Black-Box Language Extension of Non-Interactive Zero-Knowledge Arguments

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 paper 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) [6] is a beneficial building block for efficiently 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 (a proof) that is indistinguishable from the one 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) [48] 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 [48]. 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., [6, 18, 24], and general constructions have been designed to strengthen them to SS-NIZK, e.g., [48, 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 [29, 50] (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, given a NIZK system for language $\mathcal{L}$ and given another $\hat{\mathcal{L}}$, we consider the question of whether it is possible to construct a NIZK system for $\mathcal{L} \diamond \hat{\mathcal{L}}$ where $\diamond \in \{\wedge, \vee\}$ in a black-box manner. Many non-black-box techniques for disjunctive language extension can be found in the literature, e.g., [1, 11, 12, 20, 23, 25, 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. Note that 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.

Some black-box language extensions are straightforward, e.g., a conjunctive extension from a NIZK for $\mathcal{L}$ to a NIZK for $\mathcal{L} \wedge \mathcal{L}$ (when the system is a standard NIZK, i.e., not simulation-sound) 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 to $\mathcal{L} \vee \mathcal{L}$, because the information about which of the two instances is correct (possibly both) will be leaked. Another example pointed out by Sahai [48] is the conjunctive extension in the case of USS-NIZKs where the above simple approach fails: given two simulated proofs (π_1, π_2) and (π'_1, π'_2), for statements (x_1, x_2) and (x'_1, x'_2) respectively, one could create a new proof (π_1, π'_2) for the new statement (x_1, x'_2), winning the simulation-soundness game (as long as either x_1 or x'_2 is a false statement).

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. In particular, as pointed out by Peikert and Vaikuntanathan [45], disjunctive language extensions would be extremely useful to construct NIZK systems based on lattice-based assumptions. In that work, the authors present a transformation that achieves disjunction for a limited predicate and, to the best of our knowledge, this direction of work is not completed as the state-of-the-art of NIZK systems for NP under lattice-based assumptions [36] are restricted to the co-called *preprocessing model*, introduced by De Santis et al. in [16]. Observe that NIZKs for disjunctive languages have a vast number of applications. Among them, an important example is the framework of electronic voting [12], 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. Figure 1 summarizes our contributions, which we further describe in the rest of this section.

Impossibility of Black-Box Disjunctive Extension of various NIZKs. When building disjunctive language extensions of NIZKs the main challenge (very roughly) is to define a prover algorithm that handles (yes,no)-instances without any trapdoor and without revealing which of the statements (if not both) is true. Unfortunately, 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 considering other missing cases.

A bit more formally, we follow the *oracle separation paradigm*, cf., [7, 22, 31, 47, 51] 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 there exists no NIZK system M for $\mathcal{L} \vee \hat{\mathcal{L}}$ with any $\hat{\mathcal{L}}$ 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. It bears similarity with the *simulatable adversary paradigm* in [21] that exploits the duality of the zero-knowledge simulator and the real prover to construct a meta-reduction [8, 13]. In our approach, we let

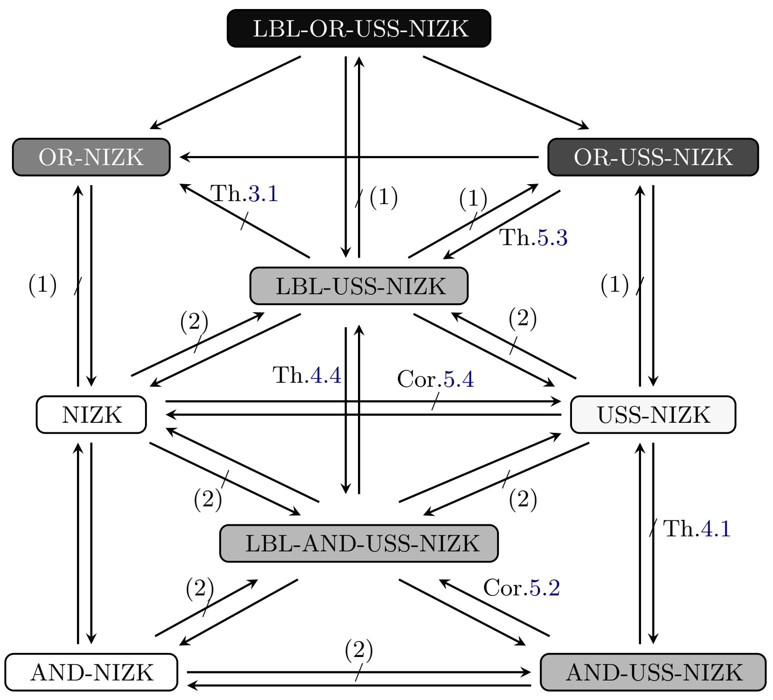

Fig. 1: Relations between variations of NIZK. Non-labelled edges correspond to straightforward black-box constructions. Separations labelled as (1) are implied by Theorem 3.1 (see Corollary 3.2). Those labelled as (2) are implied by Theorem 4.1 and hold in the *full-verification model* (see Definition 4.1 and Section 5.1).

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 verifier algorithm with the real oracle should never make a query on the fake witness, because otherwise the zero-knowledge property will be lost. 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 helpful and 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, 47] 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 [17,49] consisting of a pair of disjoint, efficiently sampleable, and indistinguishable languages, $\mathcal{L}$ and $\mathcal{C}$ (see Definition 2.1). Our result also extends to a 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, where 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*. Namely, every proof that is created internally with the prover algorithm must then be verified by the verification algorithm. See Definition 4.1 for a formal definition and Section 5.1 for more discussion on this model.

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. Such an impossibility (even in the weaker *full-verification model*) enlightens the essential difference between bounded and unbounded simulation soundness in the context of NIZKs.

Construction of Signatures without Language Extension. Motivated by our previous results about impossibility of language extensions, 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) and which is not possible as stated by our impossibility results. That is a sharp difference from previous works. 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 [4] and, unlike ours, does not allow us to conclude that *upgrading a NIZK to achieve unbounded simulation soundness 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 and they are publicly verifiable with the *crs* bound to the trapdoor. Unforgeability is argued based on the simulation

Ref.	Objective	Statements proved on the underlying NIZK
[48]	NIZK → OSS-NIZK	$R(x, \underline{\omega})$
[50]	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	$R(x, \underline{\omega}) \vee \underline{\sigma} = \mathsf{SIG}_{\underline{sk}}(vk)$
[5]	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.

soundness property. Our result is quite general in terms of the language that the underlying USS-NIZK must support. The assumption we make on the language is, roughly, that there exist efficient samplers $\mathcal{D}_{\mathcal{L}}$ and $\mathcal{D}_{\mathcal{C}}$ producing instances for $\mathcal{L}$ and $\mathcal{C}$ respectively and the produced instances are indistinguishable. We refer to Section 6 for more details.

1.3 Related Works

There exist several works for extending NIZKs to support disjunction of instances without reductions to NP-complete languages. In [12] 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 [19] (in the random oracle model [10]). Splitting a crs into two shares allows similar ideas if some algebraic properties are available. On the other hand, other works [20,41,46] follow an approach based on the Groth-Sahai proofs, which allow to prove disjunctive statements in some cases, e.g., [11,23]. Furthermore, Abdalla et al. [1] achieve disjunction through a smooth projective hash proof system [14].

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 [48] 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 [50] 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 [23], followed by other works [2, 11, 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, 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 [5] 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 [14] 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, $x \leftarrow X$ denotes that there exists an efficient sampling algorithm that takes some randomness and outputs an instance $x \in X$ with uniform distribution over X. If we need to be explicit about random coins, we write $x \leftarrow X(r)$ to represent that coin r is used to select instance x from X. 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 such that for every $\kappa \geq \kappa_0$ it holds $\epsilon(\kappa) < 1/p(x)$. negligible.

By $y \leftarrow A(x)$ we denote a process of computation where A takes x as input and outputs y. For oracle algorithm A^{O}, notation $y \leftarrow \mathsf{O}(x)$ denotes computation by O taking x as input given from A and output y returned to A. We also use the same notation $y \leftarrow \mathsf{O}(x)$ to denote input/output pair (y, x) with respect to O in order to make O explicit in the context. 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 $\mathsf{M.func}$ may be used if 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 [17, 49]. More formally:

Definition 2.1 (Hard Language). *Let R be an efficiently computable binary relation. For some fixed polynomials* poly_x *and* poly_w*, let $\mathcal{L}_\kappa := \{x : x \in \{0,1\}^{\mathrm{poly}_x(\kappa)} \wedge \exists w \in \{0,1\}^{\mathrm{poly}_w(\kappa)}$ s.t. $R(x, w) = 1\}$, 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 $\mathcal{D}_{\mathcal{C}_\kappa}$ sampling 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*

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

In this paper, we consider extending a language $\mathcal{L}$ with respect to conjunction or disjunction with an arbitrary language $\hat{\mathcal{L}}$ as formally defined below.

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 $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa := \{(x, \hat{x}) \mid (x \in \mathcal{L}_\kappa) \diamond (\hat{x} \in \hat{\mathcal{L}}_\kappa)\}$. The extension is said to be* non-trivial *if $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa \notin \mathcal{L}_{\kappa'}$ for any κ and κ'.*

Note that, for any non-empty finite $\mathcal{L}_\kappa$ and $\hat{\mathcal{L}}_\kappa$, we have $\mathcal{L}_\kappa \diamond \hat{\mathcal{L}}_\kappa \notin \mathcal{L}_\kappa$. We only consider non-trivial language extensions in this paper. A language extension of a NIZK (with respect to operator $\diamond$) consists of, given hard 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.*
- *$\pi \leftarrow \mathsf{Prv}(\sigma, x, \ell, w)$ 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, outputs either 1 or 0 representing acceptance or rejection, respectively.*

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

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

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 $R(x, w) = 0$. Otherwise, they return $\mathsf{Prv}(\sigma, x, \ell, w)$ *and* $\mathsf{PrvSim}(\sigma, x, \ell, \tau)$ *respectively.*

We call it *non-adaptive* multi-theorem zero-knowledge if all inputs to O are chosen at once by $\mathcal{A}$ before σ is generated.

Definition 2.5 (Unbounded Simulation Soundness). *A labelled non-interactive zero-knowledge argument system $\Pi :=$* $(\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ *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 $\mathcal{A}$,*

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

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

We call $(\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ a USS-NIZK system when it constitutes a non-interactive argument system that is zero-knowledge and unbounded simulation sound.

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 Π =* $(\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ *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*

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

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

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

Proof. We refer to Appendix A.1 for a formal proof. ∎

Definition 2.7 (No-instance simulation correctness). *A non-interactive argument system $\Pi = (\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim})$ 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*

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

is negligible in κ. Π 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. $\epsilon_{\mathsf{no}}(\kappa) \leq \epsilon_{\mathsf{zk}}(\kappa) + \epsilon_{\mathsf{co}}(\kappa) + \epsilon_{\mathsf{hd}}(\kappa)$.

Proof. We refer to Appendix A.1 for a formal proof. ∎

2.4 Fully Black-Box Construction and Separation

We follow the framework of fully black-box construction and separation in [31,47]. 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. That is, there is no fully black-box construction of primitive A based on B if there exists an oracle O for which there exists 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 (for any hard $\hat{\mathcal{L}}$). 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 to $\mathcal{L} \vee \hat{\mathcal{L}}$ even when compromising labelling or simulation soundness.

Theorem 3.1. *(LBL-USS-NIZK $\not\Rightarrow$ OR-NIZK) Given any hard language $\mathcal{L}$ and any labelled SS-NIZK system* L *for $\mathcal{L}$, there exists no fully black-box construction of NIZK scheme* M *for $\mathcal{L} \vee \hat{\mathcal{L}}$ with any hard language $\hat{\mathcal{L}}$ 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, and LBL-USS-NIZK.

Corollary 3.2. NIZK $\not\Rightarrow$ OR-NIZK, USS-NIZK $\not\Rightarrow$ OR-USS-NIZK, LBL-USS-NIZK $\not\Rightarrow$ LBL-OR-USS-NIZK

In the rest of this section, we prove Theorem 3.1. For that, following the standard oracle separation paradigm, 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 is equipped with two injections $H_c : \{0,1\}^\kappa \rightarrow \{0,1\}^{2\kappa}$ and $H_x : \{0,1\}^{\kappa+1} \rightarrow \{0,1\}^{2\kappa}$, and a permutation $H_p : \{0,1\}^{6\kappa} \rightarrow \{0,1\}^{6\kappa}$. 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 provides three language-related functionalities* $\mathsf{SmplYes}$*,* SmplNo*, and* $\mathsf{Promise}$*, and four NIZK-related functionalities,* Crs*,* Prv*,* PrvSim *and* Vrf *that:*

- *Given $(\mathsf{SmplYes}, w)$ for $w \in \{0,1\}^\kappa$, compute $x \leftarrow H_x(1||w)$, and output x.*
- *Given (SmplNo, w) for $w \in \{0,1\}^\kappa$, compute $x \leftarrow H_x(0||w)$, and output x.*
- *Given $(\mathsf{Promise}, x)$ for $x \in \{0,1\}^{2\kappa}$, output 0 if $\perp \leftarrow H_x^{-1}(x)$ and 1 otherwise.*
- *Given (Crs, τ) for $\tau \in \{0,1\}^\kappa$, compute $\sigma \leftarrow H_c(\tau)$ and output σ.*
- *Given $(\mathsf{Prv}, \sigma, x, \ell, w)$, output $\perp$ if $\perp \leftarrow H_c^{-1}(\sigma)$, $x \neq H_x(1||w)$, or $\ell \notin \{0,1\}^{2\kappa}$ happens. Otherwise, output $\pi \leftarrow H_p(\sigma||x||\ell)$.*
- *Given $(\mathsf{PrvSim}, \sigma, x, \ell, \tau)$, output $\perp$ if $\sigma \neq H_c(\tau)$, $\perp \leftarrow H_x^{-1}(x)$, or $\ell \notin \{0,1\}^{2\kappa}$ happens. Otherwise, output $\pi \leftarrow H_p(\sigma||x||\ell)$.*
- *Given $(\mathsf{Vrf}, \sigma, x, \ell, \pi)$, output 1 if $(\sigma||x||\ell) = H_p^{-1}(\pi)$. Output 0, otherwise.*

We denote by $\mathcal{O}_\kappa$ a set of all the oracles that follow the above syntax with security parameter κ, and by $\mathcal{O}$ the collection of $\mathcal{O}_\kappa$ for all $\kappa > 0$.

A query to O is *successful* if something other than $\perp$ (or 0 in the case of $\mathsf{O.Vrf}$) is returned. 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 a 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' 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 $S_1, S_2, \dots$ in such a way that, given a query of the form $(\mathsf{cmd}, \mathit{args})$, it first searches S_1 for matching entry $(\mathsf{cmd}, \mathit{args}, [\mathit{output}])$ 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 to fit to the definition of NIZK in 2.3 and 2.4 are needed.)

The following lemma holds for L^{O}.

Lemma 3.1. *For any $\mathsf{O} \in \mathcal{O}_\kappa$, L^{O} implements a hard promise problem $(\mathcal{L}_\kappa, \mathcal{C}_\kappa)$ 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 $\mathcal{L}_\kappa$ that is perfectly correct, perfectly yes-instance and no-instance simulation correct. Furthermore, it is adaptive zero-knowledge and unbound simulation-sound against polynomial-time adversaries given oracle access to O a polynomial number of times.*

Proof. We refer to Appendix A.2 for a formal proof. ■

We make some remarks about our design choices for O and the properties of L. It was shown in [9, 53] 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 $\widetilde{\sigma}$ (which we call *legitimate* crs's), we claim that the prover algorithm $\mathsf{M.Prv}$ cannot use them. Otherwise, the adaptive zero-knowledgeness will be compromised. That is, all crs's used in 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 not known to the prover. Using a legitimate crs only for yes-instances will lose zero-knowledgeness since the zero-knowledge simulator does not know whether the given x is a yes or a no-instance.

Let us elaborate on this point. Consider the adaptive zero-knowledge game where an adversary submits disjunctive instances $(x_j, \hat{x}_j)$ for $j = 1, \dots, q^c$ (for certain integer c) to the challenger that produces proofs either by $\mathsf{M.Prv}$ or $\mathsf{M.PrvSim}$. The adversary verifies the proofs by $\mathsf{M.Vrf}$ which may make verification queries $\mathsf{O.Vrf}$ on x_j. Let Γ_{11}^{S} denote the set of crs's given as input to $\mathsf{O.Vrf}$ to verify x_j for $j = 1, \ldots, q^c$ in the above verification when all $(x_j, \hat{x}_j)$ are taken from $\mathcal{L}_\kappa \times \hat{\mathcal{L}}_\kappa$ and proofs are made by $\mathsf{M.PrvSim}$. Similarly, let $\mathit{\Gamma}_{01}$ denote the set of crs's as well when all $(x_j, \hat{x}_j)$ are taken from $\mathcal{C}_\kappa \times \hat{\mathcal{L}}_\kappa$ and proofs are made by $\mathsf{M.Prv}$. Other combinations of suffixes are defined accordingly. Suppose the case where proofs are made by simulator $\mathsf{M.PrvSim}$. First observe that we can interpret the sets as distributions (quantified over the adversary and M). That way, the distributions of $\mathit{\Gamma}_{01}$ and $\mathit{\Gamma}^{\mathsf{S}}_{01}$ are indistinguishable due to the zero-knowledge property. It then hold that the distributions of $\mathit{\Gamma}^{\mathsf{S}}_{01}$ and $\mathit{\Gamma}^{\mathsf{S}}_{11}$ are indistinguishable because, otherwise, $\mathsf{M.PrvSim}$ can be used to distinguish between $\mathcal{L}$ and $\mathcal{C}$. Similarly, $\mathit{\Gamma}^{\mathsf{S}}_{11}$ and $\mathit{\Gamma}^{\mathsf{S}}_{10}$ are indistinguishable due to the indistinguishabilty of $\hat{\mathcal{L}}$ and $\hat{\mathcal{C}}$. Then, $\mathit{\Gamma}^{\mathsf{S}}_{10}$ and $\mathit{\Gamma}_{10}$ are indistinguishable due to the zero-knowledge property. Thus $\mathit{\Gamma}_{01}$ and $\mathit{\Gamma}_{10}$ are indistinguishable. Now observe that $\mathit{\Gamma}_{01}$ does not contain a legitimate crs since the real prover algorithm cannot prove on a no-instance with a legitimate crs whose trapdoor is not given. The same is true for $\mathit{\Gamma}_{10}$ because $\mathit{\Gamma}_{10}$ is indistinguishable from $\mathit{\Gamma}_{01}$ and because legitimate crs's can be identified (as in the *learning-phase* defined below). Thus, even if a witness for x_j is given, the prover algorithm must not use legitimate crs's to prove x_j.

We indeed show that such a NIZK system that does not use the legitimate crs for proving a given statement cannot be sound. Intuitively, if all crs's are generated internally, relevant trapdoors are available to the prover algorithm. A caveat is however that the prover algorithm must work only for yes-instances with correct witnesses and it is not clear how it is useful in forging proofs for no-instances. 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 $\mathcal{L}_\kappa$. It 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. It is indeed possible since all crs's are internally created within $\mathsf{M.Prv}$ so their trapdoors are known. There could be a case where a legitimate crs is used to prove x^*. Recall that $\mathit{\Gamma}_{10}$ 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 $\widetilde{\pi}$ and verified by $\mathsf{M.Vrf}$ 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 $\mathcal{A}$. Let $q > 2$ be a polynomial in the security parameter that represents the maximum number of queries to O that M^{O} can make in each invocation. Let $c > 1$ be a constant.

[Soundness Game]

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 a list of legitimate common reference strings and their trapdoors (σ_j, τ_j) that $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$ appears in this phase.

Step 2: Self-Learning Phase.
Given $\widetilde{\sigma}$, adversary $\mathcal{A}$ repeats $\widetilde{\pi}_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Prv}, \widetilde{\sigma}, (x_i, \hat{x}_i), (w_i, \hat{w}_i))$ and $b_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x_i, \hat{x}_i), \widetilde{\pi}_i)$ with uniformly sampled instances $(x_i, \hat{x}_i)$ and witness $(w_i, \hat{w}_i)$ from $\mathcal{L}_{\kappa} \times \hat{\mathcal{C}}_{\kappa}$ for q^c times.

Let Q_{triv} be a list of trivial CRS and its trapdoor, (σ_j, τ_j), that $[* \neq \perp] \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, *, *, \tau_j)$ or $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$ appear in an execution of M in this phase.

Step 3: Forgery Phase.
Sample $(x^*, \hat{x}^*) \in \mathcal{C}_{\kappa} \times \hat{\mathcal{C}}_{\kappa}$ by $w^* \leftarrow \{0,1\}^{\kappa}$, $x^* \leftarrow \mathsf{O}(\mathsf{SmplNo}, w^*)$ and $\hat{x}^* \leftarrow \hat{\mathcal{C}}_{\kappa}$. Let $\bar{x} := \mathsf{O}(\mathsf{SmplYes}, w^*)$. Apply instance swapping: define a partial oracle O' with 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''] Let Q_{intl} be an initially 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, do as follows.
  (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 register $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, \pi_j)$ to O'.
  (b) Else return $\pi_j \leftarrow \{0,1\}^{6\kappa}$ and register $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, \pi_j)$ and $(\mathsf{Vrf}, \sigma_j, x^*, \ell_j, \pi_j, 1)$ 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.
Given $(x^*, \hat{x}^*)$ and $\widetilde{\pi}^*$, the challenger outputs 1 if $x^* \notin \mathcal{L}_{\kappa}$, $\hat{x}^* \notin \hat{\mathcal{L}}_{\kappa}$, and $1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$ hold. It outputs 0, otherwise.

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

We will use the following lemma. 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 ([52, 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 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. The probability is taken over the choice of O and all 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 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 first some games (Game 0 to Game 6) we exclude events that happens only by chance and simplifies the game. Under the condition that these events do not happen, O'' simulates an oracle in $\mathcal{O}$ successfully making the forgery a correct proof on a yes instance in the disjunctive relation. Then in the succeeding games (Game 7 to 9) , we replace oracle O with O'' with step by step manner. In Game 7, we replace oracle O in the setup and self-learning phase with O''. In Game 8 we do the same in the final verification and argue that replacing oracle O with O'' in the final verification would not be noticed if M is zero-knowledge and relevant languages are hard. In the last Game 9, the adversary does nothing but creating a proof for a yes instance and it must be accepted in the final verification 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.

Any query to $\mathsf{O.PrvSim}$ and $\mathsf{O.Prv}$ with a crs without prior generation by $\mathsf{O.Crs}$ will be successful only if the crs is in the domain of H_c. It therefore happens with probability at most $1/2^{\kappa}$ over the choice of H_c. Any verification query on a proof without prior generation by $\mathsf{O.Prv}$ or $\mathsf{O.PrvSim}$ will output 1 with probability at most $1/2^{6\kappa}$. Given at most q queries in an execution of M and $2q^c+3$ executions of M throughout the game, there is at most $q(2q^c+3)(1/2^\kappa+1/2^{6\kappa})$ chance of halting the game by observing such a query in Game 1. We thus have $|P_1 - P_0| < q(2q^c+3)(1/2^\kappa+1/2^{6\kappa})$.

Game 2: Modify the final verification so that it does not check $x^* \notin \mathcal{L}_\kappa$ and $\hat{x}^* \notin \hat{\mathcal{L}}_\kappa$.

Since the condition is fulfilled due to the way that x^* and $\hat{x}^*$ are chosen, this modification does not change the outcome of the game. We thus have $P_2 = P_1$.

Game 3: Halt the game if $\mathcal{A}$ observes $b_i = 0$ in the self-learning phase. This is to exclude cases where the adversary learns nothing in the self-learning phase due to a bad choice of random coins given to $\mathsf{M.Prv}$. Since $\mathcal{A}$ behaves honestly in this phase, it happens only if a correctness error occurs. We thus have $|P_3 - P_2| < q^c\rho_{\mathsf{co}}(\kappa)$.

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.

Since w^* is chosen uniformly, the event happens only by chance among at most $q + q^c(1+2q)$ queries to O during the concerned phases. Hence $|P_4 - P_3| < 3(q + q^c(1+2q))/2^\kappa$.

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

There could be at most $q + 2q^{c+1} + q$ queries to O that may yield a proof by the end of the forgery phase. Thus uniformly chosen π_j could duplicate with probability at most $(2q+2q^{c+1})/2^{6\kappa}$. Taking union bound for at most q random π_j, we have $|P_5 - P_4| < q(2q+2q^{c+1})/2^{6\kappa}$.

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. First observe that $(\mathsf{Prv}, \sigma_j, x^*, \ell_j, w^*, [\pi_j])$ that causes $\pi_j \neq \pi'_j$ exists in O' only if π_j is randomly assigned at step (b) of algorithm O''. That step is executed only if (σ_j, τ_j) is not in $Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$. Observe that (σ_j, τ_j) must then present in Q_{leg} as we consider the case where σ_j is generated in advance since Game 1. For each (σ, τ) in Q_{leg}, the probability that it does not appear in the self-learning phase but appears for the first time in the forgery phase is upper-bounded by $2/(eq^c)$ due to Lemma 3.3. (Though O'' is used in the forgery phase whereas O is in the self-learning phase, simulation by O'' is perfect in this game until the time the considered event happens. Hence we can apply Lemma 3.3.) Since there are at most q entries in Q_{leg}, the probability that (σ_j, τ_j) in Q_{leg} appears for the first time in the forgery phase is at most $2/(eq^{c-1})$. Accordingly, $|P_6 - P_5| < 2/(eq^{c-1})$.

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' involves 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 change the view in the relevant phases. We thus have $P_7 = P_6$.

Game 8: In this game, we use O'' instead of O also in the final verification phase.

The view in the verification phase changes only if $\mathsf{M.Vrf}$ makes one of the following queries whose output differs in O and O''. Let O' be the partial oracle defined at the end of the forgery phase.

- $(\mathsf{PrvSim}, [\sigma_j], x^*, [\ell_j], [\tau_j])$ 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 $\pi'_j \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, x^*, \ell_j, \tau_j)$.
- A query that is not in O' but results in π_j already included in O'.
- A query included in O'.

Event that the first two queries are made can happen only by chance and we can evaluate its probability similarly as done in Games 5 and 6 respectively.

The third case requires careful analysis. We briefly present our idea for bounding the probability in the following. First observe that O' includes two types of queries. One is those involving witness w^* and the other is those verifying randomly assigned proofs. Regarding the first type, we follow the intuition that if verification function $\mathsf{M.Vrf}$ can see a witness for a given instance it is no longer zero-knowledge or the language itself is trivial. Regarding the second type, we follow the intuition discussed at the beginning of this section; legitimate non-trivial σ_j cannot be used to prove given instance otherwise zero-knowledgeness will be lost. Here we claim the following bound and present details afterwards.

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

Game 9: We then modify O'' so that it no longer uses O but instead uses random partial oracle R that, in combination with O', makes $\mathsf{O}'' = \mathsf{O}'||\mathsf{R}$ be one of oracles in $\mathcal{O}_\kappa$.

Since all queries to O'' from $\mathsf{M.Vrf}$ in Game 8 actually answered by O are consistent with partial oracle O', replacing that part with also consistent R does not change the distribution of the view of $\mathsf{M.Vrf}$. Hence we have $P_9 = P_8$.

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

By summing up the above differences of probabilities, we have

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

where

$$\begin{aligned}\epsilon(\kappa) :=& q(2q^c + 3)(1/2^{\kappa} + 1/2^{6\kappa}) + 3(q + q^c(1 + 2q))/2^{\kappa} \\ &+ q(2q + 2q^{c+1})/2^{6\kappa} + (3q^2 + 2q^{c+2})/2^{6\kappa} + 2q/2^{\kappa}\end{aligned}$$

that is negligible in κ. Accordingly, if M is correct and zero-knowledge and 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 probability not negligible in κ. ∎

Proof. (of Claim 3.3) For the first event, 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. We thus have a bound for this event as $2q/(eq^c)$.

For the second event, we apply the same argument as that for Game 5 except for counting additional q queries made during the verification itself. We thus have a bound for this event as $q(3q + 2q^{c+1})/2^{6\kappa}$.

Regarding the third event, observe that O' contains two types of queries. One is those that includes w^*, i.e., $(\mathsf{SmplYes}, w^*)$, (SmplNo, w^*), $(\mathsf{Prv}, *, \bar{x}, *, w^*)$, and $(\mathsf{Prv}, *, x^*, *, w^*)$. By AskW we denote an event that one of those queries appears. The other is queries of type $(\mathsf{Vrf}, *, x^*, *, [\pi_j])$ that verifies a randomly assigned proof π_j. By VerPi we denote an event that this type of query happens.

We first evaluate 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. It is important to observe at this point that the view produced by O'' with O' is consistent, i.e., there exists a partial oracle that produces the same view, and what is done in the forgery phase is creating a correct proof on a (yes,no)-instance 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 with $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$, respectively. Note that the trapdoor output by $\mathsf{M.CrsSim}$ is given to $\mathsf{M.PrvSim}$.

We can show that $|\Pr[\mathsf{AskW}^{8.1}] - \Pr[\mathsf{AskW}^{8.0}]| < \rho_{\mathsf{zk}}(\kappa)$ by constructing a distinguisher from the algorithm of $\mathcal{A}$. Since witness w^* is known to the distinguisher, event AskW can be observed. We then claim that $\Pr[\mathsf{AskW}^{8.1}] \leq q/2^{\kappa}$. The claim 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^*. Hence the event happens only by chance among q queries. We thus have $\Pr[\mathsf{AskW}] = \Pr[\mathsf{AskW}^{8.0}] < \rho_{\mathsf{zk}}(\kappa) + q/2^{\kappa}$.

Next we consider event VerPi. Recall that it captures the case where a randomly assigned proof π_j is verified by the verifier. Suppose that we run $\mathsf{M}^{\mathsf{O}''}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\pi})$ is done on a given input and observe that it makes a verification query $(\mathsf{Vrf}, \sigma_j, x^*, \ell_j, \pi_j)$ to O'', it is not clear if the query is the one that causes VerPi or not since we do not know whether π_j is a randomly assigned one or not. Namely, event VerPi is not observable. This will be an obstacle when we construct an adversary against zero-knowledgeness like done in the previous case. We therefore consider an alternative 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 the proof π_j is a randomly assigned one in step (b) of O'' or not. Observe that it is the case only if $(\sigma_j, [\tau_j]) \notin Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$ is satisfied. If σ_j is not an internally generated one, then it must have been generated in the setup phase. Furthermore, as σ_j appears in the final verification, it should have appeared in a verification during the self-learning phase as well. We thus define event VerCrs by final verification $\mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$ making a query $(\mathsf{Vrf}, [\sigma_j], x^*, [\ell_j], [\pi_j])$ that satisfies $\ell_j \in \{0,1\}^{2\kappa}$, $(\sigma_j, [\tau_j]) \notin Q_{\mathsf{triv}}$, 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 miss the case where a σ_j generated in the setup phase appears for the first time in the final verification. Applying Lemma 3.3, however, we can upper bound the probability for such event by q/eq^c. Regarding Q_{nt}, We thus have

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

Let Game 8.0′ be Game 8 and let $\mathsf{VerCrs}^{8.i'}$ denote the event that VerCrs happens in Game $8.i'$.

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

We claim that $|\Pr[\mathsf{VerCrs}^{8.1'}] - \Pr[\mathsf{VerCrs}^{8.0'}]| < \rho_{\mathsf{zk}}(\kappa) + 2q/eq^c$. To show this, we construct a zero-knowledge adversary that, given $\widetilde{\sigma}$, first execute the self-learning phase, and send $(x^*, \hat{x}^*)$ 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 event VerCrs happens) distributes as well as that in Game 8.0′. If, on the other hand, the challenger is working with $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$, the view in the final verification distributes in the same way as as that in Game 8.1′.

In the above argument, however, the adversary cannot actually perfectly capture event VerCrs since Q_{triv} and Q_{nt} the adversary obtains from its own self-learning and used 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 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 σ_j appeared during the final verification and present in $Q_{\mathsf{triv}} \setminus (Q_{\mathsf{triv}} \cap Q_{\mathsf{leg}})$ 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 while in a self-learning but does appear for the first time in the other self-learning. Thus, by applying Lemma 3.3 we can 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.

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

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

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

Due to the indistinguishability of $\mathcal{L}$ and $\mathcal{C}$, we have that $|\Pr[\mathsf{VerCrs}^{8.3'}] - \Pr[\mathsf{VerCrs}^{8.2'}]|$ is lower than $q/2^\kappa$.

$\mathsf{Game}\ 8.4'$: Replace $\mathsf{M.CrsSim}$ and $\mathsf{M.PrvSim}$ with $\mathsf{M.Crs}$ and $\mathsf{M.Prv}$, respectively. Note that witness $(\perp, \hat{w})$, where $\hat{w}$ is a witness for $\hat{x}$, is given as input.

We have $|\Pr[\mathsf{VerCrs}^{8.4'}] - \Pr[\mathsf{VerCrs}^{8.3'}]| < \rho_{\mathsf{zk}}(\kappa) + 2q/eq^c$.

Since w^* is no longer given, a valid proof π_j on x^* with a legitimate non-trivial σ_j that makes event $\mathsf{VerCrs}^{8.4'}$ to happen can be created only by chance by guessing a relevant trapdoor or the witness, which already excluded in previous games. Hence 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

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

as claimed. ■

4 Conjunctive Language Extension

4.1 Impossibility of AND-USS-NIZK from USS-NIZK

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.*

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. In the proof of our separation, 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 [22, 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). Note that, however, due to the difference between the scenarios, shielding internal proof queries in the proof algorithm (in our case) would make our oracle O mostly useless as a NIZK system, because the verification queries without a corresponding proof query would not be accepted for almost any inputs.

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.*

Proof. Suppose that there exists a construction M in $\mathcal{M}$ that is non-adaptive multi-theorem zero-knowledge and unbounded simulation sound. Let $\mathcal{A}$ be an adversary playing the simulation soundness game against M with challenger $\mathcal{B}$. They are equipped with oracle O, chosen uniformly from $\mathcal{O}_\kappa$, which defines languages $\mathcal{L}_\kappa$ and $\mathcal{C}_\kappa$. Let $\mathcal{L} \wedge \hat{\mathcal{L}}$ be the extended language that M is dedicated for.

$\mathcal{A}$ chooses a target of forgery $(x^*, \hat{x}^*)$ uniformly from the no-instances of form $\mathcal{C}_\kappa \times \hat{\mathcal{L}}_\kappa$. $\mathcal{A}$ also chooses no-instances of the form $x^* \times \hat{\mathcal{L}}_\kappa$, obtaining simulated proofs for them from challenger $\mathcal{B}$. By this, $\mathcal{A}$ expects that the simulated proofs contain proofs on x^* generated by O. $\mathcal{A}$ then runs the prover algorithm $\mathsf{M.Prv}$ using instance swapping, i.e., on the target $(x^*, \hat{x}^*)$ with a fake witness, say $(w^*, \hat{w}^*)$. The only way for $\mathsf{M.Prv}$ to decide whether x^* is a yes-instance or not is to make a query to O on w^*. Thus, when $\mathcal{A}$ sees such a query, it simulates O as if x^* was in the correct relation as a yes-instance with w^*. Nevertheless, $\mathcal{A}$ cannot simulate an answer to query $\mathsf{O}(\mathsf{Prv}, \sigma_j, x^*, w^*)$ with non-trivial σ_j whose trapdoor is not known to $\mathcal{A}$. For such a query, $\mathcal{A}$ must find the corresponding proof π^* that satisfies $1 \leftarrow \mathsf{O}(\mathsf{Vrf}, \sigma_j, x^*, \pi^*)$. We argue that the correct π^* is included in the view of $\mathcal{A}$ during the challenge phase where the instances including x^* have been queried to the challenger. To show that the proof that $\mathcal{A}$ created will be accepted by the legitimate verification, we follow a game transformation argument. Starting from the simulation-sound game with the above $\mathcal{A}$, we transform the game to the one where $\mathcal{A}$ runs $\mathsf{M.Prv}$ on target $(x^*, \hat{x}^*)$, which is a yes-instance relative to a modified oracle O^*. Due to the zero-knowledge property of M and the hardness of $\mathcal{L}$, such transformation is not noticeable by the verification algorithm. Accordingly, the proof will be accepted by the verification algorithm.

With the above outline in mind, we describe the simulation soundness game in the following. We consider an adversary $\mathcal{A}$ separated into three stateful polynomial-time oracle algorithms $\mathcal{A} := (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$. Similarly, the challenger $\mathcal{B}$

is separated into two algorithms $\mathcal{B} := (\mathcal{B}_0, \mathcal{B}_1)$. Let $q > 2$ be a polynomial in the security parameter that represents the maximum number of queries to O that M^{O} can make in each invocation.

[Simulation Soundness Game]

Step 1: Setup Phase.

$\mathcal{B}_0$: Generate $(\widetilde{\tau}, \widetilde{\sigma})$ by executing $(\widetilde{\tau}, \widetilde{\sigma}) \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{CrsSim}, 1^{\kappa})$.

Step 2: Challenge Phase.

$\mathcal{A}_1$: Given $\widetilde{\sigma}$ as input, sample $w^* \leftarrow \{0,1\}^{\kappa}$ and let $x^* := \mathsf{O}(\mathsf{SmplNo}, w^*)$ and $\bar{x} := \mathsf{O}(\mathsf{SmplYes}, w^*)$. Also sample $\hat{x}_1, \ldots, \hat{x}_{q^c}$ uniformly and independently from $\hat{\mathcal{L}}_{\kappa}$ and send $X := \{(x^*, \hat{x}_i)\}$ to the challenger. On receiving simulated proofs $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_{q^c}$, verify them by $b_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}_i, \widetilde{\pi}_i)$.

$\mathcal{B}_1$: On receiving $\widetilde{x}_i$, compute $\widetilde{\pi}_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{x}_i, \widetilde{\tau})$ with fresh randomness and return $\widetilde{\pi}_i$ to $\mathcal{A}_1$.

Let Q_{ver} be all successful verification queries to O of form $(\mathsf{Vrf}, [\sigma_j], x^*, [\pi_j^*])$ observed by $\mathcal{A}_1$. Let Q_{triv} be a list of common reference strings and their trapdoors, (σ_j, τ_j), that $[* \neq \bot] \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, *, \tau_j)$ or $\sigma_j \leftarrow \mathsf{O}(\mathsf{Crs}, \tau_j)$ has been observed by $\mathcal{A}_1$.

Step 3: Forgery Phase.

$\mathcal{A}_2$: Uniformly choose $\hat{x}^* \in \hat{\mathcal{L}}_{\kappa}$ and its witness $\hat{w}^*$. Apply instance swapping: define partial oracle O' with oracle entries $(\mathsf{SmplYes}, w^*, x^*)$, $(\mathsf{SmplNo}, w^*, \bar{x})$, and $(\mathsf{Prv}, *, \bar{x}, w^*, \bot)$. Run $\mathsf{M}^{\mathsf{O}''}(\mathsf{Prv}, \widetilde{\sigma}, (x^*, \hat{x}^*), (w^*, \hat{w}^*))$ where O'' is an algorithm that simulates an oracle in $\mathcal{O}$ as follows.

[Algorithm O''] Let Q_{intl} be an initially 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 $(\mathsf{Prv}, \sigma_j, x^*, w^*)$ with a valid σ_j, do as follows.
  - c1: If $(\mathsf{Vrf}, \sigma_j, x^*, [\pi_j^*])$ in Q_{ver}, then return π_j^*.
  - c2: Else if $(\sigma_j, [\tau_j]) \in Q_{\mathsf{triv}} \cup Q_{\mathsf{intl}}$, compute $\pi_j \leftarrow \mathsf{O}(\mathsf{PrvSim}, \sigma_j, x^*, \tau_j)$ and return π_j.
  - c3: Else return $\bot$.
- For every other query, forward it to O and return the received answer.

When M outputs $\widetilde{\pi}^*$, output $(x^*, \hat{x}^*)$ and $\widetilde{\pi}^*$.

Step 4: Final Verification.

$\mathcal{B}_2$: Given $(x^*, \hat{x}^*)$ and $\widetilde{\pi}^*$, output 1 if $(x^*, \hat{x}^*) \notin X$ and $(x^*, \hat{x}^*) \notin \mathcal{L}_{\kappa} \times \hat{\mathcal{L}}_{\kappa}$ and $1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, (x^*, \hat{x}^*), \widetilde{\pi}^*)$. Output 0, otherwise.

Now, we analyze the probability that $\mathcal{B}_2$ outputs 1 in the above game. We say that a common reference string σ_j is trivial if $(\sigma_j, [\tau_j])$ is included in Q_{ver}, or it is generated within the algorithm execution in consideration. Let P be probability that $\mathcal{B}_2$ outputs 1 in the simulation soundness game. It is over the choice of O and all coin flips of $\mathcal{B}$ and $\mathcal{A}$. Our goal is to show that P is non-negligible. Towards that goal, we consider a sequence of games as in the case of conjunction. Let P_i denote probability that $\mathcal{B}_2$ outputs 1 in Game i.

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

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.

This modification is exactly the same as that in Game 1 for the case of disjunction in Section 3. Applying the same analysis with adjustment of parameters, we have $|P_1 - P_0| < q(2q^c + 3)(1/2^{\kappa} + 1/2^{4\kappa})$.

Game 2: Remove the conditions $(x^*, \hat{x}^*) \notin X$ and $(x^*, \hat{x}^*) \notin \mathcal{L}_\kappa \times \hat{\mathcal{L}}_\kappa$ in the final verification.

They are satisfied unless $\hat{x}^* \in \{\hat{x}_i\}$ happens by chance, which happens with probability at most $q^c/|\hat{\mathcal{L}}_\kappa|$. Hence $|P_2 - P_1| < q^c/|\hat{\mathcal{L}}_\kappa|$.

Game 3: Halt the game if $\mathcal{A}_1$ observes $b_i = 0$. Since the challenger behaves honestly, it happens only if the simulation error occurs for input $\widetilde{x}_i \in X$. However, recall that no-instance simulation correctness is defined for instances uniformly chosen from all no-instances. On the other hand, instances $\widetilde{x}_i$ are chosen from $x^* \times \hat{\mathcal{L}}_\kappa$ in this game. Furthermore, the challenges are repeated with the same $\widetilde{\sigma}$, whereas the simulation correctness takes the choice of $\widetilde{\sigma}$ in the probability space. We will fill this gap and obtain the following bound.

Claim 4.2. $|P_3 - P_2| < \frac{1}{2} + q^c(\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa))$.

Game 4: Let $\bar{H}_x$ be H_x except for $\bar{H}_x(b||w^*) := H_x(1-b||w^*)$ for $b = 0, 1$. Let O^* be O except for using $\bar{H}_x$ instead of H_x. In this game, we use O^* instead of O in the Setup and Challenge phases. $\mathcal{A}_0$ is modified so that it chooses $x^* := \mathsf{O}(\mathsf{SmplYes}, w^*)$ and $\bar{x} := \mathsf{O}(\mathsf{SmplNo}, w^*)$.

First note that O^* is a correct oracle that belongs to $\mathcal{O}$. The use of O^* is not noticeable unless a query including w^* appear in the respective phases. Since w^* is information theoretically hidden from the view of M in these phases, it appears among at most $q(2q^c + 1)$ successful queries made during these phases with probability at most $q(2q^c + 1)/2^{\kappa}$. We thus have $|P_4 - P_3| < q(2q^c + 1)/2^{\kappa}$.

Game 5: Use O^* also in the forgery phase instead of O''.

Observe that algorithm O'' simulates O^* and the simulation is perfect unless it runs into branch c3. Let AbortA be the event that the game aborts there. Since Games 5 and 4 are the same unless AbortA happens, $|P_5 - P_4| < \Pr[\mathsf{AbortA}]$ holds. We then bound the probability of happening AbortA as follows.

Claim 4.3. $|P_5 - P_4| < \Pr[\,\mathsf{AbortA}\,] < \rho_{\mathsf{zk}}(\kappa) + \frac{q}{eq^c}$.

Game 6: Remove the challenge phase. Observe that, in Game 4, $\mathcal{A}_2$ simply computes $\widetilde{\pi}^* \leftarrow \mathsf{M}^{\mathsf{O}^*}(\mathsf{Prv}, \widetilde{\sigma}, (x^*, \hat{x}^*), (w^*, \hat{w}^*))$ in the forgery phase without using any information from the challenge phase. Therefore, this modification does not affect to the remaining phases of the game. We thus have $P_6 = P_5$.

Game 7: Modify $\mathcal{B}_0$ so that it generates $\widetilde{\sigma}$ by $\widetilde{\sigma} \leftarrow \mathsf{M}^{\mathsf{O}^*}(\mathsf{Crs}, 1^\kappa)$. In both Games 6 and 7, trapdoor $\widetilde{\tau}$ is not used. Hence it is straightforward to construct an adversary against zero-knowledgeness of M that, given $\widetilde{\sigma}$ as input, simulates the game as described and distinguishes which is the case simply by following the output of the final verification. We thus have $|P_7 - P_6| < \rho_{\mathsf{zk}}(\kappa)$.

Game 8: Use O^* also in the final verification phase instead of O.

The modification is noticed only if a query including w^* is made in the final verification phase. If, however, the verification algorithm is aware of a witness for the instance in question, the system is not zero-knowledge or the language is trivial. With a simple analysis and reduction, we can show $|P_8 - P_7| < \rho_{\mathsf{zk}}(\kappa) + q/2^\kappa$.

Finally, we observe that Game 8 is nothing but the correctness checking where a proof for a randomly chosen yes-instance is legitimately created and verified. Therefore, $P_8 > 1 - \rho_{\mathsf{co}}(\kappa)$.

By summing up the above differences of probabilities, we have

$$P > \frac{1}{2} - \frac{1}{eq^{c-1}} - q^c/|\hat{\mathcal{L}}_\kappa| - (q^c + 3)\rho_{\mathsf{zk}}(\kappa) - (q^c + 1)\rho_{\mathsf{co}}(\kappa) - \epsilon(\kappa)$$

where

$$\epsilon(\kappa) := (5q + 4q^{c+1})/2^\kappa + (3q + 2q^{c+1})/2^{4\kappa}$$

that is negligible in κ. Accordingly, if M is correct and zero-knowledge and 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 probability not negligible in κ.

Therefore, the adversary $\mathcal{A}$ breaks the simulation soundness of M over the choices of O. Finally, by applying the standard argument that there exists an oracle O in $\mathcal{O}$ according to Borel-Cantelli Lemma we complete the proof of Theorem 4.1. ■

Remark 4.1. Regarding the extended part of the language, the only property we used in the above proof is that $|\hat{\mathcal{L}}_\kappa|^{-1}$ is negligible in κ. Therefore, we can set $\hat{\mathcal{L}}_\kappa$ to be in BPP as long as the constraint on the size of the language is met. A more precise analysis about the size of $\hat{\mathcal{L}}_\kappa$ reveals that its size can be polynomial in κ as long as its inverse is sufficiently small. Setting $|\hat{\mathcal{L}}_\kappa|$ to be $\mathcal{O}(q^c)$ with a small constant factor suffices. The same is true for $\hat{\mathcal{C}}_\kappa$.

Remark 4.2. Regarding branch c3 in algorithm O'', one may think of assigning a random value to π_j and hope that it will not be verified by $\mathsf{M.Vrf}$ as we did in the case of disjunction. Then the proof might work in the full generality rather than in the full verification model as well as the previous case. Unfortunately, it does not work since $\mathsf{M.PrvSim}$ in the challenge phase may decide which proof to be verified in $\mathsf{M.Vrf}$ depending on the concrete representation of π_j. A proof π_j that remains unverified by $\mathsf{M.Vrf}$ may become verified if its representation is changed due to the random assignment.

For proofs of Claims 4.2 and 4.3, we use the following useful lemma that is a variation of the splitting lemma [19,44].

Lemma 4.1. *Let $\mathcal{X}$ and $\mathcal{Y}$ be distributions and R be a binary relation over $\mathcal{X} \times \mathcal{Y}$ that satisfies $\Pr_{x\in\mathcal{X},y\in\mathcal{Y}}[R(x,y)=1] > 1-\epsilon$ for some small ϵ. It holds that $\Pr_{y\leftarrow\mathcal{Y}}[\Pr_{x\leftarrow\mathcal{X}}[R(x,y)=1] > 1-2\epsilon] > 1/2$.*

Proof. (of Claim 4.2) Suppose that the instances in the challenge phase are uniformly chosen from $\mathcal{L}_\kappa \times \hat{\mathcal{L}}_\kappa$. According to Lemma 2.1, it holds that

$$\begin{aligned} p_1 :=& \Pr\left[\begin{array}{l} (\widetilde{\sigma},\widetilde{\tau}) \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{CrsSim}, 1^\kappa)\ ; \\ \widetilde{x} \leftarrow \mathcal{L}_\kappa \times \hat{\mathcal{L}}_\kappa\ ; \\ \widetilde{\pi} \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\tau}) \end{array} \ :\ 1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\pi})\right] \\ &> 1 - (\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa)) \end{aligned}$$

over the choice of O. Let Θ be the collection of $(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)$, where it holds that

$$p_2 := \Pr\left[\begin{array}{l} \widetilde{x} \leftarrow x^* \times \hat{\mathcal{L}}_\kappa\ ; \\ \widetilde{\pi} \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\tau}) \end{array} : 1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}, \widetilde{\pi})\right] > 1 - 2(\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa))$$

According to Lemma 4.1, a uniformly generated $(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)$ belongs to Θ with probability greater than 1/2. For a fixed $(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)$ in Θ, let ExChal be the event defined as

$$\begin{aligned} &\mathrm{ExChal}(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*) := \\ &\quad \left\{ \widetilde{\mathbf{x}} \leftarrow x^* \times \hat{\mathcal{L}}\ ;\ \widetilde{\pi} \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{\mathbf{x}}, \widetilde{\tau})\ ;\ 1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{\mathbf{x}}, \widetilde{\pi}) \right\} \end{aligned}$$

where $\widetilde{\mathbf{x}} \leftarrow x^* \times \hat{\mathcal{L}}$ is a shorthand of sampling $\widetilde{x}_i \in x^* \times \hat{\mathcal{L}}$ independently and letting $\widetilde{\mathbf{x}} := (\widetilde{x}_1, \ldots, \widetilde{x}_{q^c})$, and $\widetilde{\pi} \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{\mathbf{x}}, \widetilde{\tau})$ is a shorthand for executing $\widetilde{\pi}_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{PrvSim}, \widetilde{\sigma}, \widetilde{x}_i, \widetilde{\tau})$ for every i to obtain $\widetilde{\pi} := (\widetilde{\pi}_1, \ldots, \widetilde{\pi}_{q^c})$, and $1 \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{\mathbf{x}}, \widetilde{\pi})$ is a shorthand of running $b_i \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}_i, \widetilde{\pi}_i)$ and returning 1 if and only if $b_i = 1$ for all $i \in \{1, \ldots, q^c\}$. We then have

$$p_3 := \Pr\left[\,\mathrm{ExChal}(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)\,\right] = (p_2)^{q^c} > 1 - 2q^c(\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa)).$$

Accordingly, we have

$$p_4 := \Pr\left[\begin{array}{l} \mathsf{O} \leftarrow \mathcal{O}\ ; \\ (\widetilde{\sigma},\widetilde{\tau}) \leftarrow \mathsf{M}^{\mathsf{O}}(\mathsf{CrsSim}, 1^\kappa)\ ; \\ x^* \leftarrow \mathcal{L}_\kappa\ ; \end{array} \ :\ \mathrm{ExChal}(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)\right]$$

$$\geq \Pr[(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*) \in \Theta] \cdot \Pr[\mathrm{ExChal}(\mathsf{O}, \widetilde{\sigma}, \widetilde{\tau}, x^*)]$$
$$> \frac{1}{2} - q^c(\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa))$$

where the experiment of p_4 is the same as the setup and challenge phases in Game 3. Therefore, the probability that $\mathcal{A}_1$ aborts in Game 3 is upper-bounded by $1 - p_4 < \frac{1}{2} + q^c(\rho_{\mathsf{zk}}(\kappa) + \rho_{\mathsf{co}}(\kappa))$. ■

Proof. (of Claim 4.3) To prove the bound, we use Lemma 3.3 that states, intuitively, that an event rarely happens for the first time if the event did not happen during sufficient number of trials in the past. To argue in this way, we need to match the distribution of $\mathcal{A}$'s view in the challenge phase and the one in the forgery phase so that they belong to the same probability space. By AbortA^i we denote AbortA in the following Game i. Let Game 4.0 be Game 4.

Game 4.1: Modify $\mathcal{B}_0$ and $\mathcal{B}_1$ so that they execute $\widetilde{\sigma} \leftarrow \mathsf{M}^{\mathsf{O}^*}(\mathsf{Crs}, 1^\kappa)$ and $\widetilde{\pi}_i \leftarrow \mathsf{M}^{\mathsf{O}^*}(\mathsf{Prv}, \widetilde{\sigma}, \widetilde{x}_i, (w^*, \hat{w}_i))$, respectively, where $\hat{w}_i$ is a witness of $\hat{x}_i$ in $\widetilde{x}_i$. The trapdoor $\widetilde{\tau}$ is no longer generated but it has not been used in the previous game, either. We claim: $|\Pr[\mathsf{AbortA}^{4.1}] - \Pr[\mathsf{AbortA}^{4.0}]| < \rho_{\mathsf{zk}}(\kappa)$. To prove the claim, we construct a distinguisher, D^{O^*}, playing the non-adaptive multi-theorem zero-knowledge game for M^{O^*}. The construction is indeed straightforward once we observe that O^* is a correct oracle belonging to $\mathcal{O}$ and every $(x^*, \hat{x}_i)$ is a legitimate yes-instance with respect to O^* whose witness is $(w^*, \hat{w}_i)$. The distinguisher D^{O^*} samples $w^* \leftarrow \{0,1\}^\kappa$, and $x^* \leftarrow \mathsf{O}^*(\mathsf{SmplYes}, w^*)$, and also samples instances and witnesses as $\mathcal{A}_1$ does. It then sends the instances and witnesses to the challenger in the non-adaptive multi-theorem zero-knowledge game. Note that, the challenge instances are chosen at once, non-adaptive multi-theorem zero-knowledge suffices. On receiving $\widetilde{\sigma}$ and proofs, D^{O^*} verifies them by running $\mathsf{M}^{\mathsf{O}^*}(\mathsf{Vrf}, \widetilde{\sigma}, \widetilde{x}_i, \widetilde{\pi}_i)$ and creates Q_{ver} and Q_{triv}. It then follows the procedure of $\mathcal{A}_2$ and halts with output 1 if AbortA happens. If AbortA does not happen, D outputs 0. D^{O^*} can simulate $\mathcal{A}_2$ correctly since $\mathcal{A}_2$ does not make any query on w^* to its oracle unless event AbortA happens. Accordingly, if $\widetilde{\sigma}$ and the proofs from the challenger are the real ones, D^{O^*} has simulated the view of $\mathcal{A}_2$ in Game 4.1, otherwise, Game 4.0.

Finally, we show that $\Pr[\mathsf{AbortA}^{4.1}] < \frac{q}{eq^c}$. Observe that, until the moment that AbortA happens, the view of $\mathcal{A}_2$ is exactly the same as that of computing $\mathsf{M.Prv}$ with respect to O^*. Also observe that what the challenger in Game 4.1 is doing is computing $\mathsf{M.Prv}$ with respect to O^*. Besides, all instances are uniformly chosen from the same distribution. Now we consider the condition that AbortA happens. Event AbortA happens only if branch c1 is not the case. It means that $(\mathsf{Vrf}, \sigma_j, x^*, [\pi_j^*])$ is not observed by $\mathsf{M.Vrf}$ run by $\mathcal{A}_2$ in the challenge phase. It then means that $\mathsf{M.Prv}$ run by $\mathcal{B}$ did not make query $(\mathsf{Prv}, \sigma_j, x^*, w^*)$ since M is in the full verification model.[3] Also, AbortA happens only if branch c2 is not the case. It means that σ_j is not trivial and generated by $\mathcal{B}_0$ in the setup phase. For a specific σ_j generated by $\mathcal{B}_0$, the probability that $(\mathsf{Prv}, \sigma_j, x^*, w^*)$ appears

[3] This is where we use the condition that M is in the full verification model.

for the first time in the forgery phase after having q^c successful challenges is upper-bounded by $1/eq^c$ due to Lemma 3.3.[4] Since there could be at most q such σ_j generated in the setup phase, taking the union bound for all of them, we have the bound q/eq^c as claimed. By summing up the above bounds, we have the bound given in Claim 4.3. ■

4.2 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 $I(s) := \{i \,|\, \mathsf{f}(s)_i = 1\}$ where $\mathsf{f}(s)_i$ is the i-th bit of f(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 4.4. *(LBL-SS-NIZK ⇒ 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 $\widetilde{\sigma} := (\sigma_{11}, \sigma_{21}, \ldots, \sigma_{1n}, \sigma_{2n})$.
- Given $\widetilde{\sigma}$, $\widetilde{x} = (x_1, x_2)$, a label ℓ and a witness $\widetilde{w} = (w_1, w_2)$, $\tilde{\Pi}.\mathsf{Prv}$ chooses a random u-bitstring $r \leftarrow \{0,1\}^u$ and computes proofs $\pi_{bi} \leftarrow \Pi_b.\mathsf{Prv}(\sigma_{bi}, x_b, r, w_b)$ 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.

[4] It may worth to note that this proof strategy does not work if O.Prv supports labels since a unique random label may be included in every O.Prv query so that proofs cannot be reused. Indeed we can construct LBL-(AND)-USS-NIZK from LBL-USS-NIZK as stated in Theorem 4.4 and illustrated in Figure 1.

The simulator algorithms are constructed accordingly and zero knowledge holds immediately from $\mathit{\Pi}_1$ and $\mathit{\Pi}_2$. 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 such that $x^*_{b^*}$ is a no-instance with respective language, which exists if the above is a valid forgery for the conjunction. Now, if there exists j such that $r^* = r_j$, 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 $\mathit{\Pi}_{b^*}$ 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. ■

5 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 5.1. *(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 $\mathit{\Pi} := \{\mathsf{Crs}, \mathsf{Prv}, \mathsf{Vrf}, \mathsf{CrsSim}, \mathsf{PrvSim}\}$ 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 $\tilde{\mathit{\Pi}}$ 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 in Section 4.2.

- Given κ, $\tilde{\mathit{\Pi}}.\mathsf{Crs}$ outputs $\widetilde{\sigma} := (\sigma_1, \ldots, \sigma_n)$, where $\sigma_i \leftarrow \mathit{\Pi}.\mathsf{Crs}(1^\kappa)$.
- Given $\widetilde{\sigma}$, instance $x \in \mathcal{L}$, a label ℓ and a witness w, $\tilde{\mathit{\Pi}}.\mathsf{Prv}$ 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 $\mathit{\Pi}.\mathsf{Prv}(\sigma_i, (x, \hat{x}), (w, \hat{w}))$ for $i \in I(\ell)$.
- The verification algorithm $\tilde{\mathit{\Pi}}.\mathsf{Vrf}$ 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.

The simulators are constructed accordingly and the zero knowledge property of $\tilde{\mathit{\Pi}}$ is inherited from the one by $\mathit{\Pi}$. For unbound 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 $\mathit{\Pi}$. Otherwise, observe that if $\widetilde{\pi}^* = (\hat{x}^*, \{\widetilde{\pi}^*_i\}_{i \in I(\ell^*)})$ is valid, the USS of $\mathit{\Pi}$ is compromised, because there must exist an index i such that $(x^*, \hat{x}^*)$ has not been proven with respect to σ_i, and however, $\widetilde{\pi}^*_i$ 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 $\hat{\mathcal{L}}$. ■

The following result is obtained from Theorem 5.1 and Theorem 4.4.

Corollary 5.2. AND-USS-NIZK $\Rightarrow$ LBL-AND-USS-NIZK

When $\hat{\mathcal{L}}$ is a *hard language* we can build a reduction from OR-USS-NIZK to LBL-USS-NIZK.

Theorem 5.3. *(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 5.1. 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. ∎

Finally, the previous results lead to the following.

Corollary 5.4. NIZK $\not\Rightarrow_*$ USS-NIZK in the full verification model.

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. ∎

5.1 On the full verification model

Some of our impossibility results (marked with $*$) are in the *full verification model* (FVM), i.e., we focus on a restricted class of black-box constructions: those where all the internal proofs generated by the *prover* algorithm must be verified by the *verification* algorithm (see Definition 4.1). That is a technical requirement used in the proof of Theorem 4.1 and also affects the results derived from it. In this section we argue that the full verification model has a little impact in the conclusion that can be derived from our results in terms of generality.

First, the model covers a considerably wide variety of constructions that use the given oracle as a proof system. Second, constructions excluded by the model must exploit unverified proofs in some meaningful manner, which we believe is not possible.

In particular, we informally argue that for every black-box construction (not in the full verification model) there exists an equivalent black-box construction that is in the FVM. First of all, note that if all internal proofs are included in the final proof, but not all are verified, we can just modify the verification algorithm to verify every single proof (and abort if some of them does not pass the internal verification), without compromising zero-knowledge nor soundness. Furthermore, completeness would not be compromised either because every proof has been honestly generated. Now, consider a black-box construction whose prover algorithm calls the underlying prover, producing a set of proofs, and suppose that only a subset of them are included in the final proof. The non-included proofs may be used to decide (in an arbitrary manner) how the subsequent proofs are created. We claim that, if the black-box construction implements a secure NIZK system, the probability distribution induced by the internal proofs (that are not included) can be simulated without actually producing such proofs. That would imply that there exists an equivalent black-box construction which includes every internal proof in the final proof and (as explained above) there would exist an equivalent black-box construction in the FVM.

The key point of the previous argument is that the black-box construction must work for *every possible underlying NIZK system.* To illustrate this idea, consider a black-box construction such that it includes in the final proof only the internal proofs that start with a 0 in their binary representation. Observe that such a construction cannot implement a valid NIZK in a black-box manner, because if the underlying NIZK is such that the binary representation of its proofs starts with 1, the prover of the main construction will not produce an output.

To conclude, we would like to stress that our impossibility results, even in the full verification model, should be valid as a strong motivation to look for non-black-box approaches to the corresponding problems.

6 Signatures from USS-NIZK w/o Language Extension

6.1 A Simple Case

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). A formal statement follows.

Theorem 6.1. *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 $\mathcal{C}'_\kappa \subseteq \mathcal{C}_\kappa$.*

Proof. Correctness is immediate from the perfect no-instance correctness of Π and the fact that $\mathcal{H}$ is an injection. For EUF-CMA security, a straightforward reduction works. We construct an adversary $\mathcal{A}$ playing the simulation soundness game for Π using an adversary $\mathcal{B}$ playing the EUF-CMA game for Σ as a black-box. Given a σ, $\mathcal{A}$ selects $K \leftarrow \mathcal{K}_\kappa$ and invokes $\mathcal{B}$ with (σ, K) as a public-key. When $\mathcal{B}$ makes a signing query $m \in \mathcal{M}_\kappa$, $\mathcal{A}$ computes $x \leftarrow H_\kappa(K, m)$ and sends x to its proof oracle. On receiving a simulated proof π, $\mathcal{A}$ returns $\sigma := \pi$ to $\mathcal{B}$. If $\mathcal{B}$ finally outputs a forgery (σ^*, m^*), $\mathcal{A}$ outputs $(\pi^*, x^*) := (\sigma^*, H_\kappa(K, m^*))$. Due to the perfect no-instance correctness and the fact that H_κ is injective, the simulated proofs given to $\mathcal{B}$ pass the verification as signatures. Thus, $\mathcal{B}$ outputs a valid forgery with sufficiently high probability as assumed. Since the fresh message m^* is mapped to a fresh no-instance x^* due to the injection H_κ, the final output from $\mathcal{A}$ is a valid forgery. ∎

Note that no hardness of $\mathcal{L}$ is required for the proof. It is only required that $\mathcal{L}_\kappa$ and $\mathcal{C}_\kappa$ be disjoint. Also note that the construction allows several variations and relaxations: H_κ can be relaxed to a collision-resistant hash function family, and Π can be a *designated prover* simulation sound *quasi-adaptive* NIZK as shown in [3] where the authors develop an efficient structure-preserving signature scheme as a concrete application of this result.

6.2 General Case

We now address a more general case where the underlying USS-NIZK is not perfectly no-instance simulation correct. We 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 6.1. *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*

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

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.

We next establish some lemmas with respect to eTCR and NIZK simulation functions. We begin by introducing a result about the distribution of the outputs from eTCR functions. Ideally, we would like to have the property that the no-instances generated from messages are indistinguishable from yes-instances. That could be achieved if our function that produces no-instances from messages would distribute uniformly over the range of $\mathcal{D}_{\mathcal{C}}$.

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 6.2. *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,*

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

The following lemma allows us to argue that the distribution of no-instances produced from messages (think of messages as $m(\kappa)$-bitstrings in this case) is indistinguishable from yes-instances (see Definition 2.1).

Lemma 6.1. *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[5] $\Delta(F(X), F(Y)) \leq \Delta(X,Y)$.

[5] 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

In our case, for every $m \in \mathcal{M}_\kappa$ and every sufficiently large κ,

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

We expect that distribution $\mathcal{D}_{\mathcal{C}_\kappa}$ is close to injection and hence has a small collision probability as defined in the following. It implies that instances in $\mathcal{C}$ has a short witness.

Definition 6.3 (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*

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

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.

Theorem 6.2 (Correctness). *The signature scheme Σ defined above is correct. Concretely, for every message $m \in \mathcal{M}_\kappa$ and for every sufficiently large κ,*

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

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

$$\Pr\left[\, (pk, sk) \leftarrow \Sigma.\mathsf{Setup}(1^\kappa) \text{ ; } \sigma \leftarrow \Sigma.\mathsf{Sign}(pk, sk, m) : 1 = \Sigma.\mathsf{Vrf}(pk, m, \sigma) \,\right]$$

is equal to

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

which, by Lemma 6.1 (and for every sufficiently large κ) is greater or equal than

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

which, by Lemma 2.2, is greater than

$$1 - \epsilon_{\mathsf{zk}}(\kappa) - \epsilon_{\mathsf{co}}(\kappa) - \epsilon_{\mathsf{hd}}(\kappa) - 2\epsilon_{\mathsf{reg}}(\kappa) \ . \qquad \blacksquare$$

Theorem 6.3 (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. In fact, $\mathcal{A}$ cannot determine whether it is interacting with $\mathcal{B}$ or the *genuine signer*. Observe that some signing queries can result into invalid signatures (although only with negligible probability), i.e., for some indices i, it is possible to have $\Pi.\mathsf{Vrf}(\sigma,y_i,\pi_i) = 0$. However, this is not a problem of the simulation, 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, $\Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{B} \text{ wins}] + \Pr[\mathsf{Bad}]$. We will show an upper bound for $\Pr[\mathsf{Bad}]$. 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):

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

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 $\max\limits_{p.p.t.\ M}\{\Pr[E_M]\} \leq \max\limits_{p.p.t.\ \bar{M}}\{\Pr[\bar{E}_{\bar{M}}]\}$ where the probability of event $\bar{E}_{\bar{M}}$ for a fixed algorithm $\bar{M}$ is defined as

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

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

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

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

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

■

7 Concluding Remarks and Open Questions

On the limitations of our impossibility result. The restriction on the class of black-box constructions that we consider seems inherent to our proof strategy as our adversary does not make a guess on the internal queries visible only to the challenger. Like the unconditional separation about succinct non-interactive arguments presented in [21], the meta-reduction paradigm [8, 13] may help for eliminating our restriction. Our proof of separation essentially relies on the model of simulation soundness, where the challenger accepts no-instances. Hence, it is of a theoretical interest to study whether true-simulation soundness [15] can be separated or not.

Missing relations. We leave as an open problem the study of the relations that have not been considered in Figure 1 (the missing arrows). In particular, disjunctive language extensions could have a tremendous impact in many applications and we believe that developing methods (or arguing their inexistence) for upgrading OR-NIZKs to any kind of USS-NIZK in a black-box manner is an appealing target for future work.

Acknowledgments

We thank Carsten Baum and Kyosuke Yamashita for their valuable comments. We also thank Arnab Roy for his suggestion on labelled NIZKs.

References

1. M. Abdalla, F. Benhamouda, and D. Pointcheval. Disjunctions for hash proof systems: New constructions and applications. *EUROCRYPT 2015*, *LNCS* 9057, pp.69–100. Springer.
2. 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.
3. M. Abe, C.S. Jutla, M. Ohkubo, and A. Roy. Improved (almost) tightly-secure simulation-sound QA-NIZK with applications. In *ASIACRYPT 2018*, 2018.
4. B. Barak and M. Mahmoody-Ghidary. Lower bounds on signatures from symmetric primitives. In *FOCS 2007*, pp.680–688. IEEE Computer Society.
5. 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.
6. M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). *STOC 1988*, pp.103–112. ACM, 1988.
7. 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.
8. D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. *EUROCRYPT 1998*, *LNCS* 1403, pp.59–71. Springer.

9. 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.
10. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS1993*, pp. 62–73. ACM.
11. 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.
12. 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. Springer.
13. J.S. Coron. Optimal security proofs for PSS and other signature schemes. *EUROCRYPT 2002*, *LNCS* 2332, pp.272–287. Springer.
14. 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.
15. Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. *ASIACRYPT 2010*, *LNCS* 6477, pp.613–631. Springer.
16. A. De Santis, S. Micali, G. Persiano. Non-Interactive Zero-Knowledge with Preprocessing. *CRYPTO 1988*, *LNCS* 403, pp.269–282. Springer.
17. S. Even, A.L. Selman, Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.
18. 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.
19. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. *CRYPTO 1986*, *LNCS* 263, pp.186–194. Springer.
20. S. Garg and D. Gupta. Efficient round optimal blind signatures. *EUROCRYPT 2014*, *LNCS* 8441, pp. 477–495, Springer.
21. C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. *STOC 2011*, pp.99–108. ACM, 2011.
22. 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.
23. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. *ASIACRYPT 2006*, *LNCS* 4284, pp.444–459. Springer.
24. J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. *EUROCRYPT 2006*, *LNCS* 4004, pp. 339–358.
25. Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *J. ACM*, 59(3):11:1–11:35, 2012.
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.Springer.
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. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *J. Cryptology*, 13(3):361–396, 2000.
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. O. Reingold, L. Trevisan, and S.P. Vadhan. Notions of reducibility between cryptographic primitives. In *TCC 2004*, *LNCS* 2951, pp.1–20. Springer.
48. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS 1999*, pp.543–553. IEEE Computer Society.
49. A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003.
50. 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.
51. D.R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? *EUROCRYPT 1998*, *LNCS* 1403, pp.334–345.
52. E. Vahlis. *Cryptography: Leakage Resilience, Black Box Separations, and Credential-Free Key Exchange.* PhD thesis, Graduate Department of Computer Science, University of Toronto, 2010.
53. 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.

A Proofs of the Main Body

A.1 Proofs of Section 2

Proof of Lemma 2.1. Let R be a relation associated to $\mathcal{L}_\kappa$. For every κ, fix (x, w) that satisfies $R(x, w) = 1$, and consider a p.p.t. algorithm $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1$ takes σ as input and outputs (x, ℓ, w, st) where $st = \sigma$, and $\mathcal{A}_2$ takes st and π as input and outputs 1 if and only if $1 \neq \mathsf{Vrf}(\sigma, x, \ell, \pi)$. Since $\mathcal{A}$ is a p.p.t. algorithm and Π is zero-knowledge, for every sufficiently large κ, we have

$$\left|\Pr\left[\begin{array}{r}(\sigma,\tau)\leftarrow \mathsf{CrsSim}(1^\kappa)\,;\ (x,\ell,w,st)\leftarrow\mathcal{A}_1(\sigma)\,;\\ \pi\leftarrow\mathsf{PrvSim}(\sigma,x,\ell,\tau)\end{array}\ :\ 1\leftarrow\mathcal{A}_2(st,\pi)\right]\right.$$
$$\left.-\Pr\left[\begin{array}{r}\sigma\leftarrow \mathsf{Crs}(1^\kappa)\,;\ (x,\ell,w,st)\leftarrow\mathcal{A}_1(\sigma)\,;\\ \pi\leftarrow\mathsf{Prv}(\sigma,x,\ell,w)\end{array}\ :\ 1\leftarrow\mathcal{A}_2(st,\pi)\right]\right| < \epsilon_{\mathsf{zk}}(\kappa)\ .$$

Now, correctness of Π implies that

$$\begin{aligned}&\Pr[\sigma\leftarrow\mathsf{Crs}(1^\kappa); (x,\ell,w,st)\leftarrow\mathcal{A}_1(\sigma); \pi\leftarrow\mathsf{Prv}(\sigma,x,\ell,w) : 1\leftarrow\mathcal{A}_2(st,\pi)]\\ &\qquad = \Pr[\sigma\leftarrow\mathsf{Crs}(1^\kappa); \pi\leftarrow\mathsf{Prv}(\sigma,x,\ell,w) : 1\neq\mathsf{Vrf}(\sigma,x,\ell,\pi)] < \epsilon_{\mathsf{co}}(\kappa)\ .\end{aligned}$$

We thus have

$$\begin{aligned}&\Pr\left[\,(\sigma,\tau)\leftarrow\mathsf{CrsSim}(1^\kappa)\ :\ 1\neq\mathsf{Vrf}(\sigma,x,\mathsf{PrvSim}(\sigma,x,\ell,\tau))\,\right]\\ &\quad=\Pr\left[\begin{array}{r}(\sigma,\tau)\leftarrow\mathsf{CrsSim}(1^\kappa)\,;\quad(x,\ell,w,st)\leftarrow\mathcal{A}_1(\sigma)\\ \pi\leftarrow\mathsf{PrvSim}(\sigma,x,\ell,\tau)\end{array}\ :\ 1\leftarrow\mathcal{A}_2(st,\pi)\right]\\ &\quad<\epsilon_{\mathsf{zk}}(\kappa)+\Pr\left[\begin{array}{r}\sigma\leftarrow\mathsf{Crs}(1^\kappa)\,;\ (x,\ell,w,st)\leftarrow\mathcal{A}_1(\sigma)\\ \pi\leftarrow\mathsf{Prv}(\sigma,x,\ell,w)\end{array}\ :\ 1\leftarrow\mathcal{A}_2(st,\pi)\right]\end{aligned}$$

which is lower than $\epsilon_{\mathsf{zk}}(\kappa) + \epsilon_{\mathsf{co}}(\kappa)$ for every $x \in \mathcal{L}_\kappa$. ■

Proof of Lemma 2.2. For an arbitrary label $\ell \in \{0,1\}^{poly_\ell(\kappa)}$, consider the following algorithm that on input $x \in \mathcal{L}_\kappa \cup \mathcal{C}_\kappa$ outputs a bit:

$$\mathcal{A}(x):\qquad (\sigma,\tau)\leftarrow\mathsf{CrsSim}(1^\kappa);\quad \pi\leftarrow\mathsf{PrvSim}(\sigma,x,\ell,\tau);\quad \mathsf{return}\,\mathsf{Vrf}(\sigma,x,\ell,\pi)\ .$$

We may think of $\mathcal{A}$ as a distinguisher for the hardness of the language $\mathcal{L}_\kappa$ (with respect to $\mathcal{C}_\kappa$). Since $\mathcal{A}$ is a p.p.t. algorithm, the hardness of $\mathcal{L}_\kappa$ ensures that $\Pr[x\leftarrow\mathcal{D}_{\mathcal{L}_\kappa} : 1\leftarrow\mathcal{A}(x)] - \Pr[x\leftarrow\mathcal{D}_{\mathcal{C}_\kappa} : 1\leftarrow\mathcal{A}(x)] < \epsilon_{\mathsf{hd}}(\kappa)$. Therefore,

$$\begin{aligned}&\Pr\left[\,(\sigma,\tau)\leftarrow\mathsf{CrsSim}(1^\kappa)\,;\ x\leftarrow\mathcal{D}_{\mathcal{C}_\kappa}\ :\ 1=\mathsf{Vrf}(\sigma,x,\mathsf{PrvSim}(\sigma,x,\ell,\tau))\,\right]\\ &\quad=\Pr\left[\,x\leftarrow\mathcal{D}_{\mathcal{C}_\kappa}\ :\ 1\leftarrow\mathcal{A}(x)\,\right]\\ &\quad>\Pr\left[\,x\leftarrow\mathcal{D}_{\mathcal{L}_\kappa}\ :\ 1\leftarrow\mathcal{A}(x)\,\right]-\epsilon_{\mathsf{hd}}(\kappa)\\ &\quad=\left(\textstyle\sum_{x\in\mathcal{L}_\kappa}\Pr[1\leftarrow\mathcal{A}(x)]\cdot\Pr\left[\,y\leftarrow\mathcal{D}_{\mathcal{L}_\kappa}\ :\ y=x\,\right]\right)-\epsilon_{\mathsf{hd}}(\kappa)\quad\text{(by Lemma 2.1)}\\ &\quad\geq(1-\epsilon_{\mathsf{zk}}(\kappa)-\epsilon_{\mathsf{co}}(\kappa))\left(\textstyle\sum_{x\in\mathcal{L}_\kappa}\Pr\left[\,y\leftarrow\mathcal{D}_{\mathcal{L}_\kappa}\ :\ y=x\,\right]\right)-\epsilon_{\mathsf{hd}}(\kappa)\\ &\quad=1-\epsilon_{\mathsf{zk}}(\kappa)-\epsilon_{\mathsf{co}}(\kappa)-\epsilon_{\mathsf{hd}}(\kappa)\ .\end{aligned}$$

■

A.2 Proofs of Section 3

Proof of Lemma 3.1. The L^{O} satisfies:

- *Hard Promise Problem:* Observe that $\mathcal{L}_\kappa$ and $\mathcal{C}_\kappa$ are disjoint because H_x is injective and their witnesses are disjoint due to the unique 1-bit prefix. Over the choice of H_x, the instances of $\mathcal{L}_\kappa \cup \mathcal{C}_\kappa$ distribute uniformly over $\{0,1\}^{2\kappa}$, therefore, they are perfectly indistinguishable. Note that it is true even in the presence of other functions in O.
- *Correctness and Simulation Correctness:* L^{O} as a NIZK is perfectly correct because, for any σ generated as $\sigma \leftarrow \mathsf{O}(\mathsf{Crs}, \tau)$ and any yes-instance $x \leftarrow \mathsf{O}(\mathsf{SmplYes}, w)$, it holds that $\perp \not\leftarrow H_c(\tau)$, $x = H_x(1||w)$ and $(\sigma||x||\ell) = H_p^{-1}(H_p(\sigma||x||\ell))$. Thus, the verification always outputs 1. It is perfectly simulation correct for both yes and no-instances because, for any crs-trapdoor pair satisfying $\sigma = H_c(\tau)$ and any instance x satisfying $\perp \neq H_x^{-1}(x)$, $\mathsf{L}^{\mathsf{O}}.\mathsf{PrvSim}$ computes a proof $\pi \leftarrow H_p(\sigma||x||\ell)$ that always passes the verification $(\sigma||x||\ell) = H_p^{-1}(\pi)$.
- *Adaptive Zero-knowledge:* Observe that $\mathsf{L}^{\mathsf{O}}.\mathsf{Crs}$ and $\mathsf{L}^{\mathsf{O}}.\mathsf{CrsSim}$ differ only in their interface and actually compute the same $\mathsf{O}(\mathsf{Crs}, 1^\kappa)$. Additionally, observe that the proof created by $\mathsf{O}(\mathsf{Prv}, \sigma, x, \ell, w)$ is equal to the one created by $\mathsf{O}(\mathsf{PrvSim}, \sigma, x, \ell, \tau)$ for any σ and x that are sampled correctly by $\mathsf{O}.\mathsf{Crs}$ and $\mathsf{O}.\mathsf{SmplYes}$. Therefore, the system is same-string perfect adaptive zero-knowledge.
- *Simulation Soundness:* Let $\mathcal{A}$ be an adversary that is given access to O and *wins* the simulation soundness game with a challenger. Namely, $\mathcal{A}^{\mathsf{O}}$ is given σ generated by $(\sigma, \tau) \leftarrow \mathsf{L}^{\mathsf{O}}(\mathsf{CrsSim}, 1^\kappa)$ for certain τ chosen uniformly at random from $\{0,1\}^\kappa$. $\mathcal{A}$ is then allowed to send arbitrary instances x to the challenger and receives the corresponding simulated proofs π generated by $\pi \leftarrow \mathsf{L}^{\mathsf{O}}(\mathsf{PrvSim}, \sigma, x, \ell, \tau)$. After making such queries up to q_1 times, $\mathcal{A}$ eventually outputs (x^*, ℓ^*, π^*) that satisfies $x = \mathsf{O}(\mathsf{SmplNo}, w)$ for some $w \in \{0,1\}^\kappa$ and $1 \leftarrow \mathsf{O}(\mathsf{Vrf}, \sigma, x^*, \ell^*, \pi^*)$, and x^* has never been queried to the challenger. Let q_2 be the number of queries from $\mathcal{A}$ to O. Since (x^*, ℓ^*, π^*) passes the verification, it satisfies $\pi^* = H_p(\sigma||x^*||\ell^*)$. Observe that x^* has never appeared in the conversations with the challenger. Hence, H_p has never been evaluated on $(\sigma||x^*||\ell^*)$ by the challenger. The same is true with respect to the process of generating σ. Thus, $H_p(\sigma||x^*||\ell^*)$ could have been evaluated directly by interacting with $\mathsf{O}.\mathsf{Prv}$ or $\mathsf{O}.\mathsf{PrvSim}$. However, calling $\mathsf{O}.\mathsf{Prv}$ is not useful as it returns $\perp$ for any no-instance x^*. On the other hand, calling $\mathsf{O}.\mathsf{PrvSim}$ on σ requires the correct trapdoor τ, which can only be identified by making queries to $\mathsf{O}.\mathsf{Crs}$ or $\mathsf{O}.\mathsf{PrvSim}$ on a random candidate τ'. (Making a query $\mathsf{O}(\mathsf{PrvSim}, \sigma, x, \ell, \tau')$ with a valid instance x results in a valid proof if and only if τ' is the correct one.) Now, after q_2 queries to O, such a guess succeeds with, at most, probability $q_2/2^\kappa$. If $H_p(\sigma||x^*||\ell^*)$ has never been evaluated, the final verification will be successful only if $H_p^{-1}(\pi^*)$ has already been evaluated. Since H_p^{-1} is evaluated only inside $\mathsf{O}.\mathsf{Vrf}$, the adversary $\mathcal{A}$ (who must make queries to $\mathsf{O}.\mathsf{Vrf}$ on fresh random candidates

of π^*) can trigger the event $(\sigma||x^*||\ell^*) = H_p^{-1}(\pi^*)$ with probability lower than $q_2/(2^{6\kappa} - q_1 - (q_2 - 1)) < 2q_2/2^{6\kappa}$ for $q_1, q_2 \ll 2^{6\kappa}$. Therefore, the probability that $\mathcal{A}$ wins the simulation soundness game is upper-bounded by $q_2/2^{\kappa} + 2q_2/2^{6\kappa}$, which is negligible in κ. ■