On Black-Box Extensions of Non-Interactive Zero-Knowledge Arguments, and Signatures Directly from Simulation Soundness

Masayuki Abe[1], Miguel Ambrona[1], and Miyako Ohkubo[2]

[1] Secure Platform Laboratories, NTT Corporation, Japan
{masayuki.abe.cp, miguel.ambrona.fu}@hco.ntt.co.jp
[2] Security Fundamentals Laboratory, CSR, NICT, Japan
m.ohkubo@nict.go.jp

Abstract. Highly efficient non-interactive zero-knowledge arguments (NIZK) are often constructed for limited languages and it is not known how to extend them to cover wider classes of languages in general. In this work we initiate a study on black-box language extensions for conjunctive and disjunctive relations, that is, building a NIZK system for $\mathcal{L} \diamond \hat{\mathcal{L}}$ (with $\diamond \in \{\wedge, \vee\}$) based on NIZK systems for languages $\mathcal{L}$ and $\hat{\mathcal{L}}$. While the conjunctive extension of NIZKs is straightforward by simply executing the given NIZKs in parallel, it is not known how disjunctive extensions could be achieved in a black-box manner. Besides, observe that the simple conjunctive extension does not work in the case of simulation-sound NIZKs (SS-NIZKs), as pointed out by Sahai (Sahai, FOCS 1999). Our main contribution is an impossibility result that negates the existence of the above extensions and implies other non-trivial separations among NIZKs, SS-NIZKs, and labelled SS-NIZKs.
Motivated by the difficulty of such transformations, we additionally present an efficient construction of signature schemes based on unbounded simulation-sound NIZKs (USS-NIZKs) for any language without language extensions.

1 Introduction

1.1 Background

A non-interactive zero-knowledge argument system (NIZK) [8] is a beneficial building block for constructing a wide variety of cryptographic schemes and protocols. Very roughly, given an NP language $\mathcal{L}$ for certain relation R, i.e., $\mathcal{L} := \{x \mid \exists w \text{ s.t. } R(x, w) = 1\}$, a NIZK argument system for $\mathcal{L}$ allows a *prover* (who owns a pair x, w such that $R(x, w) = 1$) to convince a *verifier* of the fact that $x \in \mathcal{L}$. The communication between the two parties is unilateral and the verifier learns no new information about possible witnesses for x, except the fact that there exists one. (That is enforced by the presence of a *simulator* which, without any witness for x, produces an output that is indistinguishable from the proof produced by a real prover.) A NIZK system is said to be *correct* if an honest prover can always convince a verifier of a true statement. On the other hand, the system is said to be *sound* if a (possibly malicious) prover cannot convince an honest verifier of a false statement (except with negligible probability). A simulation-sound NIZK (SS-NIZK) [50] is a strengthening of NIZK whose soundness holds even in the presence of simulated proofs on arbitrary statements. SS-NIZKs receive much attention due to their usefulness in the construction of public-key encryption schemes secure against adaptive chosen message attacks [50]. Another application of SS-NIZKs is on building Threshold Password-Authenticated Key Exchange [42]. Furthermore, they have recently been used to build tightly secure CCA2 encryption in the multi-challenge and multi-user setting [37] or to design tightly secure signature schemes [29].

Thanks to a considerable and prolonged effort by the community of cryptographers, there exist NIZK systems for NP-complete languages in several settings, e.g., [8, 19, 25], and general constructions have been designed to strengthen them to SS-NIZK, e.g., [17, 50]. Some of these settings provide very efficient NIZK systems: Schnorr proofs [43], Groth-Sahai proofs [27], Quasi-Adaptive NIZKs (QA-NIZKs) [32] that are designed for particular languages. However, when NIZK systems are used for building advanced cryptographic schemes and protocols, it is frequently assumed that a convenient language is covered by the NIZK, or that the system can be extended to support such a language. For instance, the general transformations from NIZK to unbound SS-NIZK (USS-NIZK) in [17, 29] (see Definition.2.5) require the NIZK support a disjunctive statement combining two instances of certain specific languages.

Given the relevance of these works, where additional assumptions are made on the languages supported by the NIZK systems, we study *black-box language extensions* of NIZKs for conjunctive and disjunctive relations. More concretely, we consider the question of whether for some language $\hat{\mathcal{L}}$, there exists a generic compiler that on input a NIZK system for language $\mathcal{L}$, produces a NIZK system for $\mathcal{L} \diamond \hat{\mathcal{L}}$, where $\diamond \in \{\wedge, \vee\}$. Many non black-box techniques for disjunctive language extension can be found in the literature, e.g., [2, 13, 14, 21, 24, 41, 46], but not much is known in the case of black-box extensions, which are a relevant area of study due to their potential for building efficient and more advanced cryptographic primitives. In the settings where generic NIZKs for NP are not very efficient, using a generic transformation from a less expressive (but more efficient) NIZK may be a better approach than going through the Karp reduction.

Due to the commodity of NIZK in cryptographic design, there are strong demands to construct efficient NIZKs from various assumptions, but this is not an easy task. For instance, NIZKs for NP-complete languages based on lattice-based assumptions were known only in relaxed scenarios such as designated-verifier NIZKs [12] and preprocessing NIZKs [36], while very efficient NIZKs for limited lattice-related languages are known in the standard common reference string (CRS) model [5, 45, 48]. Very recently, Peikert and Shiehian finally developed a NIZK for NP based on *learning with errors* (LWE) [44]. A natural question is whether such efforts have to be done every time we want to use a new assumption. A widely useful abstraction such as black-box constructions aims to reduce the burden of cryptographic design. Its importance is remarkable in post-quantum cryptography, where several new assumptions such as isogeny-based assumptions [47, 51] and multi-variable problems [56] are under investigation. Our impossibility results justify the design and study of approaches relying on particular properties of the underlying assumptions.

Note that some black-box language extensions are straightforward, e.g., a conjunctive extension can be achieved by computing both proofs and concatenating them. Others are more involved, for example, a similar approach fails in the case of disjunctive language extension, or, as pointed out by Sahai [50], the conjunctive extension does not work in the case of USS-NIZKs. Contrary to the case of conjunctive extensions, generic methods for achieving disjunction of languages in the framework of NIZKs are not known. A black-box disjunctive language extension could be a great tool for building more advanced and secure NIZK systems. Observe that NIZKs for disjunctive languages have a vast number of applications. Among them, an important example is the framework of electronic voting [14], where disjunction is used to argue that a vote is valid. In general, it is very useful in any secure function evaluation scenario where a proof of a disjunctive relation is used to guarantee that the input to each wire is either 0 or 1. Furthermore, disjunctive relations are used as a building block for achieving tight security (they often simplify the simulation in the security reduction).

1.2 Our Results

Our main contribution is a series of (im)possibility results about black-box language extensions among different types of NIZK systems. In the case of impossibility, the constructions ruled out by this work correspond to what we would normally think of as *"black-box language extensions"*.

- There exists no generic compiler that, given two NIZK systems for hard languages $\mathcal{L}$ and $\hat{\mathcal{L}}$, outputs a NIZK system for $\mathcal{L} \vee \hat{\mathcal{L}}$. This holds even if we are given stronger types of NIZKs, i.e. labelled or simulation-sound, as building blocks. This justifies existing non black-box approaches.
- It is hard to extend, in a black-box manner, a simulation-sound NIZK to cover conjunctive and disjunctive languages, or to support labels.
- Unbound simulation-soundness is hard to obtain in a black-box manner. This justifies that black-box constructions of simulation-sound NIZKs from standard NIZKs [50] are *bounded* in the number of simulations.

As it is common in all standard black-box separations, our impossibility results do not apply when extra ingredients are available. It is indeed an interesting open problem to understand what the minimal additional functionalities to achieve certain constructions are.

Additionally, we provide a construction of a secure digital signature scheme from any USS-NIZK for a hard language in NP that *does not require language extensions*. This construction is motivated by our previous impossibility results and the observation that all constructions of signatures from USS-NIZK require the underlying language be extended to support specific relations (see Table 1).

Figure 1 illustrates a more precise summary of our separations and contributions, which we further describe in the rest of this section.

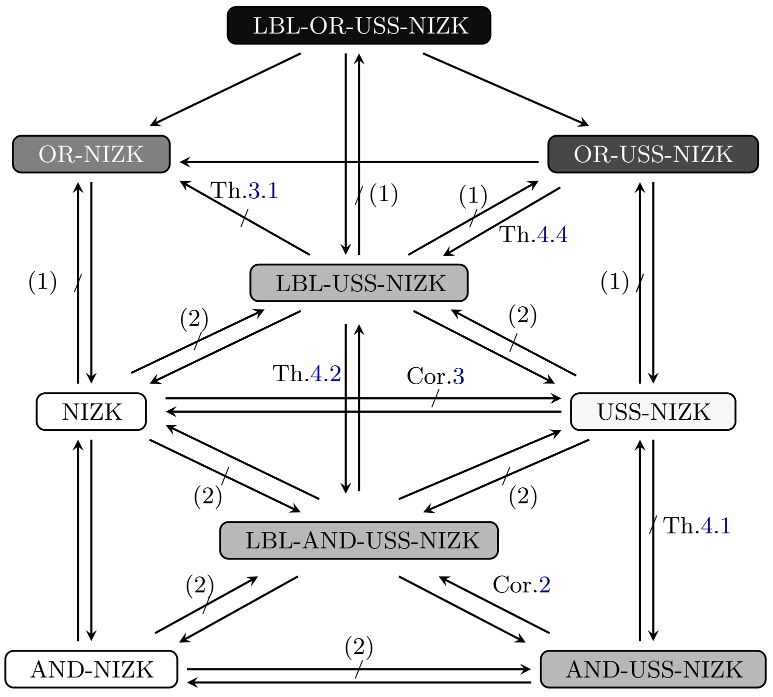

Fig. 1: Relations between variations of NIZK. Non-labelled edges correspond to straightforward black-box constructions. Separations labelled as (1) are implied by Theorem 3.1 (see Corollary 1). Those labelled as (2) are implied by Theorem 4.1 and hold in the *full-verification model* (see Definition 4.1).

Impossibility of Black-Box Disjunctive Extension of various NIZKs. As our first contribution, we show that there is no fully black-box disjunctive language extension for NIZKs. We show this result in a stronger form by proving the absence of reductions from a labelled USS-NIZK system (see Definition.2.3) for $\mathcal{L}$ to a NIZK scheme for $\mathcal{L} \vee \hat{\mathcal{L}}$ (for any $\hat{\mathcal{L}}$). (Note that we focus on labelled USS-NIZKs for its generality.) To explain the core idea of our argument, let us define a *legitimate crs* as a crs generated with the underlying NIZK's crs generation algorithm. Roughly, our proof goes as follows: i) we show that the prover algorithm of the disjunctive extension cannot invoke the underlying NIZK's prover on legitimate crs's (or otherwise the resulting NIZK will not be zero-knowledge); ii) we then argue that because all calls to the underlying prover must be on non-legitimate crs's, very roughly, their trapdoor is known to the prover of the extended NIZK and thus, soundness is compromised. In Section 3 we formalize the previous intuition and rigorously consider other missing cases.

A bit more formally, we follow the *oracle separation paradigm*, cf., [9,23,31,49,53] where we construct an oracle O relative to which there exists a language $\mathcal{L}$ and a secure labelled USS-NIZK system L for it, but for any language $\hat{\mathcal{L}}$, there does not exist a NIZK system M for $\mathcal{L} \vee \hat{\mathcal{L}}$ that is zero-knowledge and sound at the same time. Our contribution also includes a novel approach in the construction of an adversary against simulation soundness, exploiting the simulability of the NIZK in a reverse manner: simulating the zero-knowledge simulator with a real prover, as we elaborate below. This technique bears similarity with the *simulatable adversary paradigm* in [22] that exploits the duality of the zero-knowledge simulator and the real prover to construct a meta-reduction [10,15]. In our approach, we let the adversary simulate the oracle so that a no-instance of the language can look like a yes-instance with a certain witness. This can be done by redefining the language in such a way that a no-instance and a yes-instance are swapped (we call this *instance swapping*).

The way we simulate the oracle simplifies the analysis and results in eliminating the use of PSPACE power from the adversary, which used to be essential in standard approaches from the literature. We believe this new technique is of independent interest and could be applicable to other impossibility results.

Impossibility of Black-Box Conjunctive Extension of USS-NIZK. It is remarkable that a *conjunctive language extension* is hard to achieve in a black-box way in the case of USS-NIZKs. Specifically, in Section 4, we show that there is no fully black-box reduction [31,49] from a USS-NIZK system L for a hard language $\mathcal{L}$ to a USS-NIZK system M for the extended language $\mathcal{L} \wedge \hat{\mathcal{L}}$, for any arbitrary hard language $\hat{\mathcal{L}}$. (We refer to Theorem 4.1 for a more formal statement.) Here, a hard language is, in short, a language that constitutes a promise problem [18,52] consisting of a pair of disjoint, efficiently sampleable, and indistinguishable languages, $\mathcal{L}$ and $\mathcal{C}$ (see Definition 2.1). Our result also applies to the case where the extended part of the language $\hat{\mathcal{L}}$ is trivial (i.e., in BPP) as long as the inverse of its size is negligible in the security parameter.

A very high level view of our proof strategy is similar to the one for the impossibility of disjunctive extensions. However, since the simulation soundness game is interactive, when oracle queries from $\mathsf{M.PrvSim}$ run by the challenger cannot be seen by the adversary, it is more difficult to collect enough information for producing a forgery and the details of the proof differ considerably. Another important difference from the case of disjunction is that our impossibility result about the conjunctive extension is limited to what we introduce as the *full-verification model* (Definition 4.1). Namely, every proof that is created internally with the prover algorithm must then be verified by the verification algorithm.

Implications and More. Our two impossibility results, in combination with a simple analysis, allow us to discover other impossibility relations (see Figure 1). A remarkable one is the impossibility of fully black-box construction of USS-NIZKs from NIZKs (Corollary 3). Such an impossibility (even in the *full-verification model*) enlightens the essential difference between bounded and unbounded simulation soundness in the context of NIZKs.

Another remarkable result is fully black-box constructions of *labelled* USS-NIZKs for $\mathcal{L}$ from USS-NIZK for $\mathcal{L} \wedge \hat{\mathcal{L}}$ (Theorem 4.3) or $\mathcal{L} \vee \hat{\mathcal{L}}$ (Theorem 4.4).

A heuristic way of attaching a label would be to involve it into a hash function used somewhere in the proof function. While it is possible for particular types of constructions such as those using Fiat-Shamir transformation [20], it is not necessarily trivial in other cases such as Groth-Sahai proofs [27] and Quasi-adaptive NIZKs [32] with structure-preserving property [4], even if the label is left as a non-group string. Our construction can be seen as a feasibility result: labelling is achievable in a black-box manner given USS and support for extended (conjunctive or disjunctive) languages. If either of the properties is not provided, labelling is not black-box achievable as shown in Figure 1.

Construction of Signatures without Language Extension. Motivated by our previous results, we show that any USS-NIZK for any hard language in NP can be used by itself *without language extensions* as a secure digital signature scheme. Our construction retains almost the same computation and space complexity and hence has a practical value. Concretely, given a USS-NIZK for any hard language $\mathcal{L}$, we construct a signature scheme that is unforgeable against adaptive chosen message attacks. We emphasize that our result does not require $\mathcal{L}$ support particular relations, which was required by related works on building signatures from USS-NIZKs, e.g., [26, 34] (see Table 1). That is a sharp difference from previous works, because our impossible results suggest that such specific relations in the language cannot be achieved in a black-box way. Furthermore, the only additional building block (used to create a signature scheme for arbitrary long messages) other than USS-NIZK is an extended target-collision-resistant function that is a "secret-key-free" primitive unlike "authenticating" ones used in the literature. Note that, in theory, a signature scheme can be constructed solely from NIZK by using its common-reference generator as a one-way function. However, the resulting scheme suffers from a significant performance overhead [6] and, unlike ours, does not allow us to conclude that *upgrading a NIZK to an unbounded simulation sound NIZK requires the use of a signature-like primitive*.

Our general construction shares an idea with other works, e.g. [33]: the trapdoor for zero-knowledge simulation can be used as a signing-key and the simulated proofs should work as signatures (because the simulation function can only be invoked with the trapdoor), which are publicly verifiable with the crs bound to the trapdoor. Unforgeability is argued based on the simulation soundness property. Our result is quite general in terms of the language that the underlying USS-NIZK must support.

1.3 Related Works

There exist several works for extending NIZKs to support disjunction of instances without reductions to NP-complete languages. In [14] Cramer et al., presented a very useful framework to extend any sigma-protocol to handle disjunctive relations among instances. The idea is to split a challenge into two so that one of them can be predicted in advance for simulating the 'no' side of the two instances. This idea applies to a wide range of NIZK constructions based on the Fiat-Shamir heuristic [20], splitting a crs into two shares allows similar ideas if

Ref.	Objective	Statements proved on the underlying NIZK
[50]	NIZK → OSS-NIZK	$R(x, \underline{\omega})$
[17]	NIZK → USS-NIZK	$R(x, \underline{\omega}) \vee (y{=}\mathsf{PRF}_{\underline{s}}(vk) \wedge \mathsf{Com}(\underline{s}; \underline{r}){=}\sigma) \vee (\hat{\sigma}{=}\mathsf{PRG}(\underline{s}))$
[29]	NIZK → USS-NIZK	$R(x, \underline{\omega}) \vee \underline{\sigma} = \mathsf{SIG}_{\underline{sk}}(vk)$
[7]	NIZK → SIG	$y = \mathsf{PRF}_{\underline{s}}(m) \wedge \mathsf{Com}(\underline{s}; \underline{r}) = \sigma$
[34]	USS-NIZK → LRSIG	$C = \mathsf{ENC}_{pk}(\underline{x}\|\|m; \underline{\omega}) \wedge y = \mathsf{TCR}_k(\underline{x})$
[26]	SE-SNARK → SoK	$R(x, \underline{\omega}) \wedge y = \mathsf{TCR}_k(m)$

Table 1: Upper block: General transformations from NIZK to USS-NIZK. Lower block: Constructions of various signature schemes based on non-interactive arguments. Underlined symbols are witnesses. OSS-NIZK stands for *one-time simulation-sound NIZK*. R: relation associated to the original language. σ, $\hat{\sigma}$: common reference strings. PRF: pseudo-random function. PRG: pseudo-random generator. ENC: CPA-secure encryption. LRSIG: leakage-resilient signatures. TCR: target-collision-resistant function. SIG: signature scheme. SoK: signature of knowledge.

some algebraic properties are available. Other works [21,41,46] follow an approach based on Groth-Sahai proofs, which can be used to prove disjunctive statements, e.g., [13,24]. Furthermore, Abdalla et al. [2] achieve disjunction through a smooth projective hash proof system [16].

Many works also try to upgrade NIZK systems to achieve simulation-soundness. Such upgrades usually require additional cryptographic primitives or the language associated to the NIZK be extended. In Table 1 we exemplify some of these transformations. The construction by Sahai in [50] is based on the generation of multiple common-reference strings of the original NIZK. It is a fully black-box construction that works for any NIZK systems and languages but results only in bounded simulation soundness that allow preliminary bounded number of queries. De Santis et al. built the first USS-NIZK in [17] by using a pseudo-random function (PRF) and a commitment scheme, in combination with a general NIZK that supports disjunction. The essential idea of this work is to prove that certain statement is true *or* the PRF was computed correctly with a secret key that was previously committed in the crs. Groth [24], followed by other works [3,13,29], combined a signature scheme and a one-time signature scheme with a NIZK system for satisfiability of relations over bilinear groups. Kiltz et al. combined randomized PRFs with a QA-NIZK based on the Matrix DH and the Kernel DH assumptions [35]. In summary, all these works for obtaining USS are non black-box, since they require specific properties.

Our last contribution is motivated by our impossibility results and the observation that the attempts from the literature to build signature schemes from USS-NIZKs require the language be extended or the use of additional primitives. For example, Bellare and Goldwasser [7] construct a signature scheme by combining a PRF and a public-key encryption scheme (as a commitment scheme) with a standard NIZK. Another attempt in [34] combines a *labelled* PKE scheme [16] with a USS-NIZK system to produce a signature scheme where messages are embedded into a label of the encryption. Libert et al. [37, 38] combined a SS-QA-NIZK system with a signature scheme to achieve new functionalities. Groth and Maller [26] present a general framework for constructing signature of knowledge (SoK) schemes based on succinct simulation extractable non-interactive arguments of knowledge (SE-SNARK) requiring conjunctive extension.

2 Preliminaries

2.1 Notations

For a finite set X, we write $x \leftarrow X$ to denote that x is uniformly sampled from set X. If we need to be explicit about random coins, r, used in the sampling, we write $x \leftarrow X(r)$. For $n \in \mathbb{N}$, we denote by U_n the uniform distribution over $\{0,1\}^n$. A positive function $\epsilon : \mathbb{N} \to [0,1] \subseteq \mathbb{R}$ is called *negligible* if for every polynomial $p(x) \in \mathbb{R}[X]$ there exists a constant κ_0 s.t. $\forall \kappa \geq \kappa_0$, $\epsilon(\kappa) < 1/p(x)$.

By $y \leftarrow A(x)$ we denote a process of computation where A takes x as input and outputs y. By A^{O}, we denote oracle algorithm A that interacts with oracle O. For oracle O, we use notation $y \leftarrow \mathsf{O}(x)$ also to represent a pair of input and output, (x, y), when we need to be explicit about O. Variables with brackets $[\cdot]$ match to any value. For instance, $y \leftarrow \mathsf{O}([x])$ matches to any oracle query whose output is y and we refer to the input value by x thereafter. When the matched value will not be referred afterwards, we use $*$ and write $y \leftarrow \mathsf{O}(*)$ to mean that there exists an input to O that results in y. We also use the wildcard $[* \neq \bot] \leftarrow \mathsf{O}(x)$ to denote that O outputs something other than $\bot$ for input x.

Algorithms and oracles often implement several functions identified by an input. By $\mathsf{M}(\mathsf{func}, \mathit{args})$ we mean that algorithm M works as a function specified by func taking *args* as input. Dot notation $\mathsf{M}.\mathsf{func}$ is used as well when inputs are not important in the context.

2.2 Hard Language and Language Extension

We say that $\mathcal{L}$ is a hard language accompanied by $\mathcal{C}$ if $\mathcal{L}$ and $\mathcal{C}$ are efficiently sampleable, disjoint, and hard to distinguish. Accordingly, $(\mathcal{L}, \mathcal{C})$ constitutes a promise problem [18, 52]. More formally:

Definition 2.1 (Hard Language). *Let $R_{\mathcal{L}}$ be an efficiently computable binary relation. For fixed polynomials poly_x and poly_w, let $\mathcal{L}_\kappa := \{x \in \{0,1\}^{\mathrm{poly}_x(\kappa)} \mid \exists w \in \{0,1\}^{\mathrm{poly}_w(\kappa)} : R(x,w) = 1\}$, and $\mathcal{L} := \cup_\kappa \mathcal{L}_\kappa$. Let $\mathcal{C}_\kappa \subseteq \{0,1\}^{\mathrm{poly}_x(\kappa)}$ and $\mathcal{C} := \cup_\kappa \mathcal{C}_\kappa$. Given a negligible function ϵ_{hd}, we say $\mathcal{L}$ is ϵ_{hd}-hard (with respect to $\mathcal{C}$) if for every $\kappa \in \mathbb{N}$, $\mathcal{L}_\kappa \cap \mathcal{C}_\kappa = \emptyset$ and the following properties are satisfied:*

- *For all $\kappa \in \mathbb{N}$, there exist efficiently sampleable distributions $\mathcal{D}_{\mathcal{L}_\kappa}$ and $\mathcal{D}_{\mathcal{C}_\kappa}$ producing elements from $\mathcal{L}_\kappa$ and $\mathcal{C}_\kappa$ respectively.*
- *$\mathcal{L}$ and $\mathcal{C}$ are indistinguishable, w.r.t. $\mathcal{D}_{\mathcal{L}} = \{\mathcal{D}_{\mathcal{L}_\kappa}\}_{\kappa \in \mathbb{N}}$ and $\mathcal{D}_{\mathcal{C}} = \{\mathcal{D}_{\mathcal{C}_\kappa}\}_{\kappa \in \mathbb{N}}$, i.e., for every p.p.t. algorithm A and for all sufficiently large κ, it holds*

$$|\Pr[\, x \leftarrow \mathcal{D}_{\mathcal{L}_\kappa} \;:\; 1 \leftarrow A(x)\,] - \Pr[\, x \leftarrow \mathcal{D}_{\mathcal{C}_\kappa} \;:\; 1 \leftarrow A(x)\,]| < \epsilon_{\mathsf{hd}}(\kappa) \;.$$

When it is clear from the context, we will write $x \leftarrow \mathcal{L}_\kappa$ or $x \leftarrow \mathcal{C}_\kappa$ instead of $x \leftarrow \mathcal{D}_{\mathcal{L}_\kappa}$ or $x \leftarrow \mathcal{D}_{\mathcal{C}_\kappa}$ respectively.

Definition 2.2 (Extended Language). *Given two languages $\mathcal{L}$ and $\hat{\mathcal{L}}$, and a logical binary operator $\diamond \in \{\wedge, \vee\}$, an extended language (denoted by $\mathcal{L} \diamond \hat{\mathcal{L}}$) is defined as the union $\cup_\kappa(\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa)$ where $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa := \{(x, \hat{x}) \,|\, (x \in \mathcal{L}_\kappa) \diamond (\hat{x} \in \hat{\mathcal{L}}_\kappa)\}$. The extension is said to be* non-trivial *if $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa \not\subseteq \mathcal{L}_{\kappa'}$ for any κ and κ'.*

Note that, for any non-empty finite $\mathcal{L}_\kappa$ and $\hat{\mathcal{L}}_\kappa$, we have $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa \not\subseteq \mathcal{L}_\kappa$. In this work, we only consider non-trivial language extensions. A language extension of a NIZK (with respect to operator $\diamond$) consists of, given languages $\mathcal{L}$ and $\hat{\mathcal{L}}$ and a NIZK scheme L for $\mathcal{L}$, build a NIZK scheme M for $\mathcal{L} \diamond \hat{\mathcal{L}}$.

2.3 Non-Interactive Zero-Knowledge Argument System

In this section we present syntactical and security definitions for *labelled* NIZKs. Fixing label ℓ to a default, e.g. the empty string, results in the standard definitions for (non-labelled) NIZKs.

Definition 2.3 (Labelled Non-Interactive Argument System). *A labelled non-interactive argument system for language $\mathcal{L}$ associated to relation R is a tuple of polynomial-time algorithms $(\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf})$ where:*

- *$\sigma \leftarrow \mathsf{Crs}(1^\kappa)$ takes a security parameter and generates a crs, σ.*
- *$\pi \leftarrow \mathsf{Prv}(\sigma, x, \ell, w)$ takes σ, an instance x, a label ℓ, and a witness w as input and outputs a proof π or $\perp$.*
- *$b \leftarrow \mathsf{Vrf}(\sigma, x, \ell, \pi)$ takes σ, an instance x, a label ℓ, and a proof π, and outputs either 1 or 0 representing acceptance or rejection, respectively.*

For correctness, it is required that there exists a negligible function ϵ_{co} in κ such that, for all sufficiently large κ, all $(x, w) \in \{0,1\}^{\mathrm{poly}_x(\kappa)} \times \{0,1\}^{\mathrm{poly}_w(\kappa)}$ with $R(x, w) = 1$, and all $\ell \in \{0,1\}^{\mathrm{poly}_\ell(\kappa)}$, it holds:

$$\Pr\left[\sigma \leftarrow \mathsf{Crs}(1^\kappa); \pi \leftarrow \mathsf{Prv}(\sigma, x, \ell, w) \,:\, 1 \neq \mathsf{Vrf}(\sigma, x, \ell, \pi)\right] < \epsilon_{\mathsf{co}}(\kappa) \ .$$

For soundness, it is required that there exists a negligible function ϵ in κ such that, for any p.p.t. algorithm $\mathcal{A}$ and all sufficiently large κ:

$$\Pr\left[\,\sigma \leftarrow \mathsf{Crs}(1^\kappa)\,;\, (x, \ell, \pi) \leftarrow \mathcal{A}(\sigma) \ : \ x \notin \mathcal{L}_\kappa \wedge 1 = \mathsf{Vrf}(\sigma, x, \ell, \pi)\,\right] \leq \epsilon(\kappa) \ .$$

Definition 2.4 (Adaptive Zero-Knowledge). *A labelled non-interactive argument system $(\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf})$ is* adaptive zero-knowledge *if there exists a pair of p.p.t. algorithms CrsSim and PrvSim and a negligible function ϵ_{azk} in κ such that for every p.p.t. algorithm $\mathcal{A}$ and for every sufficiently large κ,*

$$\left|\Pr\left[\sigma \leftarrow \mathsf{Crs}(1^\kappa) : 1 \leftarrow \mathcal{A}^{O_1(\cdot,\cdot,\cdot)}(\sigma)\,\right] - \Pr\left[(\sigma, \tau) \leftarrow \mathsf{CrsSim}(1^\kappa) : 1 \leftarrow \mathcal{A}^{O_0(\cdot,\cdot,\cdot)}(\sigma)\,\right]\right|$$

is lower than $\epsilon_{\mathsf{azk}}(\kappa)$. Oracles O_1, O_0, on input (x, ℓ, w) output $\perp$ if $R(x, w) = 0$, and otherwise, they return $\mathsf{Prv}(\sigma, x, \ell, w)$ and $\mathsf{PrvSim}(\sigma, x, \ell, \tau)$ respectively.

We say it is a *non-adaptive* multi-theorem zero-knowledge if the above holds when the adversary $\mathcal{A}$ is limited interact with O only before σ is generated.

Definition 2.5 (Unbounded Simulation Soundness). *A labelled non-interactive zero-knowledge argument system* $\mathit{\Pi} := (\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ *for language* $\mathcal{L}$ *is unbounded simulation sound if there exists a negligible function* ϵ_{ss} *in* κ *such that, for any p.p.t. algorithm* $\mathcal{A}$*,*

$$\mathrm{Adv}^{\mathsf{USS}}_{\mathit{\Pi},\mathcal{A}}(\kappa) := \Pr\left[\begin{array}{l} (\sigma,\tau) \leftarrow \mathsf{CrsSim}(1^{\kappa}) \\ (x,\ell,\pi) \leftarrow \mathcal{A}^{\mathsf{PrvSim}(\sigma,\cdot,\cdot,\tau)}(\sigma) \end{array} : \begin{array}{l} (x,\ell) \notin Q \wedge x \notin \mathcal{L}_\kappa \\ \wedge\ 1 = \mathsf{Vrf}(\sigma,x,\ell,\pi) \end{array}\right] < \epsilon_{\mathsf{ss}}(\kappa)$$

holds, where Q *is a list of queries sent to* PrvSim.

A non-interactive argument system that is zero-knowledge and unbounded simulation sound is called USS-NIZK.

We next present two lemmas related to the behavior of a zero-knowledge simulator. They state that the simulator must produce valid proofs for an overwhelming amount of yes-instances of the language (due to the *zero-knowledge property*) and valid proofs for an overwhelming amount of no-instances of the language (due to the *hardness of the language*).

Definition 2.6 (Yes-instance simulation correctness). *A non-interactive argument system* $\mathit{\Pi} = (\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ *for language* $\mathcal{L}$ *is yes-instance simulation correct if, for any* $x \in \mathcal{L}_\kappa$ *and* $\ell \in \{0,1\}^{poly_\ell(\kappa)}$*, the probability*

$$\epsilon_{\mathsf{yes}}(\kappa) := \Pr[(\sigma,\tau) \leftarrow \mathsf{CrsSim}(1^{\kappa})\ :\ 0 = \mathsf{Vrf}(\sigma, x, \mathsf{PrvSim}(\sigma, x, \ell, \tau))]$$

is negligible in κ*. The NIZK system* $\mathit{\Pi}$ *is perfectly yes-instance simulation correct if* $\epsilon_{\mathsf{yes}}(\kappa) = 0$.

Lemma 2.1. $\epsilon_{\mathsf{yes}}(\kappa) \leq \epsilon_{\mathsf{zk}}(\kappa) + \epsilon_{\mathsf{co}}(\kappa)$.

Definition 2.7 (No-instance simulation correctness). *A non-interactive argument system* $\mathit{\Pi} = (\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ *for* ϵ_{hd}*-hard language* $\mathcal{L}$ *accompanied by* $\mathcal{C}$ *is no-instance simulation correct if for every* $\ell \in \{0,1\}^{poly_\ell(\kappa)}$*, the probability*

$$\epsilon_{\mathsf{no}}(\kappa) := \Pr[(\sigma,\tau) \leftarrow \mathsf{CrsSim}(1^{\kappa})\ ;\ x \leftarrow \mathcal{D}_{\mathcal{C}_\kappa}\ :\ 0 = \mathsf{Vrf}(\sigma, x, \mathsf{PrvSim}(\sigma, x, \ell, \tau))]$$

is negligible in κ*.* $\mathit{\Pi}$ *is perfectly no-instance simulation correct if* $\epsilon_{\mathsf{no}}(\kappa) = 0$.

Observe that the yes-instance simulation correctness is universally quantified for all $x \in \mathcal{L}_\kappa$. However, the same is too restrictive in the case of no-instance simulation correctness, because, in general, a proof simulator may not produce valid proofs for a small set of no-instances without violating zero-knowledge.

Lemma 2.2. $\epsilon_{\mathsf{no}}(\kappa) \leq \epsilon_{\mathsf{zk}}(\kappa) + \epsilon_{\mathsf{co}}(\kappa) + \epsilon_{\mathsf{hd}}(\kappa)$.

We refer to the full version of this paper for a formal proof of the lemmas [1].

2.4 Fully Black-Box Constructions and Separation

We follow the framework of fully black-box constructions and separation in [31,49]. We say that there is a fully black-box construction of primitive A based on primitive B if, given L securely implementing B as an oracle, there exists an oracle machine M such that M^{L} securely implements A.

On the other hand, to show the absence of a fully black-box construction, we use the so-called *single oracle separation technique* [31]. That is, there is no fully black-box construction of primitive A based on B if there exists an oracle O and an oracle machine L such that L^{O} securely implements B, but any oracle machine M such that M^{O} implements A, is insecure. In Section 3, we show an oracle O such that L^{O} is a NIZK system for $\mathcal{L}$, but any construction M^{O} of a NIZK system for language $\mathcal{L} \vee \hat{\mathcal{L}}$ is insecure (no matter what $\hat{\mathcal{L}}$ is, as long as it is hard). As we investigate constructions that do not rely on particular structures or properties, we treat $\mathcal{L}$ as a black-box as well. (Therefore, it would be more precise to denote the language as $\mathcal{L}^{\mathsf{O}}$ but we abuse notation and use $\mathcal{L}$ instead.)

By $A \Rightarrow B$, we mean that there exists a fully black-box construction of B based on A. A fully black-box separation is denoted by $A \not\Rightarrow B$. If a separation holds for a restricted class of black-box constructions, we denote it by $A \not\Rightarrow_* B$. Though separations for restricted classes of black-box constructions can bring insight to a particular problem, rigorously, they are weaker than fully black-box separations, so we make it explicit.

3 Disjunctive Language Extension

We show that given a NIZK system with strong properties such as labelling and unbound simulation soundness, it is hard to disjunctively extend the language, even when compromising labelling or simulation soundness.

Theorem 3.1. *(LBL-USS-NIZK $\not\Rightarrow$ OR-NIZK) Let $\hat{\mathcal{L}}$ be a hard language. There does not exist a fully black-box construction M that converts any labelled USS-NIZK system L (for some language $\mathcal{L}$), into a NIZK scheme for $\mathcal{L} \vee \hat{\mathcal{L}}$ that is correct, adaptive zero-knowledge, and sound.*

Given the straightforward implications among NIZK, USS-NIZK, and LBL-USS-NIZK, Theorem 3.1 implies that no black-box disjunctive language extension is possible with respect to NIZK, USS-NIZK, or LBL-USS-NIZK.

Corollary 1. *It holds NIZK $\not\Rightarrow$ OR-NIZK, and USS-NIZK $\not\Rightarrow$ OR-USS-NIZK, and LBL-USS-NIZK $\not\Rightarrow$ LBL-OR-USS-NIZK.*

In the rest of this section, we prove Theorem 3.1. For that, we first describe an oracle used for constructing a hard language and a labelled USS-NIZK for it.

Definition 3.1 (Oracle O). *Oracle O_κ is equipped with two injections $H_c : \{0,1\}^\kappa \to \{0,1\}^{2\kappa}$ and $H_x : \{0,1\}^{\kappa+1} \to \{0,1\}^{2\kappa}$, and a permutation $H_p : \{0,1\}^{6\kappa} \to \{0,1\}^{6\kappa}$. Let H_c^{-1}, H_x^{-1} and H_p^{-1} be their respective inverse functions* *that output $\perp$ for inputs having no preimages. Oracle O_κ provides three language-related functionalities* SmplYes*,* SmplNo*, and* Promise*, and four NIZK-related functionalities,* Crs*,* Prv*,* PrvSim *and* Vrf *that:*

- $\mathsf{O}_\kappa(\mathsf{SmplYes}, w) \to x$: *Compute $x \leftarrow H_x(1||w)$, and output x.*
- $\mathsf{O}_\kappa(\mathsf{SmplNo}, w) \to x$: *Compute $x \leftarrow H_x(0||w)$, and output x.*
- $\mathsf{O}_\kappa(\mathsf{Promise}, x) \to 0/1$: *Output 0 if $\perp \leftarrow H_x^{-1}(x)$. Output 1, otherwise.*

- $\mathsf{O}_\kappa(\mathsf{Crs}, \tau) \to \sigma$: *Compute $\sigma \leftarrow H_c(\tau)$ and output σ.*
- $\mathsf{O}_\kappa(\mathsf{Prv}, \sigma, x, \ell, w) \to \pi/\perp$: *Output $\pi \leftarrow H_p(\sigma||x||\ell)$. ($\star$)*
- $\mathsf{O}_\kappa(\mathsf{PrvSim}, \sigma, x, \ell, \tau) \to \pi/\perp$: *Output $\pi \leftarrow H_p(\sigma||x||\ell)$. ($\star\star$)*
- $\mathsf{O}_\kappa(\mathsf{Vrf}, \sigma, x, \ell, \pi) \to 0/1$: *Output 1 if $(\sigma||x||\ell) = H_p^{-1}(\pi)$, else 0.*

($\star$) *Output $\perp$ instead if $\perp \leftarrow H_c^{-1}(\sigma)$, $x \neq H_x(1||w)$, or $\ell \notin \{0,1\}^{2\kappa}$.*

($\star\star$) *Output $\perp$ instead if $\sigma \neq H_c(\tau)$, $\perp \leftarrow H_x^{-1}(x)$, or $\ell \notin \{0,1\}^{2\kappa}$.*

We denote by O *an oracle consisting of a set of* O_κ *for all $\kappa \in \mathbb{N}$. Given an input,* O *defines κ based on the size of the second argument and checks if all other arguments follow the appropriate size. On a successful check, it forwards the input to* O_κ *and outputs the result, otherwise* O *outputs $\perp$. By $\mathcal{O}$ we denote the set of all possible oracles* O*.*

A query to O is *successful* if the answer is not $\perp$ (or not 0 in the case of $\mathsf{O}.\mathsf{Vrf}$). We say that a common reference string σ is *valid* (with respect to O) if there exists τ that satisfies $\sigma = H_c(\tau)$. Given σ (without τ), it is easy to assure its validity by checking that $\mathsf{O}(\mathsf{Prv}, \sigma, x, \ell, w)$ is different from $\perp$, where x can be any yes-instance and w its corresponding witness.

The oracle O can be seen as a set consisting of entries of the form (cmd, *args*, *output*) where command cmd is one of $\{\mathsf{SmplYes}, \mathsf{SmplNo}, \mathsf{Promise}, \mathsf{Crs}, \mathsf{Prv}, \mathsf{PrvSim}, \mathsf{Vrf}\}$, *args* denotes inputs for each command, and *output* is the answer. Inputs and outputs may include wildcards such as $*$. Then, a set S of entries of this form is called a *partial oracle* as it can be seen as an oracle that accepts only limited inputs. A partial oracle S is called *consistent* if there exists another set S' such that $S \cup S'$ forms a complete oracle in $\mathcal{O}$. Otherwise S is called *inconsistent.* A *hybrid oracle*, denoted as $S := S_1|S_2|\cdots$, is an oracle that combines partial oracles $S_1, S_2, \ldots$ in such a way that, given a query of the form (cmd, *args*), it first searches S_1 for matching entry (cmd, *args*, [*output*]) and returns *output* if it exists. If no such entry is found in S_1, it searches S_2 and so forth. Note that a hybrid oracle may not be consistent.

Let L be an oracle machine that, given O as an oracle, forwards its input to O and outputs whatever O outputs. L^O implements a hard promise problem and a NIZK argument system for it. (Some trivial syntactical adjustments are needed to fit to the definition of NIZK in 2.3 and 2.4.) The following lemma holds for L^O.

Lemma 3.1. *With probability* 1 *over the choice*[3] *of* $\mathsf{O} \in \mathcal{O}$*,* L^{O} *implements a hard promise problem* $(\mathcal{L}, \mathcal{C})$ *for* $\mathcal{L}_\kappa := \{x \mid \exists w \in \{0,1\}^\kappa \text{ s.t. } x = H_x(1||w)\}$ *and* $\mathcal{C}_\kappa := \{x \mid \exists w \in \{0,1\}^\kappa \text{ s.t. } x = H_x(0||w)\}$*. It also implements a non-interactive zero-knowledge argument system for* $\mathcal{L}$ *that is perfectly correct, perfectly yes-instance and no-instance simulation correct. Furthermore, it is adaptive zero-knowledge and unbound simulation-sound against uniform adversaries given oracle access to* O *a polynomial number of times.*

By design of the oracle, adversaries that interact with it can only win if some *bad events* occur. It is not hard to see that these events occur with negligible probability. We refer to the full version of this paper for a formal proof [1].

We make some remarks about our design choices for O and the properties of L. It was shown in [11, 55] that a simpler witness-indistinguishable oracle suffices to construct a simulation sound NIZK. It is however essential for their construction that the oracle supports an NP-complete language (or a specific disjunctive language). The NIZK implemented by the above L is deterministic but one can make it probabilistic so that (simulated) proofs have κ-bit entropy simply by attaching κ-bit randomness to the proof. The simulation soundness of L^{O} will not be directly used in our proof of impossibility. What is important here is to see that O suffices to construct a USS-NIZK for $\mathcal{L}$.

Intuition for the impossibility. If the construction M is such that, $\mathsf{M.Crs}$ generates some σ_j by calling $\mathsf{O.Crs}$ and encoding them into $\widetilde{\sigma}$ (we name such crs's *legitimate*), we claim that the prover algorithm $\mathsf{M.Prv}$ cannot use them. Otherwise, the adaptive zero-knowledgeness will be compromised. That is, every crs that used when proving a given statement, should be generated within the prover algorithm (except for some eccentric cases that we explain later). A crucial observation is that, to prove disjunction for an instance $(x, \hat{x})$, it may be the case that $(x, \hat{x}) \in \mathcal{C}_\kappa \times \hat{\mathcal{L}}_\kappa$ and the no-instance x cannot be proven with a legitimate crs whose trapdoor is unknown to the prover. On the other hand, using a legitimate crs only for yes-instances contradicts the zero-knowledgeness of the underlying scheme (or the hardness of $\mathcal{L}$), because the zero-knowledge simulator does not know whether the given x is a yes or a no-instance.

Let us elaborate on this point. In the next, let $q(\kappa)$ be a (non-constant) polynomial in the security parameter that upper-bounds the number of queries to O that M^{O} performs in each invocation. Let $c > 1$ be a constant. Consider the adaptive zero-knowledge game where an adversary submits disjunctive instances $(x_j, \hat{x}_j)$ for $j = 1, \ldots, q^c$ (for certain integer c) to the challenger that produces proofs either by $\mathsf{M.Prv}$ (in the *real* world) or $\mathsf{M.PrvSim}$ (in the *simulated* world). The adversary verifies the proofs by $\mathsf{M.Vrf}$ which may make verification queries $\mathsf{O.Vrf}$ on the left-hand instance, x_j. Consider the case where instances $(x_j, \hat{x}_j)$

[3] Technically, for every machine $\mathcal{A}$, there exist a set of measure 0 of oracles for which $\mathcal{A}$ has a significant advantage either against the hardness of the language or the NIZK system. As it is standard after the application of the Borel-Cantelli Lemma, given that there exist countably many machines, for a measure 1 sets of oracles in $\mathcal{O}$ we can say that for all p.p.t. machines $\mathcal{A}$ our result holds.

are taken from $\mathcal{C}_\kappa \times \hat{\mathcal{L}}_\kappa$ as pairs of $(\mathsf{no}, \mathsf{yes})$ instances, and the challenger responds with $\mathsf{M.Prv}$ in the real-world. We then define $(\mathsf{real}, \mathsf{no}, \mathsf{yes})$ as the distribution of all crs's (over the choice of instances and coins of M) that are given as input to $\mathsf{O.Vrf}$ to verify the left-hand instances x_j for $j = 1, \dots, q^c$. We define $(\mathsf{sim}, \mathsf{no}, \mathsf{yes})$ similarly for the case the challenger respond with $\mathsf{M.PrvSim}$ in the $\mathsf{simulated}$-world. Switching yes and no according to whether x_j (and $\hat{x}_j$, resp.) are chosen from $\mathcal{L}_\kappa$ or $\mathcal{C}_\kappa$ ($\hat{\mathcal{L}}_\kappa$ or $\hat{\mathcal{C}}_\kappa$, resp.), we have

$$(\mathsf{real}, \mathsf{no}, \mathsf{yes}) \overset{(\S)}{\approx} (\mathsf{sim}, \mathsf{no}, \mathsf{yes}) \overset{(\dagger)}{\approx} (\mathsf{sim}, \mathsf{yes}, \mathsf{yes}) \overset{(\ddagger)}{\approx} (\mathsf{sim}, \mathsf{yes}, \mathsf{no}) \overset{(\S)}{\approx} (\mathsf{real}, \mathsf{yes}, \mathsf{no})$$

where $\approx$ denotes indistinguishability of distributions and (§) is given by the zero-knowledge property and (†) and (‡) are given by the hardness of $\mathcal{L}$ and $\hat{\mathcal{L}}$ respectively. (Observe that $\mathsf{M.PrvSim}$ could be used to distinguish between $\mathcal{L}$ and $\mathcal{C}$ if the indistinguishability denoted by (†) did not hold). Also, observe that $(\mathsf{real}, \mathsf{no}, \mathsf{yes})$ does not contain a legitimate crs, because the real prover algorithm cannot prove on a no-instance with a legitimate crs whose trapdoor is not given. Because legitimate crs's can be identified (we refer to the *learning-phase* defined below for more details), and given the above indistinguishability relations, the same is true for $(\mathsf{real}, \mathsf{yes}, \mathsf{no})$: it does not contain legitimate crs's. Therefore, even if a witness for x_j is given, the prover must not use legitimate crs's to prove x_j.

We then argue that a NIZK system that does not use the legitimate crs for proving a given statement cannot be sound. We construct an adversary that runs the prover algorithm, $\mathsf{M.Prv}$, on $(x^*, \hat{x}^*) \in \mathcal{C}_\kappa \times \hat{\mathcal{C}}_\kappa$ and performs an *instance swapping* to fool it as if x^* was taken from $\mathcal{L}_\kappa$. This is done by giving $\mathsf{M.Prv}$ a random fake witness, w^*, for x^* and simulating O on queries involving x^*. Concretely, if $\mathsf{M.Prv}$ makes $\mathsf{O.SmplYes}$ queries on the fake witness w^* (to check its correctness), we simulate the answer by returning x^*. If $\mathsf{O.Prv}$ queries are made on x^* under a crs, σ_j, we replace the query with $\mathsf{O.PrvSim}$ using a trapdoor τ_j for σ_j. This is possible because (as we have argued above) all crs's used within $\mathsf{M.Prv}$ must have been internally generated, so their trapdoors are known. There could be a case where a legitimate crs is used to prove x^*. Recall that $(\mathsf{real}, \mathsf{yes}, \mathsf{no})$ does not include legitimate crs's, i.e., proofs with a legitimate crs will not be verified by $\mathsf{M.Vrf}$. Yet, $\mathsf{M.Prv}$ may create and verify proofs with a legitimate crs for internal use only. Therefore, the adversary must fool $\mathsf{M.Prv}$ by simulating such proofs with random strings and answering accordingly to the respective verification queries. Once $\mathsf{M.Prv}$ is done, the resulting proof $\widetilde{\pi}$ should pass the final verification since all proofs π_j embedded in $\widetilde{\pi}$ are genuine and independent of the fake witness.

Nevertheless, the above sketch ignores the possibility of a *trivial* legitimate crs whose trapdoor is also embedded to $\widetilde{\sigma}$ and available in public. Algorithm $\mathsf{M.Prv}$ may use a trivial trapdoor for no-instances and a relevant witness for yes-instances. But the witness we give to the algorithm is a fake one that does not work properly. To handle such a case, the adversary must find the trivial trapdoors in advance and use them for proofs. Although we do not know how the trivial trapdoors are encoded into $\widetilde{\sigma}$, they can be extracted by running $\mathsf{M.Prv}$ on a number of instances and observing the trapdoors used therein. Since there can be a bounded number of trivial legitimate crs's embedded in $\widetilde{\sigma}$, sufficiently repeating the proofs on random instances exhausts them with high probability.

Breaking Soundness. In the following proof, we construct adversary $\mathcal{A}$ attacking soundness of M (against a challenger $\mathcal{B}$) and use the above observation about the legitimate crs to lower bound the success probability of $\mathcal{A}$.

[Soundness Game for OR-NIZK M**]**
At the beginning, oracle O is chosen from $\mathcal{O}$. Then challenger $\mathcal{B}$ and adversary $\mathcal{A}$ engage in the following procedures.

Step 1: Setup Phase.

The challenger generates a common reference string by $\widetilde{\sigma} \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Crs}, 1^\kappa)$.
Let Q_{leg} be the list of legitimate crs's and their corresponding trapdoors (σ_j, τ_j) that have been generated in this phase, as $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$.

Step 2: Self-Learning Phase.

Given $\widetilde{\sigma}$, for every $i = 1, \ldots, q^c$, adversary $\mathcal{A}$ uniformly samples instance $(x_i, \hat{x}_i)$ and the corresponding witness $(w_i, \perp)$ from $\mathcal{L}_\kappa \times \hat{\mathcal{C}}_\kappa$ and it computes $\widetilde{\pi}_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Prv}, \widetilde{\sigma}, (x_i, \hat{x}_i), (w_i, \perp))$ and $b_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x_i, \hat{x}_i), \widetilde{\pi}_i)$.

Let Q_{triv} be the list of trivial pairs of crs and trapdoor, (σ_j, τ_j), that appeared in a computation like $[* \neq \perp] \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, *, *, \tau_j)$ or $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$ during some execution of M in this phase.

Step 3: Forgery Phase.

Sample $(x^*, \hat{x}^*) \in \mathcal{C}_\kappa \times \hat{\mathcal{C}}_\kappa$ as $w^* \leftarrow \{0,1\}^\kappa$, $x^* \leftarrow \mathsf{O}(\mathsf{SmplNo}, w^*)$ and $\hat{x}^* \leftarrow \hat{\mathcal{C}}_\kappa$
Let $\bar{x} := \mathsf{O}(\mathsf{SmplYes}, w^*)$ and apply instance swapping:

Let O' be the partial oracle given by the entries $(\mathsf{SmplYes}, w^*, x^*)$, $(\mathsf{SmplNo}, w^*, \bar{x})$, and $(\mathsf{Prv}, *, \bar{x}, *, w^*, \perp)$. Run $\mathsf{M}^{\mathsf{O}''}(\mathsf{Prv}, \widetilde{\sigma}, (x^*, \hat{x}^*), (w^*, \perp))$ where O'' is an algorithm that simulates an oracle in $\mathcal{O}$ as follows.

[Algorithm O''] Initialize Q_{intl} as an empty list.

- If a given query is defined in O', return the output accordingly.
- Given $(\mathsf{Crs}, [\tau_j])$, return $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$ and record (σ_j, τ_j) to Q_{intl}.
- Given $(\mathsf{Prv}, [\sigma_j], x^*, [\ell_j], w^*)$ with valid σ_j:
  (a) if $(\sigma_j, [\tau_j]) \in Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$:
  return $\pi_j \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, x^*, \ell_j, \tau_j)$ and add $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, \pi_j)$ to O'.
  (b) else:
  return $\pi_j \leftarrow \{0,1\}^{6\kappa}$; add $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, \pi_j)$, $(\mathsf{Vrf}, \sigma_j, x^*, \ell_j, \pi_j, 1)$ to O'.
- For every other query, forward it to O and return the output.

When M outputs a proof $\widetilde{\pi}^*$, $\mathcal{A}$ outputs $(x^*, \hat{x}^*)$ and $\widetilde{\pi}^*$ as a forgery.

Step 4: Final Verification Phase.

Challenger $\mathcal{B}$ outputs 1 (interpreted as *the adversary wins*) if $x^* \notin \mathcal{L}_\kappa$, $\hat{x}^* \notin \hat{\mathcal{L}}_\kappa$ and $1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$; otherwise, it outputs 0 (*the adversary loses*).

Lemma 3.2. *The above adversary $\mathcal{A}$ wins the simulation soundness game of M^{O} with non-negligible probability if M is non-adaptive multi-theorem zero-knowledge and correct.*

We will use the following lemma in the proof of Lemma 3.2. It states that if an event happens for n successive independent attempts, the probability that it suddenly does not happen is upper-bounded by an inverse polynomial of n. We use this lemma to claim that, during the challenge phase, the adversary observes all trivial σ_j embedded in $\widetilde{\sigma}$ generated by the challenger.

Lemma 3.3 ([54, Fact 4.6.1]). *Let $X_1, \cdots X_{n+1}$ be independent Bernoulli random variables, where $Pr[X_i = 1] = p$ and $Pr[X_i = 0] = 1 - p$ for $i = 1, \cdots, n+1$, and some $p \in [0, 1]$. Let E be the event that the first n variables are sampled at 1, and X_{n+1} is sampled at 0. Then, $Pr[E] \leq \frac{1}{e \cdot n}$, where $e \simeq 2.71$ is the base of the natural logarithm.*

Proof. (of Lemma 3.2) We analyze the probability that the forged proof passes the verification in the above game. Let $\rho_{\mathsf{zk}}(\kappa)$ and $\rho_{\mathsf{co}}(\kappa)$ denote upper-bounds for non-adaptive multi-theorem zero-knowledge and correctness for M as defined in Definitions 2.3 and 2.4, respectively. We consider these parameters as universal for all O. Let P be the probability that challenger $\mathcal{B}$ outputs 1 in the final verification phase, which is taken over the choice of O and coin flips by $\mathcal{B}$ and $\mathcal{A}$.

Our goal is to show that P is not negligible. Towards that goal, we consider a sequence of games (where each game is identical to the precedent, except for the mentioned details) that introduces arbitrarily small (though not necessarily negligible) differences in the considered probability and eventually reach the situation where $\mathcal{B}$ outputs 1 trivially. We denote the probability for the same event in Game i by P_i. In the first games (Game 0 to Game 6) we exclude events that happen only with negligible probability, simplifying the next transitions. Under the condition that these events do not happen, O'' simulates an oracle in $\mathcal{O}$ that successfully produces a correct proof on a yes instance (of the disjunctive relation). In the succeeding games (Game 7 to 9), we replace oracle O by O'', relying on the zero-knowledge property and the hardness of the languages. In the last Game 9, the adversary does nothing but creating a proof for a yes instance, which must be accepted with high probability.

Game 0: The above soundness game. So $P_0 = P$.

Game 1: For every successful query to $\mathsf{O.PrvSim}$ or $\mathsf{O.Prv}$ with respect to some σ_j, query $\mathsf{O.Crs}$ that generates σ_j must have been made in advance within the same execution of M or in the setup phase. Similarly, for every successful query to $\mathsf{O.Vrf}$ for verifying a proof, a query to $\mathsf{O.PrvSim}$ or $\mathsf{O.Prv}$ that outputs the queried proof must have been made in advance. If any of these are not the case, the game halts.

Game 2: Modify the final verification to skip the check $x^* \notin \mathcal{L}_\kappa$ and $\hat{x}^* \notin \hat{\mathcal{L}}_\kappa$.

Game 3: Halt the game if $\mathcal{A}$ observes $b_i = 0$ in the self-learning phase.

Game 4: The game halts if there has been a query on w^* or x^* or $\bar{x}$ made by M invoked in the setup and self-learning phases.

Game 5: The game halts if any of randomly assigned π_j at step (b) of O'' appear as a result of other $\mathsf{O}.\mathsf{Prv}$ or $\mathsf{O}.\mathsf{PrvSim}$ queries by the end of the forgery phase.

Game 6: The game halts if, O'' receives a query $(\mathsf{PrvSim}, [\sigma_j], x^*, [\ell_j], [\tau_j])$ that there exists $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, [\pi_j])$ in O', and $\pi_j \neq \pi'_j \neq \perp$ holds for $\pi'_j \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, x^*, \ell_j, \tau_j)$. The modification is to exclude a case where a trapdoor τ_j for some σ_j suddenly appears for the first time in the forgery phase while σ_j itself has appeared so far.

Claim 3.2. $|P_0 - P_6| < \varepsilon(\kappa) + q^c \rho_{\mathsf{co}}(\kappa) + 2/(eq^{c-1})$, for a negligible function $\varepsilon(\kappa)$.
Proof. We include a formal proof in the full version of this paper [1].

Game 7: Replace O in the setup and self-learning phase with algorithm O'' with partial oracle O' defined at the end of the forgery phase after Game 4.

Since queries defined in O' involve x^* or $\bar{x}$, they do not appear in the setup and forgery phase. Any queries to O'' not defined in O' are answered by O. Thus, this modification does not introduce any relevant change and we have $P_7 = P_6$.
Game 8: In this game, we use O'' instead of O also in the final verification phase.

Claim 3.3. $|P_8 - P_7| < 7/eq^{c-1} + (3q^2 + 2q^{c+2})/2^{6\kappa} + 2q/2^{\kappa} + \hat{\epsilon}_{\mathsf{ind}}(\kappa) + 3\rho_{\mathsf{zk}}(\kappa)$.
Proof. See the end of this section.

Game 9: We then modify O'' so that it no longer uses O. Instead, it uses a random partial oracle R such that, $\mathsf{O}'' = \mathsf{O}'||\mathsf{R}$ is an oracle in $\mathcal{O}_\kappa$.

Since all queries to O'' from $\mathsf{M}.\mathsf{Vrf}$ in Game 8 that answered by O are consistent with the partial oracle O', the replacement by a consistent R does not change the distribution of the view of $\mathsf{M}.\mathsf{Vrf}$. Therefore, $P_9 = P_8$.

Now O'' is an oracle in $\mathcal{O}_\kappa$ and in Game 8, the adversary is creating a proof $\widetilde{\pi}^*$ on a (yes,no)-instance $(x^*, \hat{x}^*)$ with a correct witness with respect to O''. Therefore, the created proof will be accepted unless except for the correctness error. We thus have $P_9 > 1 - \rho_{\mathsf{co}}(\kappa)$.

By summing up the above differences of probabilities, we have

$$P > 1 - \left(9/(eq^{c-1}) + \hat{\epsilon}_{\mathsf{ind}}(\kappa) + 3\rho_{\mathsf{zk}}(\kappa) + (q^c + 1)\rho_{\mathsf{co}}(\kappa) + \epsilon(\kappa)\right) \ .$$

Accordingly, if M is correct, zero-knowledge, language $\hat{\mathcal{L}}$ is hard and constant c is set so that the second term of the right-hand-side of the above inequality is small enough, $\mathcal{A}$ is successful in breaking the soundness of M with non-negligible probability in κ. □

Proof. (of Claim 3.3) The view in the verification phase changes only if $\mathsf{M}.\mathsf{Vrf}$ makes one of the following queries whose output differs in O and O''.

- A query $(\mathsf{PrvSim}, [\sigma_j], x^*, [\ell_j], [\tau_j])$ such that for $\pi_j \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, x^*, \ell_j, \tau_j)$, there exists $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, [\pi'_j])$ in O', with $\pi_j \neq \pi'_j$.
- A query that is not in O' but results in a π_j that already appears in O'.
- A query included in O'.

We can bound (by a negligible function) the probability that the first two types of queries from above occur, by following similar arguments as for the transitions to Game 6 and Game 5 respectively. However, the third case requires careful analysis. In the following, we briefly present our idea for bounding the probability. First observe that O' includes two types of queries: those involving witness w^* and those where randomly assigned proofs are verified. Informally, the first type of query will occur with low probability because, otherwise, $\mathsf{M.Vrf}$ would have a valid witness for a given instance, what contradicts zero-knowledge or the hardness of the language. The second type of query must also occur with low probability because, as discussed at the beginning of this section: a legitimate non-trivial σ_j cannot be used to prove a given instance, or otherwise zero-knowledgeness will be lost.

More concretely, to bound the probability of the first type of query happening, the same argument as that for Game 6 can be applied. We consider queries done by $\mathsf{M.Vrf}$ instead of $\mathsf{M.Prv}$ in the self-learning phase and the verification phase but the analysis remains the same. This gives us an upper-bound of $2q/(eq^c)$. The second type of query can be can be bounded as in the transition to Game 5, except for additional q queries made during the verification itself. We can establish a bound of $q(3q + 2q^{c+1})/2^{6\kappa}$. Finally, for the third type of query, observe that they can be splitted into: *i)* queries including w^* and *ii)* queries for verifying a randomly assigned proof. Queries of type *i)*, are all of the form $(\mathsf{SmplYes}, w^*)$, (SmplNo, w^*), $(\mathsf{Prv}, *, \bar{x}, *, w^*)$, and $(\mathsf{Prv}, *, x^*, *, w^*)$. Queries of type *ii)* are of the form $(\mathsf{Vrf}, *, x^*, *, [\pi_j])$. Let AskW (respectively VerPi) denote the event that queries of type *i)* (respectively *ii)*) occur.

We first bound the probability that AskW happens. Let Game 8.0 be Game 8. Let $\mathsf{AskW}_{8.i}$ denote the event that AskW happens in Game $8.i$ (defined below). It is important to observe that, at this point, the view produced by O'' with O' is consistent, i.e., there exists a partial oracle that produces the same view, and in the forgery phase a correct proof on a (yes,no)-instance is being created with a correct witness with respect to the partial oracle.

Game 8.1: Replace $\mathsf{M.Crs}$ in the setup phase and $\mathsf{M.Prv}$ in the forgery phase respectively by $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$. Note that the trapdoor output by $\mathsf{M.CrsSim}$ is given to $\mathsf{M.PrvSim}$.

We can show that $|\Pr[\mathsf{AskW}_{8.0}] - \Pr[\mathsf{AskW}_{8.1}]| < \rho_{\mathsf{zk}}(\kappa)$ by constructing a zero-knowledge distinguisher from adversary $\mathcal{A}$. We then claim that $\Pr[\mathsf{AskW}_{8.1}] \leq q/2^\kappa$. This is justified by the fact that $\mathsf{M.PrvSim}$ no longer takes w^* as input and hence the view of $\mathsf{M.Vrf}$ in the final verification is independent of w^*. Therefore, the event happens only by chance among q queries. We have that $\Pr[\mathsf{AskW}] = \Pr[\mathsf{AskW}_{8.0}] < \rho_{\mathsf{zk}}(\kappa) + q/2^\kappa$.

We now bound the probability of event VerPi. Suppose that we run an execution of $\mathsf{M}^{\mathsf{O}''}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\pi})$ on certain input and suppose that it makes a verification query of the form $(\mathsf{Vrf}, \sigma_j, x^*, \ell_j, \pi_j)$ to O''. There are two options, either this query causes event VerPi or not, depending on whether π_j is a randomly assigned proof. However, we cannot decide which one is the case, event VerPi is not observable. This will be an obstacle if we try to construct an adversary against zero-knowledgeness as we did in previous cases. Alternatively we consider an event, VerCrs, that is observable and happens almost whenever VerPi happens. Suppose that a query $(\mathsf{Vrf}, \sigma_j, x^*, \ell_j, \pi_j)$ happens in the final verification phase. We would like to see if proof π_j was randomly assigned in step (b) of O'' or not. Observe that it is the case only if $(\sigma_j, [\tau_j]) \notin Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$. Furthermore, if σ_j is not an internally generated crs, it must have been generated in the setup phase. Also, as σ_j appears in the final verification, it should have appeared in a verification during the self-learning phase as well. We define VerCrs as the event that in the final verification, $\mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$, a query of the following form is done: $(\mathsf{Vrf}, [\sigma_j], x^*, [\ell_j], [\pi_j])$, satisfying $\ell_j \in \{0,1\}^{2\kappa}$, $(\sigma_j, [\tau_j]) \notin Q_{\mathsf{triv}}$, and $\sigma_j \in Q_{\mathsf{nt}}$ where Q_{nt} is a list of all σ_j queried in the self-learning phase but not included in Q_{triv}. This way, event VerCrs is observable based on the view in the self-learning phase. Yet, VerCrs can differ from VerPi when a σ_j generated in the setup phase appears for the first time in the final verification. However, applying Lemma 3.3, we can upper-bound the probability for such event by q/eq^c. We thus have

$$\Pr[\mathsf{VerPi}] \leq \Pr[\mathsf{VerCrs}] + q/eq^c \ .$$

Now, let Game 8.0′ be Game 8 and let $\mathsf{VerCrs}_{8.i'}$ denote the event that VerCrs happens in Game $8.i'$, where:

Game 8.1′: Replace $\mathsf{M.Crs}$ and $\mathsf{M.Prv}$ with $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$, respectively.

We claim that $|\Pr[\mathsf{VerCrs}_{8.0'}] - \Pr[\mathsf{VerCrs}_{8.1'}]| < \rho_{\mathsf{zk}}(\kappa) + 2q/eq^c$. To show this, we construct a zero-knowledge adversary that, given $\widetilde{\sigma}$, first executes the self-learning phase, and sends $(x^*, \hat{x}^*)$ and $(w^*, \perp)$ to the challenger. On receiving $\widetilde{\pi}^*$, the adversary runs $\mathsf{M}^{\mathsf{O}''}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$ and outputs 1 if event VerCrs happens. It outputs 0, otherwise. If the challenger is working with $\mathsf{M.Crs}$ and $\mathsf{M.Prv}$, the view in the final verification (up to the point when event VerCrs happens) distributes as in Game 8.0′. On the other hand, if the challenger is working with $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$, the view in the final verification distributes as in Game 8.1′.

In the above argument, the adversary cannot perfectly capture event VerCrs since the lists Q_{triv} and Q_{nt} that the adversary obtains from its own self-learning and uses to capture event VerCrs can be different from the ones defined for O''. This issue can be handled as follows. First, regarding σ_j in Q_{triv}, it suffices to consider those included also in Q_{leg}. This is justified by observing that the condition $(\sigma_j, [\tau_j]) \notin Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$ is equivalent to $(\sigma_j, [\tau_j]) \notin (Q_{\mathsf{triv}} \cap Q_{\mathsf{leg}}) \cup Q_{\mathsf{intl}}$, because every $\sigma_j \in Q_{\mathsf{triv}}$ that appeared during the final verification and is not present in Q_{leg} must be in Q_{intl}. Let Q'_{triv} be the lists the adversary obtained. If $Q'_{\mathsf{triv}} \cup Q_{\mathsf{leg}}$ and $Q_{\mathsf{triv}} \cup Q_{\mathsf{leg}}$ differ, there exists $(\sigma_j, \tau_j) \in Q_{\mathsf{leg}}$ that does not appear a self-learning but does in the other self-learning. We can apply Lemma 3.3 to upper-bound the probability of having different Q_{triv} and Q'_{triv} by q/eq^c (the same argument applies to Q_{nt}). This results in adding $2q/eq^c$ to the bound, as claimed. We recall that, up to this point, $(x^*, \hat{x}^*) \in \mathcal{L}_\kappa \times \hat{\mathcal{C}}_\kappa$ with respect to oracle O'.

Game 8.2′: Sample $\hat{x}^*$ from $\hat{\mathcal{L}}_\kappa$, that is, $(x^*, \hat{x}^*)$ is chosen from (yes,yes)-instances.

Any change in event VerCrs can be reduced to distinguishing $\hat{\mathcal{L}}$ and $\hat{\mathcal{C}}$. We have $|\Pr[\mathsf{VerCrs}_{8.1'}] - \Pr[\mathsf{VerCrs}_{8.2'}]| < \hat{\epsilon}_{\mathsf{ind}}(\kappa)$ where $\hat{\epsilon}_{\mathsf{ind}}(\kappa)$ is the advantage of distinguishing $\hat{\mathcal{L}}$ and $\hat{\mathcal{C}}$.

Game 8.3′: Sample x^* from $\mathcal{C}_\kappa$. That is, $(x^*, \hat{x}^*)$ is chosen from (no,yes)-instances.

Due to the indistinguishability of $\mathcal{L}$ and $\mathcal{C}$, $|\Pr[\mathsf{VerCrs}_{8.2'}] - \Pr[\mathsf{VerCrs}_{8.3'}]| < q/2^\kappa$. (Note that language $\mathcal{L}$ is implemented by oracle O.)

Game 8.4′: Replace M.CrsSim and M.PrvSim by M.Crs and M.Prv, respectively. Note that to use M.Prv, one needs a witness, which in this case is known. It is actually $(\perp, \hat{w})$, where $\hat{w}$ is a witness for $\hat{x}$ (received as input).
We have $|\Pr[\mathsf{VerCrs}_{8.3'}] - \Pr[\mathsf{VerCrs}_{8.4'}]| < \rho_{\mathsf{zk}}(\kappa) + 2q/eq^c$.

Since w^* is no longer given, a valid proof π_j on x^* with a legitimate non-trivial σ_j that triggers event $\mathsf{VerCrs}_{8.4'}$ can be created only by chance by guessing a relevant trapdoor or the witness, cases that have been already excluded in previous games. Therefore, we conclude that $\Pr[\mathsf{VerCrs}_{8.4'}] = 0$.

By summing up the above probabilities, we have

$$\Pr[\mathsf{VerPi}] < \Pr[\mathsf{VerCrs}] + q/eq^c < \hat{\epsilon}_{\mathsf{ind}}(\kappa) + q/2^\kappa + 2\rho_{\mathsf{zk}}(\kappa) + 5q/eq^c.$$

Finally, we have

$$\begin{aligned}|P_8 - P_7| &< 2q/eq^c + q(3q + 2q^{c+1})/2^{6\kappa} + \Pr[\mathsf{AskW}] + \Pr[\mathsf{VerPi}] \\ &< 7/eq^{c-1} + (3q^2 + 2q^{c+2})/2^{6\kappa} + 2q/2^\kappa + \hat{\epsilon}_{\mathsf{ind}}(\kappa) + 3\rho_{\mathsf{zk}}(\kappa).\end{aligned} \qquad \square$$

4 Conjunctive Language Extension

In this section, we consider non-labelled NIZKs. For that purpose, we drop the labels from the definition of O in the previous section. The internal random function H_p is adjusted to $H_p : \{0,1\}^{4\kappa} \to \{0,1\}^{4\kappa}$.

We consider a class $\mathcal{M}$ of constructions where every $\mathsf{M} \in \mathcal{M}$ satisfies the constraint that, roughly, all internally generated proofs π_j must be verified in the process of verifying the resulting proof. We call such M a construction in the *full verification model.* In the following, we use symbol $\not\Rightarrow_*$ to denote separations that hold in the full verification model.

Definition 4.1. *(Class of constructions with full verification.)* $\mathcal{M} := \{\mathsf{M}\}$ *is a class of black-box constructions of NIZK with respect to* $\mathcal{O}$ *such that, for every algorithm* $\mathsf{M} \in \mathcal{M}$*, the following condition is met: For all sufficiently large* $\kappa > 0$*, for every* $\mathsf{O} \in \mathcal{O}_\kappa$*,* $\widetilde{\sigma}$*,* $\widetilde{x}$*,* $\widetilde{w}$*, and query/answer pair* $[\pi_j \neq \perp] \leftarrow \mathsf{O}(\mathsf{Prv}, [\sigma_j], [x_j], [w_j])$ *observed during the execution of* $[\widetilde{\pi} \neq \perp] \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Prv}, \widetilde{\sigma}, \widetilde{x}, \widetilde{w})$*, there exists a query* $\mathsf{O}(\mathsf{Vrf}, \sigma_j, x_j, \pi_j)$ *during the execution of* $\mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\pi})$*.*

The condition captures the idea of properly using O as a proof system because whatever was proven internally by a prover is then verified by a verifier. Requiring "every" internal proof to appear also at verification is in fact needed for technical reasons. We construct an adversary that simulates proofs π_j by looking for query-answer pairs of O obtained during the challenge phase. However, such a view is only with respect to $\mathsf{M.Vrf}$ executed by the adversary itself and those with respect to $\mathsf{M.PrvSim}$ are not available, because they are executed by the challenger. So if only a subset of the internal proofs are verified in $\mathsf{M.Vrf}$, the adversary cannot simulate the distribution of the internal proofs needed to run $\mathsf{M.PrvSim}$. We do not know how to prove the separation if this condition is relaxed to, for instance, "at least one". It carries a resemblance to the constraint used in [23, footnote 9] to show a black-box separation of semantically secure encryption from chosen ciphertext secure ones. Their result applies to a class of constructions where, for every decryption query, there must exist a corresponding encryption query, or no encryption query can be made during decryption (a.k.a. the shielding model).

The following theorem can be proven by following a similar approach as the one used in the proof of Theorem 3.1. We refer to the full version of this paper [1] for a formal proof.

Theorem 4.1. *(USS-NIZK $\not\Rightarrow_*$ AND-USS-NIZK in the full verification model.) Given any two hard languages $\mathcal{L}$ and $\hat{\mathcal{L}}$ and any USS-NIZK system L for $\mathcal{L}$, there exists no fully black-box construction of USS-NIZK scheme M in class $\mathcal{M}$ for $\mathcal{L} \wedge \hat{\mathcal{L}}$ that is non-adaptive multi-theorem zero-knowledge and unbounded simulation sound.*

4.1 Constructing AND-USS-NIZK from labelled USS-NIZK

Contrary to the impossibility in the previous section, conjunctive language extension is possible for USS-NIZKs if they support labels. Exploiting the integrity of labels, an easy solution could be the following: to prove an instance (x_1, x_2) under a label ℓ we can define a label for the USS-NIZK scheme $\ell' := x_1||x_2||\ell$ and run the prover algorithm in both pairs (x_1, ℓ') and (x_2, ℓ'). Such a simple transformation works, as long as the underlying USS-NIZK can handle the longer labels that we defined. We provide a transformation that is valid independently of the label space of the underlying NIZK as long as it supports *poly-length* labels.

For a bitstring s, let $\mathsf{f}(s)$ be a Merkle encoding of s [39] as defined by $\mathsf{f}(s) := s||\mathsf{tag}(s)$, where $\mathsf{tag}(s)$ is a bitstring representing the bit length of s minus its hamming weight. The length of $\mathsf{f}(s)$ is exactly $\mathsf{len}(s) + \lceil \log_2(\mathsf{len}(s)) \rceil$. Now, let $I(s) := \{i \mid \mathsf{f}(s)_i = 1\}$ where $\mathsf{f}(s)_i$ is the i-th bit of $\mathsf{f}(\mathsf{s})$. It then holds that, $I(s)$ is not empty for any s, and for different s, s', $I(s) \not\subseteq I(s')$ and vice versa.

Theorem 4.2. *(LBL-SS-NIZK $\Rightarrow$ LBL-AND-SS-NIZK.) Given any two languages $\mathcal{L}$ and $\hat{\mathcal{L}}$ and two labelled USS-NIZKs for both $\mathcal{L}$ and $\hat{\mathcal{L}}$, there exists a fully black-box construction of labelled USS-NIZK for language $(\mathcal{L} \wedge \hat{\mathcal{L}})$.*

Proof. Let Π_1 and Π_2 be two labelled USS-NIZKs for $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. We construct a LBL-USS-NIZK, $\tilde{\Pi}$, for $\mathcal{L}_1 \wedge \mathcal{L}_2$ with labels of length $\mathsf{len}(\ell) := \mathrm{poly}_\ell(\kappa)$.

Let u_i be the label bit-size supported by Π_i and let $u = \min(u_1, u_2)$. We require u be polynomial in κ so that polynomial number of random independent samplings from $\{0,1\}^u$ produces collisions with negligible probability. Let $v := \mathsf{len}(x_1) + \mathsf{len}(x_2) + \mathsf{len}(\ell)$ where $\mathsf{len}(x_b)$ $(b = 1, 2)$ denotes the bit length of the instances in $\mathcal{L}_b$ at security parameter κ. Let $n := v + \lceil \log_2(v) \rceil$.

- Given κ as input, $\tilde{\Pi}.\mathsf{Crs}$ runs $\sigma_{bi} \leftarrow \Pi_b.\mathsf{Crs}(1^\kappa)$ for $b \in \{1,2\}$ and $i \in [n]$ and outputs $\widetilde{\sigma} := (\sigma_{11}, \sigma_{21}, \dots, \sigma_{1n}, \sigma_{2n})$.
- Given $\widetilde{\sigma}$, $\widetilde{x} = (x_1, x_2)$, a label ℓ and $\widetilde{w} = (w_1, w_2)$, $\tilde{\Pi}.\mathsf{Prv}$ chooses a random u-bitstring $r \leftarrow \{0,1\}^u$ and computes $\pi_{bi} \leftarrow \Pi_b.\mathsf{Prv}(\sigma_{bi}, x_b, r, w_b)$ for every $i \in I(\widetilde{x}||\ell)$. It produces $\widetilde{\pi}$ by concatenating all the previous proofs with r.
- Given $\widetilde{\sigma}$, $\widetilde{x}$, ℓ and $\widetilde{\pi}$ as input, the verification algorithm $\tilde{\Pi}.\mathsf{Vrf}$ verifies the proofs included in $\widetilde{\pi}$ on the corresponding σ_{bi}, $b \in \{1,2\}$, $i \in I(\widetilde{x}||\ell)$. It accepts the proof if and only if all verifications succeed.

Simulators are constructed accordingly and zero knowledge holds immediately from Π_1 and Π_2. For unbound simulation soundness, suppose that, after seeing simulated proofs $\widetilde{\pi}_j$ for chosen label-instance pairs $(\widetilde{x}_j, \ell_j)$, an adversary outputs a proof $\widetilde{\pi}^*$ (consisting of π^*_{bi} and r^*) on a fresh $(\widetilde{x}^*, \ell^*)$. Let $b^* \in \{1,2\}$ be s.t. $x^*_{b^*}$ is a no-instance with respective language, which exists if the above is a valid forgery for the conjunction. Now, if there exists j s.t. $r^* = r_j$, let i^* be such that $i^* \in I(\widetilde{x}^*||\ell^*)$ and $i^* \notin I(\widetilde{x}_j||\ell_j)$. Otherwise, r^* is fresh, let i^* be any index from $I(\widetilde{x}^*||\ell^*)$. Observe that $(\pi^*_{b^*i^*}, x^*_{b^*}, r^*)$ is a forgery against Π_{b^*} with respect to σ_{i^*}, if every r_j is unique as expected. That is because the chosen $(x^*_{b^*}, r^*)$ is a fresh instance-label pair and $x^*_{b^*}$ is a no-instance of the respective language. □

4.2 Implications and Language Preserving Reductions

We first show that a labelled USS-NIZK for $\mathcal{L}$ can be constructed from (non-labelled) USS-NIZK for $\mathcal{L} \wedge \hat{\mathcal{L}}$.

Theorem 4.3. *(AND-USS-NIZK ⇒ LBL-USS-NIZK) Given any NIZK system for $\mathcal{L} \wedge \hat{\mathcal{L}}$ that is unbounded simulation sound and adaptive zero-knowledge, there exists a fully black-box construction of LBL-USS-NIZK for $\mathcal{L}$.*

Proof. Let $\Pi := \{\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim}\}$ be a USS-NIZK for $\mathcal{L} \wedge \hat{\mathcal{L}}$. It is assumed that $\hat{\mathcal{L}}$ is efficiently and uniformly sampleable (with witnesses) and includes a sufficiently large number of instances. We construct a LBL-USS-NIZK $\tilde{\Pi}$ for $\mathcal{L}$ with labels $\mathsf{len}(\ell) := \mathrm{poly}_\ell(\kappa)$ as follows. Let n be $n := \mathsf{len}(\ell) + \lceil \log_2(\mathsf{len}(\ell)) \rceil$ and function I as defined before.

- Given κ, $\tilde{\Pi}.\mathsf{Crs}$ outputs $\widetilde{\sigma} := (\sigma_1, \dots, \sigma_n)$, where $\sigma_i \leftarrow \Pi.\mathsf{Crs}(1^\kappa)$.
- Given $\widetilde{\sigma}$, instance $x \in \mathcal{L}$, a label ℓ and a witness w, $\tilde{\Pi}.\mathsf{Prv}$ samples a random yes-instance $\hat{x} \leftarrow \hat{\mathcal{L}}_\kappa$ with corresponding witness $\hat{w}$. It then creates a proof $\tilde{\pi}$ by concatenating $\hat{x}$ with all proofs $\Pi.\mathsf{Prv}(\sigma_i, (x, \hat{x}), (w, \hat{w}))$ for $i \in I(\ell)$.
- The verification algorithm $\tilde{\Pi}.\mathsf{Vrf}$ verifies the proofs in $\tilde{\pi}$ with the corresponding σ_i in $i \in I(\ell)$. It accepts $\tilde{\pi}$ only if all verifications succeed.

Simulators are constructed accordingly and the zero knowledge property of $\tilde{\Pi}$ is inherited from the one by Π. For simulation soundness, consider an adversary who produces a proof $\widetilde{\pi}^*$ on a fresh (x^*, ℓ^*), where x^* is a no-instance of $\mathcal{L}$. If a proof on (x^*, ℓ) was never asked by the adversary for any ℓ, $\widetilde{\pi}^*$ cannot be valid due to the USS of Π. Otherwise, observe that if $\widetilde{\pi}^* = (\hat{x}^*, \{\widetilde{\pi}_i^*\}_{i \in I(\ell^*)})$ is valid, the USS of Π is compromised, because there must exist an index i s.t. $(x^*, \hat{x}^*)$ has not been proven with respect to σ_i, but $\widetilde{\pi}_i^*$ is a valid proof for that instance. Observe that the above reasoning requires that the probability of collisions when sampling $\hat{x} \leftarrow \hat{\mathcal{L}}_\kappa$ is negligible, which is guaranteed by the assumption on $\hat{\mathcal{L}}$. $\square$

Corollary 2. *AND-USS-NIZK $\Rightarrow$ LBL-AND-USS-NIZK*
If $\hat{\mathcal{L}}$ is a *hard language* we can reduce OR-USS-NIZK to LBL-USS-NIZK.

Theorem 4.4. *(OR-USS-NIZK $\Rightarrow$ LBL-USS-NIZK) Any NIZK system for $\mathcal{L} \vee \hat{\mathcal{L}}$ for a hard language $\hat{\mathcal{L}}$ that is unbound simulation sound and adaptive zero-knowledge, can be transformed into a LBL-USS-NIZK for $\mathcal{L}$ in a black-box way.*

Proof. (Sketch) The transformation is analogous to the one provided in the proof of Theorem 4.3. The difference is that $\hat{x}$ is chosen to be a no-instance from $\hat{\mathcal{C}}_\kappa$ and its witness $\hat{w}$ together with $\hat{x}$ is included in the proof $\widetilde{\pi}$. The verifier algorithm checks that $\hat{x} \in \hat{\mathcal{C}}_\kappa$ using $\hat{w}$. Everything else remains unchanged. $\square$

Finally, the following result holds in the full verification model.

Corollary 3. *(NIZK $\not\Rightarrow_*$ USS-NIZK in the full verification model.) There does not exist an oracle machine* M *such that for every language $\mathcal{L}$ and for every NIZK Π for $\mathcal{L}$,* $\mathsf{M}^{\Pi,\mathcal{L}}$ *is a USS-NIZK for $\mathcal{L}$.*

Proof. Suppose that a USS-NIZK for $\mathcal{L}$ is black-box constructable from a $NIZK$ for $\mathcal{L}$ in the full verification model. Then, by applying the construction to $\mathcal{L} := (\mathcal{L}' \wedge \hat{\mathcal{L}}')$, we can construct a USS-NIZK for $(\mathcal{L}' \wedge \hat{\mathcal{L}}')$ from a $NIZK$ for $(\mathcal{L}' \wedge \hat{\mathcal{L}}')$. Since USS-NIZK implies $NIZK$, we could start from a USS-NIZK for $\mathcal{L}'$ to construct a USS-NIZK for $(\mathcal{L}' \wedge \hat{\mathcal{L}}')$, which contradicts Theorem 4.1. $\square$

5 Signatures from USS-NIZK w/o Language Extension

We begin with a simple yet useful case where a USS-NIZK system Π for hard language $\mathcal{L}$ is *perfectly no-instance simulation correct*, i.e., $\Pi.\mathsf{PrvSim}$ works for any no-instances in $\mathcal{C}$. Let $\mathcal{H}$ be a family of functions $\mathcal{H} := \{H_\kappa : \mathcal{K}_\kappa \times \mathcal{M}_\kappa \to \mathcal{C}'_\kappa\}$ that maps messages in $\mathcal{M}_\kappa$ to a subset of no-instances $\mathcal{C}'_\kappa \subseteq \mathcal{C}_\kappa$. We construct a signature scheme $\Sigma := (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Vrf})$ as follows.

```
$\Sigma.\mathsf{Setup}(1^\kappa)$ :
    $(\sigma, \tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa)$
    $K \leftarrow \mathcal{K}_\kappa$
    $pk := (\sigma, K)$
    $sk := \tau$
    return $(pk, sk)$

$\Sigma.\mathsf{Sign}(pk, sk, m)$ :
    $(\sigma, K) \leftarrow pk$
    $\tau \leftarrow sk$
    $x \leftarrow H_\kappa(K, m)$
    $\sigma \leftarrow \Pi.\mathsf{PrvSim}(\sigma, x, \tau)$
    return $\sigma$

$\Sigma.\mathsf{Vrf}(pk, m, \sigma)$ :
    $(\sigma, K) \leftarrow pk$
    $x \leftarrow H_\kappa(K, m)$
    $b \leftarrow \Pi.\mathsf{Vrf}(\sigma, x, \sigma)$
    return $b$
```

Since each message is mapped to a no-instance exclusively, simulation soundness is literally translated into EUF-CMA: It is hard to find new message m^* (new no-instance x^*) and valid signature σ^* (valid proof π^*) after seeing signatures σ_i (simulated proofs π_i) for arbitrary messages m_i (arbitrary no-instances x_i). Due to space constraint, we only show a formal statement below.

Theorem 5.1. *The above Σ is a EUF-CMA secure signature scheme for message space $\mathcal{M}_\kappa$ if, Π is a perfectly no-instance simulation correct USS-NIZK system for hard language $\mathcal{L}$ accompanied by $\mathcal{C}$, and $\mathcal{H}$ is a family of efficiently sampleable injections from $\mathcal{M}_\kappa$ to any $\mathcal{C}'_\kappa \subseteq \mathcal{C}_\kappa$.*

Now we proceed to more general case. We first introduce building blocks and establish some technical lemmas before presenting the construction.
Extended Target-Collision-Resistant Functions. A family of functions $\{H\}$ is target-collision-resistant if any p.p.t. adversary $\mathcal{A}$ wins in the following experiment only with negligible probability, say ϵ_{tcr}: $\mathcal{A}$ chooses an input x and it is given a random key K; $\mathcal{A}$ wins if it can produce a different input x' such that $H(K,x) = H(K,x')$. This notion was extended by Halevi and Krawczyk [28] in such a way that the adversary is allowed to select a different key for the second evaluation, i.e., the probability that the adversary comes up with a new x' and K' satisfying $H(K,x) = H(K',x')$ is upper bound by a negligible function ϵ_{etcr}. Hülsing et al. [30] considered a further extension called multi-target extended target-collision-resistant (m-eTCR) hash functions, where the above experiment is hard even if the adversary is allowed to choose several targets. More precisely:

Definition 5.1. *A family of functions $\mathcal{H} = \{H : \{0,1\}^{k(\kappa)} \times \{0,1\}^{m(\kappa)} \to \{0,1\}^{h(\kappa)}\}_{\kappa\in\mathbb{N}}$ (for certain polynomials k, m, h in κ) is said to be $\epsilon_{\mathsf{metcr}}$-multi-target extended target-collision-resistant if for every p.p.t. adversary $\mathcal{A}$ and every sufficiently large κ, it holds that*

$$\Pr\left[(\hat{x},\hat{K}) \leftarrow \mathcal{A}^{\mathsf{Key}(\cdot)}(1^\kappa) \ : \ \begin{array}{c} \exists (x_i,K_i) \in Q \text{ such that } \hat{x} \neq x_i \text{ and} \\ H(\hat{K},\hat{x}) = H(K_i,x_i) \end{array} \right] < \epsilon_{\mathsf{metcr}}(\kappa)$$

where $\mathsf{Key}(\cdot)$ is an oracle that on input $x_i \in \{0,1\}^{m(\kappa)}$, samples K_i uniformly at random from $\{0,1\}^{k(\kappa)}$, stores the pair (x_i,K_i) in Q and returns K_i.

Clearly, $\epsilon_{\mathsf{metcr}} \leq q \cdot \epsilon_{\mathsf{etcr}}$ holds for up to q queries. Though we use m-eTCR in our construction for simplicity of the argument, the same argument holds with standard single-target eTCR with polynomial loss in the security bound. We also note that, according to [40], eTCR can be constructed easily from TCR by appending the key to the output. That is, $H(K,m)||K$ is extended target-collision-resistant if H is target-collision-resistant.

Given a message m, we will output a no-instance by computing $x = H(K,m)$ for a randomly chosen $K \in \mathcal{K}$ and then returning $\mathcal{D}_{\mathcal{C}}(x)$. One could expect that the output of eTCR functions distributes uniformly over all possible values, but collision-resistance is not enough to guarantee such a property. (Consider an eTCR family of functions that output bitstrings where the last bit is constantly zero, i.e., non-uniform.) To overcome this limitation, we assume an additional property on the m-eTCR family: ϵ_{reg}*-regularity*. Roughly, every function in the family must be statistically close to the uniform distribution over its output.

Definition 5.2. *We say a family of functions $\mathcal{H} = \{H : \{0,1\}^{k(\kappa)} \times \{0,1\}^{m(\kappa)} \to \{0,1\}^{h(\kappa)}\}_{\kappa\in\mathbb{N}}$ is ϵ_{reg}-regular if for every sufficiently large κ and every $x \in \{0,1\}^{m(\kappa)}$, the distribution $\mathcal{D}_x$ defined as $\mathcal{D}_x := (K \leftarrow \{0,1\}^{k(\kappa)}\,;\, \mathsf{return}\, H(K,x))$ is statistically close to the uniform distribution over $\{0,1\}^{h(\kappa)}$. More precisely,*

$$\Delta(\mathcal{D}_x,\, U_{h(\kappa)}) := \frac{1}{2} \sum_{\alpha\in\{0,1\}^{h(\kappa)}} \left| \Pr[\mathcal{D}_x = \alpha] - \underbrace{\Pr[U_{h(\kappa)} = \alpha]}_{1/2^{h(\kappa)}} \right| < \epsilon_{\mathsf{reg}}(\kappa) \enspace .$$

The following lemma allows us to argue that the distribution of no-instances produced from messages is indistinguishable from yes-instances.

Lemma 5.1. *Let $\mathcal{L}_\kappa$ be a ϵ_{hd}-hard language (with respect to $\mathcal{C}_\kappa$) with sampling distributions $(\mathcal{D}_{\mathcal{L}_\kappa}, \mathcal{D}_{\mathcal{C}_\kappa})$ where $\mathcal{D}_{\mathcal{C}_\kappa} : \{0,1\}^{h(\kappa)} \to \mathcal{C}_\kappa$ (for certain polynomial h in κ). Let $\mathcal{H} = \{H : \mathcal{K}_\kappa \times \mathcal{M}_\kappa \to \{0,1\}^{h(\kappa)}\}_{\kappa\in\mathbb{N}}$ be a $\epsilon_{\mathsf{metcr}}$-multi-target extended target-collision-resistant function family that is ϵ_{reg}-regular. Consider the distribution $\mathcal{D}_m$ defined as $K \leftarrow \mathcal{K}_\kappa; \mathsf{return}\, \mathcal{D}_{\mathcal{C}_\kappa}(H(K,m))$. For every $m \in \mathcal{M}_\kappa$ and every sufficiently large κ, $\Delta(\mathcal{D}_m, \mathcal{D}_{\mathcal{C}_\kappa}) < \epsilon_{\mathsf{reg}}(\kappa)$.*

Proof. Observe that for every pair of random variables X, Y and every function F whose domain is the range of X and Y, it holds[4] $\Delta(F(X), F(Y)) \leq \Delta(X, Y)$. In our case, for every $m \in \mathcal{M}_\kappa$ and every sufficiently large κ,

$$\begin{aligned}\Delta(\mathcal{D}_m, \mathcal{D}_{\mathcal{C}_\kappa}) &= \Delta(K \leftarrow \mathcal{K}_\kappa\,;\, \mathsf{return}\, \mathcal{D}_{\mathcal{C}_\kappa}(H(K,m)),\, x \leftarrow U_{h(\kappa)}\,;\, \mathsf{return}\, \mathcal{D}_{\mathcal{C}_\kappa}(x)) \\ &\leq \Delta(K \leftarrow \mathcal{K}_\kappa\,;\, \mathsf{return}\, H(K,m),\, U_{h(\kappa)}) < \epsilon_{\mathsf{reg}}(\kappa) \enspace . \qquad \square\end{aligned}$$

We expect that distribution $\mathcal{D}_{\mathcal{C}_\kappa}$ is close to an injection, having small collision probability. This implies that instances in $\mathcal{C}$ have a short witness.

Definition 5.3 (Collision probability). *A function $f : \{0,1\}^{m(\kappa)} \to \{0,1\}^{n(\kappa)}$ for some polynomials m,n in κ is said have ϵ_{cp}-*collision probability *(for some function ϵ_{cp} in κ) if for every sufficiently large κ, it holds*

$$\left| \left\{ x \in \{0,1\}^{m(\kappa)} : \exists y \in \{0,1\}^{m(\kappa)} \text{ such that } x \neq y \wedge f(x) = f(y) \right\} \right| < \epsilon_{\mathsf{cp}}(\kappa) \cdot 2^{m(\kappa)} \enspace .$$

Let $(\mathcal{L}_\kappa, \mathcal{C}_\kappa)$ be a ϵ_{hd}-hard promise problem over efficiently sampleable distributions $(\mathcal{D}_{\mathcal{L}_\kappa}, \mathcal{D}_{\mathcal{C}_\kappa})$ where $\mathcal{D}_{\mathcal{C}_\kappa} : \{0,1\}^{h(\kappa)} \to \mathcal{C}_\kappa$ (for certain polynomial h in κ) has ϵ_{cp}-collision probability. Let $\mathcal{H} := \{H_\kappa : \mathcal{K}_\kappa \times \mathcal{M}_\kappa \to \{0,1\}^{h(\kappa)}\}_{\kappa\in\mathbb{N}}$ be a $\epsilon_{\mathsf{metcr}}$-multi-target extended target-collision-resistant function family that is ϵ_{reg}-regular. Let $\Pi := (\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ be a simulation sound non-interactive zero-knowledge proof system.

Figure 2 defines the signature scheme $\Sigma := (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Vrf})$. For correctness we only show the bound here.

[4] We abuse notation and write $F(X)$ to denote the composition $F \circ X$, i.e., the distribution $x \leftarrow X\,;\, \mathsf{return}\, F(x)$.

```
• $\Sigma.\mathsf{Setup}(1^\kappa)$ :
    $(\sigma,\tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa)$
    $pk := (\sigma, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa})$
    $sk := \tau$
    return $(pk, sk)$

• $\Sigma.\mathsf{Sign}(pk, sk, m)$ :
    $K \leftarrow \mathcal{K}_\kappa$
    $y := \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K, m))$
    $\pi \leftarrow \Pi.\mathsf{PrvSim}(\sigma, y, \tau)$
    $\sigma := (\pi, K)$
    return $\sigma$

• $\Sigma.\mathsf{Vrf}(pk, m, \sigma)$ :
    parse $\sigma$ as $(\pi, K)$
    $y := \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K, m))$
    return $\Pi.\mathsf{Vrf}(\sigma, y, \pi)$
```

Fig. 2: Construction of signature scheme from SS-NIZK

Theorem 5.2 (Correctness). *The signature scheme Σ defined above is correct. Concretely, for every message $m \in \mathcal{M}_\kappa$ and for every sufficiently large κ,*

$$\Pr\left[\, (pk, sk) \leftarrow \Sigma.\mathsf{Setup}(1^\kappa)\ ;\ \sigma \leftarrow \Sigma.\mathsf{Sign}(pk, sk, m) : 1 = \Sigma.\mathsf{Vrf}(pk, m, \sigma) \,\right] > 1 - \epsilon_{\mathsf{zk}}(\kappa) - \epsilon_{\mathsf{co}}(\kappa) - \epsilon_{\mathsf{hd}}(\kappa) - 2\epsilon_{\mathsf{reg}}(\kappa)\ .$$

Proof. For every $m \in \mathcal{M}_\kappa$,

$$\Pr\left[\, (pk, sk) \leftarrow \Sigma.\mathsf{Setup}(1^\kappa)\ ;\ \sigma \leftarrow \Sigma.\mathsf{Sign}(pk, sk, m) : 1 = \Sigma.\mathsf{Vrf}(pk, m, \sigma) \,\right]$$
$$= \Pr\left[\begin{array}{r} K \leftarrow \mathcal{K}_\kappa (\sigma,\tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa) \\ y := \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K, m)) ;\pi \leftarrow \Pi.\mathsf{PrvSim}(\sigma, y, \tau) \end{array} : 1 = \Pi.\mathsf{Vrf}(\sigma, y, \pi)\right]$$

which, by Lemma 5.1 (and for every sufficiently large κ) is greater or equal than

$$\Pr\left[\begin{array}{c} y \leftarrow \mathcal{D}_{\mathcal{C}_\kappa} (\sigma,\tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa) \\ \pi \leftarrow \Pi.\mathsf{PrvSim}(\sigma, y, \tau) \end{array} : 1 = \Pi.\mathsf{Vrf}(\sigma, y, \pi)\right] - 2\epsilon_{\mathsf{reg}}(\kappa)$$

which, by Lemma 2.2, is greater than $1 - \epsilon_{\mathsf{zk}}(\kappa) - \epsilon_{\mathsf{co}}(\kappa) - \epsilon_{\mathsf{hd}}(\kappa) - 2\epsilon_{\mathsf{reg}}(\kappa)$. □

Theorem 5.3 (Unforgeability). *The signature scheme Σ defined above is existentially unforgeable against adaptive chosen message attacks. In particular, for every p.p.t. adversary $\mathcal{A}$ against the EUF-CMA experiment of Σ that makes at most q queries to its signing oracle, there exists a p.p.t. algorithm $\mathcal{B}$ such that*

$$\mathrm{Adv}^{\mathsf{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(\kappa) \leq \mathrm{Adv}^{\mathsf{USS}}_{\Pi,\mathcal{B}}(\kappa) + \epsilon_{\mathsf{metcr}}(\kappa) + q\epsilon_{\mathsf{cp}}(\kappa) + 2q\epsilon_{\mathsf{reg}}(\kappa)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \mathsf{poly}(\kappa)$ *where* $\mathsf{poly}(\kappa)$ *is independent of* $\mathsf{Time}(\mathcal{A})$. *(Note that factor q multiplies to statistical errors only.)*

Proof. For every adversary $\mathcal{A}$ against the signature scheme, we build an attacker $\mathcal{B}$ against the *simulation soundness* of the underlying Π primitive. $\mathcal{B}$ is given the security parameter κ and a common reference string σ and oracle access to $\Pi.\mathsf{PrvSim}(\sigma, \cdot, \tau)$, where τ is the trapdoor associated to σ. $\mathcal{B}$ wins the game if it can produce a valid proof on a no-instance that was not queried to its oracle. $\mathcal{B}$ sends the public key $pk = (\sigma, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa})$ to $\mathcal{A}$. $\mathcal{A}$ is allowed to ask for valid signatures of messages of its choice. On input m_i, $\mathcal{B}$ produces a valid signature by sampling $K_i \leftarrow \mathcal{K}_\kappa$, computing $y_i = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K_i, m_i))$ and calling its oracle, getting $\pi_i = \Pi.\mathsf{PrvSim}(\sigma, y_i, \tau)$. Now, $\mathcal{B}$ returns $\sigma_i = (\pi_i, K_i)$ as to $\mathcal{A}$ as a signature for m_i. Eventually, $\mathcal{A}$ will come up with a pair $(\hat{m}, \hat{\sigma})$ such that $\hat{m} \neq m_i$ for every i. At this moment, $\mathcal{B}$ parses $\hat{\sigma}$ as $(\hat{\pi}, \hat{K})$ and computes $\hat{y} = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(\hat{K}, \hat{m}))$ and returns $(\hat{y}, \hat{\pi})$ as the solution for its challenge.

Note that $\mathcal{B}$ succeeds in simulating the EUF-CMA experiment correctly. Some signing queries can result into invalid signatures (although only with negligible probability), i.e., for some indices i, it is possible to have $\Pi.\mathsf{Vrf}(\sigma, y_i, \pi_i) = 0$. But this is not a problem, since in the real EUF-CMA experiment this event occurs with the same probability. We define the bad event, $\mathsf{Bad} \equiv$ *'There exists i such that $y_i = \hat{y}$'*. Note that, if Bad does not occur, then $\mathcal{B}$ wins if so does $\mathcal{A}$. More precisely, $\Pr[\mathcal{B} \text{ wins}] \geq \Pr[\mathcal{A} \text{ wins} \mid \neg\mathsf{Bad}] \geq \Pr[\mathcal{A} \text{ wins}] - \Pr[\mathsf{Bad}]$ or equivalently, $\Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{B} \text{ wins}] + \Pr[\mathsf{Bad}]$. Note that $\Pr[\mathsf{Bad}] \leq \max_{p.p.t.\ M} \{\Pr[E_M]\}$, where for a fixed M, the probability of event E_M is defined as

$$\Pr\left[\begin{array}{l} (\sigma,\tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa) \\ (\hat{m}, (\hat{K}, \hat{\pi})) \leftarrow M^{\mathsf{Sign}(\cdot)}(\sigma, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa}) \end{array} : \begin{array}{l} \exists (m_i, K_i, \pi_i) \in Q \text{ such that} \\ \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K_i, m_i)) = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(\hat{K}, \hat{m})) \end{array}\right]$$

where $\mathsf{Sign}(\cdot)$ is an oracle that, on input m_i, samples $K_i \leftarrow \mathcal{K}_\kappa$, computes $y_i = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K_i, m_i))$ and $\pi_i = \Pi.\mathsf{PrvSim}(\sigma, y_i, \tau)$, adds (m_i, K_i, π_i) to Q and outputs (π_i, K_i). For every p.p.t. M, there exists a p.p.t. $\bar{M}$ such that the above probability is upper-bounded by the following ($\bar{M}$ is given the trapdoor τ):

$$\Pr\left[\begin{array}{l} (\sigma,\tau) \leftarrow \Pi.\mathsf{CrsSim}(1^\kappa) \\ (\hat{m}, \hat{K}) \leftarrow \bar{M}^{\mathsf{Key}(\cdot)}(\sigma, \tau, H, \mathcal{D}_{\mathcal{C}_\kappa}) \end{array} : \begin{array}{l} \exists (m_i, K_i) \in Q \text{ such that} \\ \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K_i, m_i)) = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(\hat{K}, \hat{m})) \end{array}\right]$$

where $\mathsf{Key}(\cdot)$ is an oracle that, on input m_i, samples $K_i \leftarrow \mathcal{K}_\kappa$ adds (m_i, K_i) to Q and returns K_i. Note that the sampling of the (σ, τ) using $\Pi.\mathsf{CrsSim}$ requires polynomial time, and therefore, that operation can be included inside the machine $\bar{M}$. Then, we have that $\max_{p.p.t.\ M} \{\Pr[E_M]\} \leq \max_{p.p.t.\ \bar{M}} \{\Pr[\bar{E}_{\bar{M}}]\}$ where the probability of event $\bar{E}_{\bar{M}}$ for a fixed algorithm $\bar{M}$ is defined as

$$\Pr\left[(\hat{m}, \hat{K}) \leftarrow \bar{M}^{\mathsf{Key}(\cdot)}(1^\kappa, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa}) : \begin{array}{l} \exists (m_i, K_i) \in Q \text{ such that} \\ \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(K_i, m_i)) = \mathcal{D}_{\mathcal{C}_\kappa}(H_\kappa(\hat{K}, \hat{m})) \end{array}\right] .$$

Now, let $\mathcal{X}_\kappa$ be the set of inputs to $\mathcal{D}_{\mathcal{C}_\kappa}$ that share an image, i.e.,

$$\mathcal{X}_\kappa = \{x \in \{0,1\}^{h(\kappa)} : \exists y \in \{0,1\}^{h(\kappa)} \text{ such that } x \neq y \wedge \mathcal{D}_{\mathcal{C}_\kappa}(x) = \mathcal{D}_{\mathcal{C}_\kappa}(y)\} .$$

Since $\mathcal{D}_{\mathcal{C}_\kappa}$ has ϵ_{cp}-collision probability, we have $|\mathcal{X}_\kappa| \leq \epsilon_{\mathsf{cp}} \cdot 2^{h(\kappa)}$. Now, the probability of Bad is upper-bounded by

$$\max_{p.p.t.\ \bar{M}} \left\{\Pr\left[(\hat{m}, \hat{K}) \leftarrow \bar{M}^{\mathsf{Key}(\cdot)}(1^\kappa, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa}) : \begin{array}{c} \exists (m_i, K_i) \in Q \text{ such that} \\ H_\kappa(K_i, m_i) = H_\kappa(\hat{K}, \hat{m}) \end{array}\right]\right\} +$$

$$\max_{p.p.t.\ \bar{M}} \left\{\Pr\left[\perp \leftarrow \bar{M}^{\mathsf{Key}(\cdot)}(1^\kappa, H_\kappa, \mathcal{D}_{\mathcal{C}_\kappa}) : \begin{array}{l} \exists (m_i, K_i) \in Q \text{ such that} \\ H_\kappa(K_i, m_i) \in \mathcal{X}_\kappa \end{array}\right]\right\} .$$

The $\epsilon_{\mathsf{metcr}}$-*multi-target extended target-collision-resistance* of function H_κ guarantees that the first summand of the above expression is upper-bounded by $\epsilon_{\mathsf{metcr}}(\kappa)$. On the other hand, if machine $\bar{M}$ performs q queries to its oracle $\mathsf{Key}(\cdot)$, the second summand is upper-bounded by $q(2\epsilon_{\mathsf{reg}}(\kappa) + \epsilon_{\mathsf{cp}}(\kappa))$, because, thanks to the ϵ_{reg}-regularity of H_κ, for every $m \in \mathcal{M}_\kappa$ (and for sufficiently large κ),

$$\Pr\left[K \leftarrow \mathcal{K}_\kappa : H_\kappa(K, m) \in \mathcal{X}_\kappa\right] < \Pr\left[x \leftarrow \{0,1\}^{h(\kappa)} : x \in \mathcal{X}_\kappa\right] + 2\epsilon_{\mathsf{reg}}(\kappa)$$

upper-bounded by $\epsilon_{\mathsf{cp}}(\kappa) + 2\epsilon_{\mathsf{reg}}(\kappa)$, so we apply the union bound over all q queries.

For every adversary $\mathcal{A}$ against the signature scheme, the described $\mathcal{B}$ is an adversary against the simulation soundness of the underlying NIZK such that

$$\mathrm{Adv}^{\text{EUF-CMA}}_{\Sigma_\Pi,\mathcal{A}}(\kappa) \leq \mathrm{Adv}^{USS}_{\Pi,\mathcal{B}}(\kappa) + \epsilon_{\mathsf{metcr}}(\kappa) + q\epsilon_{\mathsf{cp}}(\kappa) + 2q\epsilon_{\mathsf{reg}}(\kappa)\ . \qquad \square$$

Acknowledgments

We would like to thank the anonymous reviewers for their effort and valuable feedback. We also thank Carsten Baum and Kyosuke Yamashita for their helpful comments and Arnab Roy for his suggestion on labelled NIZKs.

References

1. M. Abe, M. Ambrona, and M. Ohkubo. *On Black-Box Extensions of Non-Interactive Zero-Knowledge Arguments, and Signatures Directly from Simulation Soundness.* Cryptology ePrint Archive, Report 2019/696.
2. M. Abdalla, F. Benhamouda, and D. Pointcheval. Disjunctions for hash proof systems: New constructions and applications. *EUROCRYPT 2015*, *LNCS* 9057, pp.69–100. Springer.
3. M. Abe, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Tagged one-time signatures: Tight security and optimal tag size. *PKC 2013*, *LNCS* 7778, pp.312–331.Springer.
4. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-Preserving Signatures and Commitments to Group Elements. *Journal of Cryptology*, 29(2):363–421, 2016.
5. N. Alamati, C. Peikert, and N. S.-Davidowitz. New (and Old) Proof Systems for Lattice Problems. In *PKC 2018*, *LNCS* 10770, pp.619–643. Springer.
6. B. Barak and M. Mahmoody-Ghidary. Lower bounds on signatures from symmetric primitives. In *FOCS 2007*, pp.680–688. IEEE Computer Society.
7. M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. *CRYPTO 1989*, *LNCS* 435, pp.194–211. Springer, 1989.
8. M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). *STOC 1988*, pp.103–112. ACM, 1988.
9. D. Boneh, P. Papakonstantinou, C. Rackoff, Y. Vahlis, and B. Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *FOCS*, pp.283–292. IEEE Computer Society, 2008.
10. D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. *EUROCRYPT 1998*, *LNCS* 1403, pp.59–71. Springer.

11. Z. Brakerski, J. Katz, G. Segev, and A. Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. *TCC 2011*, *LNCS* 6597, pp.559–578. Springer, 2011.
12. R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, R. D. Rothblum, and D. Wichs. Fiat-Shamir: from practice to theory. In *STOC 2019*, pp. 1082–1090.
13. J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. *EUROCRYPT 2009*, *LNCS* 5479, pp.351–368. Springer.
14. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. *CRYPTO 1994*,*LNCS* 839, pp. 174–187.
15. J.S. Coron. Optimal security proofs for PSS and other signature schemes. *EUROCRYPT 2002*, *LNCS* 2332, pp.272–287. Springer.
16. R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. *EUROCRYPT 2002*,*LNCS* 2332, pp. 45–64. Springer.
17. Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. *CRYPTO 2001*, *LNCS* 2139, pp.566–598. Springer.
18. S. Even, A.L. Selman, Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 1984.
19. U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *FOCS*, pp.308–317. IEEE Computer Society, 1990.
20. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. *CRYPTO 1986*, *LNCS* 263, pp.186–194. Springer.
21. S. Garg and D. Gupta. Efficient round optimal blind signatures. *EUROCRYPT 2014*, *LNCS* 8441, pp. 477–495, Springer.
22. C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. *STOC 2011*, pp.99–108. ACM, 2011.
23. Y. Gertner, T. Malkin, and S. Myers. Towards a separation of semantic and CCA security for public key encryption. In *TCC 2007 LNCS* 4392, pp.434–455. Springer.
24. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. *ASIACRYPT 2006*, *LNCS* 4284, pp.444–459. Springer.
25. J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. *EUROCRYPT 2006*, *LNCS* 4004, pp. 339–358.
26. J. Groth and M. Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable snarks. *CRYPTO 2017*, *LNCS* 10402, pp.581–612. Springer.
27. J. Groth and A. Sahai. Efficient Noninteractive Proof Systems for Bilinear Groups. *SIAM J. Comput.*, 41(5):1193–1232, 2012.
28. S. Halevi and H. Krawczyk. Strengthening digital signatures via randomized hashing. *CRYPTO 2006*, *LNCS* 4117, pp.41–59. Springer. Germany.
29. D. Hofheinz and Tibor . Jager. Tightly secure signatures and public-key encryption. *Des. Codes Cryptography*, 80(1): 29-61, 2016.
30. A. Hülsing, J. Rijneveld, and F. Song. Mitigating multi-target attacks in hash-based signatures. *PKC 2016*, *LNCS* 9614, pp.387–416. Springer.
31. R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. *STOC 1989*, pp.44–61. ACM.
32. C.S. Jutla and A. Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. *ASIACRYPT 2013*, *LNCS* 8269, pp.1–20. Springer.
33. J. Katz. Signatures from one-way functions. Random bits; Random thoughts about random things, Feb 4, 2010, 2010.

34. J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. *ASIACRYPT 2009*, *LNCS* 5912, pp.703–720. Springer.
35. E. Kiltz and H. Wee. Quasi-adaptive NIZK for linear subspaces revisited. *EUROCRYPT 2015*, *LNCS* 9057, pp.101–128. Springer.
36. S. Kim and D J. Wu. Multi-theorem preprocessing nizks from lattices. *CRYPTO 2018*, *LNCS* 10991, pp. 733–765, Springer.
37. B. Libert, M. Joye, M. Yung, and T. Peters. Concise multi-challenge cca-secure encryption and signatures with almost tight security. *ASIACRYPT 2014*, *LNCS* 8874, pp.1–21. Springer.
38. B. Libert, T. Peters, and M. Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. *CRYPTO 2015*, *LNCS* 9216, pp.296–316. Springer.
39. R C. Merkle. A certified digital signature. *CRYPTO'89*, *LNCS* 435, pp. 218–238.
40. I. Mironov. Domain extension for enhanced target collision-resistant hash functions. *FSE 2010*, *LNCS* 6147, pp.153–167. Springer.
41. T. Malkin, I. Teranishi, Y. Vahlis, and M. Yung. Signatures resilient to continual leakage on memory and computation. *TCC 2011*, *LNCS* 6597, pp. 89–106. Springer.
42. P. D. MacKenzie, T. Shrimpton, and M. Jakobsson. Threshold Password-Authenticated Key Exchange. *J. Cryptology*, 19(1): 27-66, 2006.
43. D. Pointcheval and J. Stern. Security proofs for signature schemes. *EUROCRYPT1996*, *LNCS* 1070, pp. 387–398. Springer.
44. C, Peikert and S. Shiehian. Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors. *CRYPTO 2019*, *LNCS* 11692, pp. 89–114. Springer.
45. C. Peikert and V. Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. *CRYPTO2008*, *LNCS* 5157, pp. 536–553. Springer.
46. C. Ràfols. Stretching groth-sahai: NIZK proofs of partial satisfiability. *TCC 2015*, *LNCS* 9015, pp. 247–276, Springer.
47. A. Rostovtsev and A. Stolbunov. Public-Key Cryptosystem Based on Isogenies. IACR Cryptology ePrint Archive, Report 2006/145.
48. R. D. Rothblum, A. Sealfon, and K. Sotiraki. Towards Non-Interactive Zero-Knowledge for NP from LWE. *PKC 2019*, *LNCS* 11443, pp. 472–503, Springer.
49. O. Reingold, L. Trevisan, and S.P. Vadhan. Notions of reducibility between cryptographic primitives. In *TCC 2004*, *LNCS* 2951, pp.1–20. Springer.
50. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS 1999*, pp.543–553. IEEE Computer Society.
51. A. Stolbunov. Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. *Adv. in Math. of Comm.*, 2010.
52. A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003.
53. D.R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? *EUROCRYPT 1998*, *LNCS* 1403. Springer.
54. E. Vahlis. *Cryptography: Leakage Resilience, Black Box Separations, and Credential-Free Key Exchange.* PhD thesis, Graduate Department of Computer Science, University of Toronto, 2010.
55. A. Yerukhimovich. *A Study of Separations in Cryptography: New Results and New Models.* PhD thesis, The Faculty of the Graduate School, Department of Computer Science, University of Maryland, 2011.
56. T. Yasuda, X. Dahan, Y.-J. Huang, T. Takagi, and K. Sakurai. MQ Challenge: Hardness Evaluation of Solving Multivariate Quadratic Problems. NIST Workshop on Cybersecurity in a Post-Quantum World, Apr, 2015. Cryptology ePrint Archive, Report 2015/275.