model digital signature scheme satisfying EU-CMA security, Π_{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}. Σ_{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.*

[10] 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.

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

4.3 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 [17,33,34] are either relying on random oracles or only working for uniform message distributions.[11] 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.[12] 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.

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:

[11] We note that the recent scheme [33] 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 the full paper.

[12] The plain-text equality tester, presented in [39], 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.

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

Correctness requires that for every λ and m, $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, $c \leftarrow \mathsf{Enc}(pk, m)$, $\Pr[\mathsf{Dec}(sk, c) = 1 \wedge \mathsf{PCheck}(pk, c, m) = 1] = 1$, 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.
– *Checking soundness:* No efficient adversary can generate a ciphertext c and a plaintext m such that c cannot be decrypted to m but $\mathsf{PCheck}(pk, c, m) = 1$.
– *k-unlinkability:* It is infeasible to decide whether two ciphertexts encrypt the same message, when the message is from a message distribution whose min-entropy is larger than k and the message is not available to adversaries.

Formal definitions are recalled in the full paper.

The 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 $\Sigma_{\mathsf{pke}} = \{\mathsf{K}_{\mathsf{pke}}, \mathsf{E}_{\mathsf{pke}}, \mathsf{D}_{\mathsf{pke}}\}$ be a PKE scheme, and let $\Pi_{\mathsf{wa}} = \{\mathsf{S}_{\mathsf{wa}}, \mathsf{P}_{\mathsf{wa}}, \mathsf{V}_{\mathsf{wa}}, \mathsf{I}_{\mathsf{wa}}\}$ be a waNIZK for the following language: $L_{\mathsf{PCE}} := \{(c, pk); (m, r) : c = \mathsf{E}_{\mathsf{pke}}(pk, m; r)\}$ where the message m is the identifier witness. To encrypt a message m, our PCE scheme first encrypts it using Σ_{pke}, and uses Π_{wa} to prove the ciphertext is well-formed, where the CRS for Π_{wa} is a part of the public key. Everyone can check whether a ciphertext $(c_{\mathsf{pke}}, \pi_{\mathsf{wa}})$ encrypts a particular message m by running the identification algorithm I_{wa} on π_{wa} and m. We defer the formal construction Σ to the full version.

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

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

Proof (sketch). The checking completeness follows the authentication soundness of Π_{wa}, and the checking soundness is implied by the identifier uniqueness of Π_{wa}. Regarding the k-unlinkability, we argue the distribution $G = \{(x = (c, pk), w^I = m, \perp) : m \leftarrow M_\lambda; c \leftarrow \mathsf{Enc}(pk, m)\}$ for L_{PCE} is k-unpredictable w.r.t. an honest key generation $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, if the min-entropy of M_λ 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 $\bar{G} = \{(x = (c, pk), y, \perp) : m, y \leftarrow M_\lambda; c \leftarrow \mathsf{Enc}(pk, m)\}$. Ensured by the IND-CPA security of the PKE scheme, $\bar{G}$ 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. We defer the formal proof to the full version.

References

1. 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)
2. Alamélou, Q., Blazy, O., Cauchie, S., Gaborit, P.: A code-based group signature scheme. Des. Codes Cryptogr. **82**(1–2), 469–493 (2017)
3. Baecher, P., Fischlin, M., Schröder, D.: Expedient non-malleability notions for hash functions. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 268–283. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_18
4. Barak, B., Dodis, Y., Krawczyk, H., Pereira, O., Pietrzak, K., Standaert, F.-X., Yu, Yu.: Leftover hash lemma, revisited. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 1–20. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_1
5. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74143-5_30
6. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: CCS, pp. 62–73. ACM (1993)
7. Bellare, M., Stepanovs, I.: Point-function obfuscation: a framework and generic constructions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9563, pp. 565–594. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_21
8. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: STOC, pp. 103–112. ACM (1988)
9. Boldyreva, A., Cash, D., Fischlin, M., Warinschi, B.: Foundations of non-malleable hash and one-way functions. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 524–541. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10366-7_31
10. Boldyreva, A., Fischlin, M.: On the security of OAEP. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 210–225. Springer, Heidelberg (2006). https://doi.org/10.1007/11935230_14
11. Boneh, D., Eskandarian, S., Fisch, B.: Post-quantum EPID signatures from symmetric primitives. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 251–271. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_13
12. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: CCS, pp. 168–177. ACM (2004)
13. Brakerski, Z., Segev, G.: Better security for deterministic public-key encryption: the auxiliary-input setting. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 543–560. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_31
14. Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: CCS, pp. 132–145. ACM (2004)

15. 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)
16. Brzuska, C., Mittelbach, A.: Indistinguishability obfuscation versus multi-bit point obfuscation with auxiliary input. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 142–161. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45608-8_8
17. Canard, S., Fuchsbauer, G., Gouget, A., Laguillaumie, F.: Plaintext-checkable encryption. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 332–348. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_21
18. Canetti, R.: Towards realizing random oracles: hash functions that hide all partial information. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052255
19. Canetti, R., Micciancio, D., Reingold, O.: Perfectly one-way probabilistic hash functions (preliminary version). In: STOC, pp. 131–140. ACM (1998)
20. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_5
21. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_22
22. Chen, Yu., Qin, B., Zhang, J., Deng, Y., Chow, S.S.M.: Non-malleable functions and their applications. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016. LNCS, vol. 9615, pp. 386–416. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49387-8_15
23. Dachman-Soled, D., Gennaro, R., Krawczyk, H., Malkin, T.: Computational extractors and pseudorandomness. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 383–403. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_22
24. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: STOC, pp. 621–630. ACM (2009)
25. Faust, S., Hazay, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A.: Signature schemes secure against hard-to-invert leakage. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 98–115. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_8
26. 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)
27. Fischlin, M.: Security of NMAC and HMAC based on non-malleability. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 138–154. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79263-5_9
28. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
29. Hsiao, C.-Y., Lu, C.-J., Reyzin, L.: Conditional computational entropy, or toward separating pseudoentropy from compressibility. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 169–186. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_10
30. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. IACR Cryptol. ePrint Arch. **2020**, 1003 (2020)

31. Kreuter, B., Lepoint, T., Orrù, M., Raykova, M.: Anonymous tokens with private metadata bit. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 308–336. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_11
32. Libert, B., Vergnaud, D.: Group signatures with verifier-local revocation and backward unlinkability in the standard model. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 498–517. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10433-6_34
33. Ma, S., Huang, Q.: Plaintext-checkable encryption with unlink-CCA security in the standard model. In: Heng, S.-H., Lopez, J. (eds.) ISPEC 2019. LNCS, vol. 11879, pp. 3–19. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34339-2_1
34. Ma, S., Mu, Y., Susilo, W.: A generic scheme of plaintext-checkable database encryption. Inf. Sci. **429**, 88–101 (2018)
35. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC, pp. 427–437. ACM (1990)
36. 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)
37. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 89–114. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_4
38. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust noninteractive zero knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_33
39. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: FOCS, pp. 600–611. IEEE Computer Society (2017)