Dual-Mode NIZKs: Possibility and Impossibility Results for Property Transfer

Vivek Arte[1] Mihir Bellare[2]

May 2020

Abstract

This paper formulates, and studies, the problem of property transference in dual-mode NIZKs. We say that a property P (such as soundness, ZK or WI) transfers, if, one of the modes having P allows us to prove that the other mode has the computational analogue of P, as a consequence of nothing but the indistinguishability of the CRSs in the two modes. Our most interesting finding is negative; we show by counter-example that the form of soundness that seems most important for applications fails to transfer. On the positive side, we develop a general framework that allows us to show that zero knowledge, witness indistinguishability, extractability and weaker forms of soundness do transfer. Our treatment covers conventional, designated-verifier and designated-prover NIZKs in a unified way.

[1] Department of Computer Science & Engineering, University of California San Diego, USA. Email: `varte@eng.ucsd.edu`. URL: `cseweb.ucsd.edu/~varte/`.

[2] Department of Computer Science & Engineering, University of California San Diego, USA. Email: `mihir@eng.ucsd.edu`. URL: `cseweb.ucsd.edu/~mihir/`. Supported in part by NSF grant CNS-1717640 and a gift from Microsoft corporation.

Contents

1 Introduction

Non-interactive zero-knowledge (NIZK) systems [BFM88, BDSMP91] are blossoming. New applications are fueling the development of schemes that are not only more efficient than classical ones but may also be simpler, more elegant and more powerful in application. One way this happens is via the concept of dual mode [GOS06b, GOS06a, GOS12, AFH+16, HU19, LPWW20].

<u>Background.</u> Groth, Ostrovsky and Sahai [GOS06b, GOS06a, GOS12] build a pair of NIZK systems Π_0, Π_1 such that (1) the prover and verifier are the same for both (2) the two systems have CRSs that, although different, are computationally indistinguishable under the SubGroup Decision or Decision Linear Assumptions, and (3) Π_0 has perfect soundness while Π_1 has perfect zero-knowledge (ZK).

Why do this? GOS wanted a (single) NIZK system that was perfect ZK and computationally sound. The claim would be that Π_1 has these properties, because CRS indistinguishability, plus the perfect soundness of Π_0, would automatically imply computational soundness of Π_1. In our language, soundness has been "transferred" from Π_0 to Π_1, as a consequence of nothing but CRS indistinguishability.

Recognition of the power of this technique lead to the formalization of dual-mode systems. Rather than a single or accepted definition, however, there are many, with a common core and varying peripherals [AFH+16, HU19, LPWW20]. At its core, a dual-mode system $\mathsf{D}\Pi$ has a CRS generator that takes an input $\mu \in \{0, 1\}$ (the desired mode), and the CRSs generated in the two modes must be computationally indistinguishable. Proving and verification algorithms are as in a (single mode) NIZK system. The modes are called binding and hiding, and the (varying) peripheral requirements placed on them in prior works are summarized in Figure 1.

Looking across usage and applications in the literature, the value of dual-mode continues to lie in transference, namely, being able to prove that if a property (like soundness) is present in one mode then its computational analogue is also present in the other, as a consequence just of the indistinguishability of the CRSs in the two modes. But there is growing recognition that transference can be subtle and not as simple as it seems. For example, certain (weaker) forms of soundness are proven to transfer, but for other (stronger) forms, the natural proof approach fails, and whether or not the transference holds remains an open question [GOS12, LPWW20]. This lead GOS, in the journal version of their work [GOS12], to introduce culpable soundness, an intermediate notion that they could show transfers.

<u>Overview of our contributions.</u> We want to understand the limitations and possibilities of transference, including which properties transfer, which don't, and why. We divide our contributions into the following parts.

▷ **Definitions.** We start by defining a dual-mode system $\mathsf{D}\Pi$ in a different way; we ask *only* for the (core) CRS indistinguishability requirement. Unlike prior works (cf. Figure 1), no requirements are placed on the individual modes. We then define, in the natural way, the (single-mode) systems $\mathsf{D}\Pi_0, \mathsf{D}\Pi_1$ induced by $\mathsf{D}\Pi$. A property P (soudness, ZK, WI, ...) is a requirement on a single-mode system, not a dual-mode one. Transference of a property P is now the question: If $\mathsf{D}\Pi_\mu$ satisfies P, does $\mathsf{D}\Pi_{1-\mu}$ satisfy the computational analogue of P?

▷ **Negative results.** We show that certain strong (desirable, application-enabling) forms of soundness P fail to transfer. These negative results are established by giving explicit counter-examples under standard assumptions (CDH, DDH). This shows that the difficulties noted in prior work with regard to proving transference for certain forms of soundness [GOS12, LPWW20], are inherent.

▷ **Positive results.** We formalize a "property" P as a *property specification* PS and give

Who	**Requirements for mode 0**	**Requirements for mode 1**
[AFH$^+$16]	perfect soundness and extractability	perfect ZK and WI
[HU19]	statistical soundness and extractability	statistical WI
[LPWW20]	statistical soundness	statistical ZK
Our work	**None**	**None**

Figure 1: Requirements placed on the two modes in definitions of dual-mode systems.

sufficient conditions on PS for it to successfully transfer. Through this we show that ZK, WI and extractability (we define them in strong ways that are application-enabling) do transfer, as do weaker (as per what follows, not application-enabling) forms of soundness. Our framework may have future value in helping evaluate or establish transference of the many definitional variants of the basic properties that arise.

$\triangleright$ **Applicability assessment.** It is desirable that the forms of soundness that transfer be suitable for applications, leading us to ask, which are? We examine a canonical application of NIZKs, the one to digital signatures [BG90, CDG$^+$17], and find that the form of soundness it needs is one that our negative results show does not transfer. Thus, our finding is that the most application-enabling (stronger, desirable) forms of soundness fail to transfer, while weaker forms do transfer.

$\triangleright$ **Unified treatment.** We define single and dual-mode systems in a way that includes, as special cases, the conventional [BFM88, BDSMP91], designated verifier [ES02, PsV06, DFN06] and designated prover [KW18, KNYY19] settings. Our definition of CRS indistinguishability asks that it hold even when the adversary knows the proving and verification keys (if any). Soundness is required even in the presence of a verification oracle, to capture the reusable designated verifier setting [LPWW20]. Our results apply to all these settings. The motivation for this broad treatment is the many recent advances in settings beyond the conventional one [KW18, KNYY19, LPWW20, BCGI18, BCG$^+$19]. In the rest of this Introduction, however, we will for simplicity confine discussion to the conventional setting.

<u>Transference intuition.</u> One would imagine that any property P transfers, by the following proof. Suppose P holds for one mode, wlog $\mathsf{D\Pi}_0$. We want to show it also holds in $\mathsf{D\Pi}_1$. Let A be a PT (polynomial-time) adversary violating P in $\mathsf{D\Pi}_1$. We build a PT adversary D violating mode indistinguishability. As per the definition of the game for the latter, the input to D is a crs of a challenge mode $\mu \leftarrow\$ \{0,1\}$, and D is trying to determine μ. Adversary D runs A on input crs and *tests if it violates* P. If so, it predicts that $\mu = 1$, else that $\mu = 0$. The difficulty is that "testing whether A violates P" may not be doable in polynomial-time. In particular, for soundness (depending on the precise definition of the property as we consider below), it may involve testing membership in the underlying language, which, for languages of interest, is not doable in polynomial-time. This difficulty is recognized [GOS12, LPWW20]. However, it does not mean that the property necessarily fails to transfer; it just means that the obvious proof approach fails. Is there another, more clever one, that succeeds? We will answer this question negatively, giving counterexamples to show non-transference for certain properties of interest in applications.

<u>Soundness notions.</u> The underlying **NP**-relation R defines a language $\mathsf{L_R}(\mathrm{crs})$. (As per [Gro06], and to cover systems in the literature, the language is allowed to depend on the CRS.) Soundness of a (single-mode) proof system $\mathsf{\Pi}$ for R asks that an adversary given crs be unable to find an $x \notin \mathsf{L_R}(\mathrm{crs})$, and a proof pf, such that $\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, x, \mathrm{pf}) = \mathsf{true}$. The difficulty, for transference, is that testing whether the adversary wins seems to require testing that $x \notin \mathsf{L_R}(\mathrm{crs})$, which is likely not doable in PT. With attention drawn to this issue, however, one sees that whether or not the

test is required depends on exactly how the soundness game is defined. The broad format is that the game picks and gives crs to the adversary, who then provides the game with x, pf, and the game then performs a "winning test." Now we consider two definitions: SND-P (penalty style) and SND-E (exclusion style). (This follows the consideration of similar notions for IND-CCA encryption in [BHK15].) In SND-P, the winning test is that $x \notin \mathsf{L}_{\mathsf{R}}(\mathrm{crs})$ and $\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, x, \mathrm{pf}) = \mathsf{true}$, and SND-P security asks that any PT adversary has negligible winning probability. In SND-E the winning test is just that $\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, x, \mathrm{pf}) = \mathsf{true}$, and SND-E security asks for negligible winning probability, not for all adversaries, but for a subclass we call membership conscious: the probability that the x they provide is in $\mathsf{L}_{\mathsf{R}}(\mathrm{crs})$ is negligible. (Membership consciousness is an assumption on the adversary. Nothing in the game verifies it.) Clearly, SND-P is stronger: SND-P $\Rightarrow$ SND-E. (Any SND-P secure Π is also SND-E secure.) We can show that SND-E is strictly weaker: SND-E $\not\Rightarrow$ SND-P. (There exists a Π that is SND-E secure but not SND-P secure.)

<u>Soundness transference.</u> We show that (1) SND-E transfers, but (2) SND-P does not. The first follows from general results we discuss below. With regard to the second, that the winning test is not PT is an indication that transfer may fail, but not a proof that it does. (What it means is that the particular, above-discussed approach to prove transference fails.) In Theorem 4.4, we show non-transference via a counter-example. We give a dual-mode proof system $\mathsf{D\Pi}$ and relation R such that (2a) $\mathsf{D\Pi}$ satisfies mode indistinguishability (2b) $\mathsf{D\Pi}_1$ satisfies SND-P for R but (2c) $\mathsf{D\Pi}_0$ does *not* satisfy SND-P for R. These results assume hardness of the DDH (Decision Diffie-Hellman) problem in an underlying group. We show (2a) and (2b) by reductions to the assumed hardness of DDH. We show (2c) by an attack, a description of an explicit PT adversary that, with probability one, violates SND-P for $\mathsf{D\Pi}_0$.

<u>Penalty or exclusion?</u> The lack of transference of SND-P is more than an intellectual curiosity; it inhibits applicability. We consider building digital signatures from NIZKs, a canonical application that originates in [BG90], and, with [CDG$^+$17], is seeing renewed interest as a way to obtain efficient post-quantum signatures. We look closely at the proof to see that while SND-P suffices, SND-E does not appear to do so. This phenomenon seems fairly general: applications of NIZKs that rely on soundness seem to need SND-P, not SND-E.

<u>Transference framework.</u> We turn to positive results, proving that certain (important) properties P *do* transfer. We could give such proofs individually for different choices P, but this has a few drawbacks. First, proofs which at heart are very similar will be repeated. Second, proofs will be needed again for new or further property variants that we do not consider. Most important, however, from our perspective, ad hoc proofs fail to yield theoretical understanding; exactly what about a property allows it to transfer?

To address this we give a framework to obtain positive transference results. The intent is to formalize the above-described transference intuition. We start with a definition, of an object, denoted PS, that we call a *property specification* PS. While the game defining P would typically pick crs by running $\Pi.\mathsf{C}$, the corresponding property specification sees the game output (result, win or not, of executing the game with an adversary) as a function of crs, effectively pulling the latter out of the game. We then give a general result (Theorem 5.2, the Transfer Theorem) saying that property specifications transfer successfully *as long as they are polynomial time.*

To apply this to show transference of a particular property P, we must specify the corresponding property specification PS and show that it is polynomial time. We do it for SND-E, zero-knowledge, witness indistinguishability and extractability to conclude that all these properties transfer successfully. (A property specification can be given for SND-P, but it is *not* polynomial time.)

<u>Discussion.</u> It is valuable, for applications, to have proof systems satisfying SND-P. The dual-

system framework does not automatically provide this for its induced proof systems, because these properties do not transfer. This does not, however, say whether or not the induced proof systems have these properties for *particular* dual-mode systems in the literature. For example, does the PZK mode of the [GOS06b] system satisfy SND-P? We have found neither an attack to show it does not, nor a proof to show it does, and consider this an interesting question to settle.

Broadly, our work calls for care in using dual-mode systems in applications. One needs to check that the mode one is using has the desired properties, rather than expect that they arrive by transference. Our Transfer Theorem can help with such checks.

2 Preliminaries

NOTATION. If $\boldsymbol{w}$ is a vector then $|\boldsymbol{w}|$ is its length (the number of its coordinates) and $\boldsymbol{w}[i]$ is its i-th coordinate. Strings are identified with vectors over $\{0, 1\}$, so that $|Z|$ denotes the length of a string Z and $Z[i]$ denotes its i-th bit. By ε we denote the empty string or vector. By $x\|y$ we denote the concatenation of strings x, y. If x, y are equal-length strings then $x \oplus y$ denotes their bitwise xor. If S is a finite set, then $|S|$ denotes it size.

If $\mathbb{G}$ is a group, then its identity element is denoted $\mathbb{1}_{\mathbb{G}}$. If $g \in \mathbb{G}$ is a generator of a group $\mathbb{G}$ of order p, then $\mathrm{dlog}_{\mathbb{G},g}(A) \in \mathbb{Z}_p$ is the discrete logarithm of $A \in \mathbb{G}$ to base g.

If X is a finite set, we let $x \leftarrow\!\$\, X$ denote picking an element of X uniformly at random and assigning it to x. Algorithms may be randomized unless otherwise indicated. If A is an algorithm, we let $y \leftarrow A^{\mathrm{O}_1,\cdots}(x_1, \ldots; \omega)$ denote running A on inputs $x_1, \ldots$ and coins ω, with oracle access to $\mathrm{O}_1, \ldots$, and assigning the output to y. By $y \leftarrow\!\$\, A^{\mathrm{O}_1,\cdots}(x_1, \ldots)$ we denote picking ω at random and letting $y \leftarrow A^{\mathrm{O}_1,\cdots}(x_1, \ldots; \omega)$. We let $[A^{\mathrm{O}_1,\cdots}(x_1, \ldots)]$ denote the set of all possible outputs of A when run on inputs $x_1, \ldots$ and with oracle access to $\mathrm{O}_1, \ldots$. An adversary is an algorithm. Running time is worst case, which for an algorithm with access to oracles means across all possible replies from the oracles. We use $\perp$ (bot) as a special symbol to denote rejection, and it is assumed to not be in $\{0, 1\}^*$.

A function $\nu \colon \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p \colon \mathbb{N} \to \mathbb{R}$ there is a $\lambda_p \in \mathbb{N}$ such that $\nu(\lambda) \leq 1/p(\lambda)$ for all $\lambda \geq \lambda_p$. "PT" stands for "polynomial time." By 1^λ we denote the unary representation of the integer security parameter $\lambda \in \mathbb{N}$.

GAMES. We use the code-based game-playing framework of BR [BR06]. A game G (see Figure 2 for examples) starts with an optional INIT procedure, followed by a non-negative number of additional procedures, and ends with a FIN procedure. Execution of adversary A with game G consists of running A with oracle access to the game procedures (which accordingly are also called oracles), with the restrictions that A's first call must be to INIT (if present), its last call must be to FIN, and it can call these two procedures at most once each. The output of the execution is the output of FIN. By $\Pr[\mathrm{G}(\mathrm{A}) \Rightarrow y]$ we denote the probability that the execution of game G with adversary A results in this output being y, and write just $\Pr[\mathrm{G}(\mathrm{A})]$ for the probability that the execution of game G with adversary A results in the output of the execution being the boolean true.

Note that our adversaries have no input or output. The role of what in other treatments is the adversary input is, for us, played by the response to the INIT query, and the role of what in other treatments is the adversary output is, for us, played by the query to FIN.

Different games may have procedures (oracles) with the same names. If we need to disambiguate, we may write G.O to refer to oracle O of game G.

In games, integer variables, set variables boolean variables and string variables are assumed initialized, respectively, to 0, the empty set $\emptyset$, the boolean false and $\perp$.

```
Game $\mathbf{G}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}$
Init():
  $(p, \mathbb{G}, g) \leftarrow\!\$\ \mathsf{GG}(1^\lambda)$ ; $a, b \leftarrow\!\$\ \mathbb{Z}_p$
  Return $(p, \mathbb{G}, g, g^a, g^b)$
Fin($C$):
  Return $(C = g^{ab})$

Game $\mathbf{G}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}$
Init():
  $(p, \mathbb{G}, g) \leftarrow\!\$\ \mathsf{GG}(1^\lambda)$ ; $d \leftarrow\!\$\ \{0,1\}$ ; $a, b, c \leftarrow\!\$\ \mathbb{Z}_p$
  If $(d = 1)$ then $C \leftarrow g^{ab}$ else $C \leftarrow g^c$
  Return $(p, \mathbb{G}, g, g^a, g^b, C)$
Fin($d'$):
  Return $(d' = d)$
```

Figure 2: Games defining the CDH and DDH assumptions for GG.

CDH and DDH assumptions. A group generator GG is a PT algorithm that takes as input a security parameter 1^λ and outputs a triple $(p, \mathbb{G}, g) \leftarrow\!\$\ \mathsf{GG}(1^\lambda)$ consisting of a prime p, (a description of) a group $\mathbb{G}$ of order p and a generator $g \in \mathbb{G} \setminus \{\mathbb{1}_{\mathbb{G}}\}$ of $\mathbb{G}$. We recall the Computational Diffie-Hellman (CDH) and Decisional Diffie-Hellman (DDH) problems associated to GG via the games in Figure 2. We define $\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(\mathrm{A})]$ to be the cdh-advantage of an adversary A. The CDH problem is hard for GG, or the CDH assumption holds for GG, if for every PT adversary A the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(\mathrm{A})$ is negligible. We define $\mathbf{Adv}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(\mathrm{A}) = 2\Pr[\mathbf{G}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(\mathrm{A})] - 1$ to be the ddh-advantage of an adversary A. The DDH problem is hard for GG, or the DDH assumption holds for GG, if for every PT adversary A the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(\mathrm{A})$ is negligible.

3 Proof systems and Dual Mode proof systems

A proof system provides a way for one party (the prover) to prove some "claim" to another party (the verifier). A claim pertains to membership of an instance x in an **NP** language, the latter defined by an **NP** relation R.

NP relations. Following [Gro06], and to cover existing proof systems, we allow the relation to depend on the CRS. Thus a relation is a function $\mathsf{R}\colon \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to \{\mathsf{true}, \mathsf{false}\}$ that takes the CRS crs, a instance x and a candidate witness w to return either true (saying w is a valid witness establishing the claim) or false (the witness fails to validate the claim). For $\mathrm{crs}, x \in \{0,1\}^*$ we let $\mathsf{R}(\mathrm{crs}, x) = \{\, w : \mathsf{R}(\mathrm{crs}, x, w) = \mathsf{true} \,\}$ be the *witness set* of x. R is said to be an **NP** relation if it is PT and there is a polynomial $\mathsf{R.wl}\colon \mathbb{N} \to \mathbb{N}$ called the maximum witness length such that every w in $\mathsf{R}(\mathrm{crs}, x)$ has length at most $\mathsf{R.wl}(|\mathrm{crs}| + |x|)$ for all $x \in \{0,1\}^*$. We let $\mathsf{L_R}(\mathrm{crs}) = \{\, x : \mathsf{R}(\mathrm{crs}, x) \neq \emptyset \,\}$ be the language associated to R and crs.

Proof systems. A proof system is the name of the syntax for the primitive that enables the production and verification of such proofs, in the classical, single-mode sense. Soundness, zero-knowledge and many other things will be security properties for such (single-mode) proof systems. We give a general, unified syntax that allows us to recover, as special cases, various models such as the common reference/random string (CRS) models [BFM88, BDSMP91, Dam00, FF00, Ps05], the designated-verifier (DV) model [ES02, PsV06, DFN06], the designated-prover (DP) model [KW18, KNYY19], and the preprocessing (PP) model [DMP90]. (Further discussion and history of these models is provided in Appendix B.) Now proceeding formally, a *proof system* $\mathsf{\Pi}$ specifies the following PT algorithms:

- *CRS generation.* Via $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\!\$\ \mathsf{\Pi.C}(1^\lambda)$, the crs-generation algorithm $\mathsf{\Pi.C}$ takes the (unary representation of the) security parameter and returns an output crs called the common reference string, a trapdoor td, a proving key k_{P} and a verification key k_{V}.
- *Proof generation.* Via $\mathrm{pf} \leftarrow\!\$\ \mathsf{\Pi.P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w)$ the proof generation algorithm $\mathsf{\Pi.P}$ takes the

```
Game $\mathbf{G}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}$

INIT():
1 $b \leftarrow\!\!\$ \{0,1\}$
2 $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\!\!\$ \mathsf{D\Pi.C}(1^\lambda, b)$
3 Return $(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$
FIN($b'$):
4 Return $(b' = b)$
```

```
Games $\boxed{\mathbf{G}^{\mathrm{snd\text{-}p}}_{\Pi,\mathsf{R},\lambda}}$ / $\mathbf{G}^{\mathrm{snd\text{-}e}}_{\Pi,\mathsf{R},\lambda}$

INIT():
1 $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\!\!\$ \Pi.\mathsf{C}(1^\lambda)$ ; Return $(1^\lambda, \mathrm{crs}, k_{\mathsf{P}})$
VF($x$, pf):
2 $d \leftarrow \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf})$
3 If $(x \in \mathsf{L_R}(\mathrm{crs}))$ then bad $\leftarrow$ true $\boxed{\text{; Return } d}$
4 If $(d)$ then win $\leftarrow$ true
5 Return $d$
FIN():
6 Return win
```

Figure 3: Left: Game defining mode indistinguishability for a dual-mode proof system $\mathsf{D\Pi}$. Right: Games defining SND-P and SND-E soundness of proof system Π for relation R.

unary security parameter, crs, a prover key k_{P}, an instance x and a witness w to produce a proof string.

- *Proof verification.* Via $d \leftarrow \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf})$ the deterministic proof verification algorithm $\Pi.\mathsf{V}$ produces a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ indicating whether or not it considers pf valid.

We say that Π satisfies completeness for relation R if $\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \in [\Pi.\mathsf{C}(1^\lambda)]$, , all $x \in \mathsf{L_R}(\mathrm{crs})$, all $w \in \mathsf{R}(\mathrm{crs}, x)$ and all $\mathrm{pf} \in [\Pi.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w))]$. This required completeness is perfect, but this can be relaxed if necessary.

RECOVERING THE DIFFERENT MODELS WITHIN OUR SYNTAX. The *common reference string (CRS) model* is the special case of our syntax in which $\Pi.\mathsf{C}$ always sets both the proving key and verification key to the empty string, $k_{\mathsf{P}} = k_{\mathsf{V}} = \varepsilon$. The *designated verifier (DV) model* corresponds to $\Pi.\mathsf{C}$ always setting the proving key to the empty string, $k_{\mathsf{P}} = \varepsilon$. In some constructions [PsV06, ES02], the verification key is dependent on both the crs and the trapdoor td, while in other constructions [QRW19, LQR$^+$19, LPWW20] it is dependent only on the crs. This distinction can be captured as a condition on $\Pi.\mathsf{C}$. The *designated prover (DP) model* corresponds to $\Pi.\mathsf{C}$ always setting the verification key to the empty string, $k_{\mathsf{V}} = \varepsilon$. The *preprocessing (PP) model* is captured by our syntax with no further restrictions. Here, the proving and verification keys may be dependent on the crs [KNYY19], or it might even be that the crs and td are set to the empty string and the keys do not depend on these parameters [KW18], all of which can be captured as conditions on $\Pi.\mathsf{C}$.

DUAL-MODE SYSTEMS. We define a dual-mode proof system $\mathsf{D\Pi}$ as also specifying a PT CRS generation algorithm $\mathsf{D\Pi.C}$, a PT proof generation algorithm $\mathsf{D\Pi.P}$ and a PT deterministic proof verification algorithm $\mathsf{D\Pi.V}$. The syntax of the last two is identical to that in a proof system as defined above. The difference is the CRS generator $\mathsf{D\Pi.C}$, which now (in addition to 1^λ) takes an input $\mu \in \{0,1\}$ called the *mode*, and returns a tuple $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}})$, as before.

For the common reference string model, the security requirement for a dual-mode proof system would be that the common reference strings created in the two modes are computationally indistinguishable. We suggest and introduce a generalization to our broader syntax, asking that the common reference strings and the proving and verification keys created in the two modes are indistinguishable. We call this mode indistinguishability. To formalize this, consider game $\mathbf{G}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}$ of Figure 3 associated to dual-mode proof system $\mathsf{D\Pi}$, and let the mode advantage of adversary A be defined by $\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}) = 2\Pr[\mathbf{G}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A})] - 1$. Mode indistinguishability asks that for all PT adversaries A, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A})$ is negligible.

A dual-mode proof system $\mathsf{D\Pi}$ gives rise to two (standard) proof systems that we call *the* *proof systems induced by* $\mathsf{D\Pi}$ and denote $\mathsf{D\Pi}_1$ and $\mathsf{D\Pi}_0$. Their proof generation and verification algorithms are those of $\mathsf{D\Pi}$, meaning $\mathsf{D\Pi}_\mu.\mathsf{P} = \mathsf{D\Pi}.\mathsf{P}$ and $\mathsf{D\Pi}_\mu.\mathsf{V} = \mathsf{D\Pi}.\mathsf{V}$, for both $\mu \in \{0, 1\}$. The difference between the two proof systems is in their CRS generation algorithms. Namely $\mathsf{D\Pi}_\mu.\mathsf{C}$ is defined by: $(\mathrm{crs}, \mathrm{td}, k_\mathsf{P}, k_\mathsf{V}) \leftarrow\!\$\ \mathsf{D\Pi}.\mathsf{C}(1^\lambda, \mu)$; Return $(\mathrm{crs}, \mathrm{td}, k_\mathsf{P}, k_\mathsf{V})$.

<u>Comparison with prior notions.</u> Unlike prior definitions [AFH$^+$16, HU19, LPWW20], we do not tie to $\mathsf{D\Pi}$, or mandate for it, any properties like soundness or ZK. These properties are defined (only) for (single-mode) proof systems. Accordingly, we can talk about whether or not the *induced* proof systems meet them. This allows us to decouple the core dual-mode concept from particular properties.

Our definition of mode indistinguishability asks that $(\mathrm{crs}, k_\mathsf{P}, k_\mathsf{V})$ be indistinguishable across the two modes, while prior definitions asked only that crs be indistinguishable across the modes. For the cases where dual-mode systems have been defined in the past, the two coincide. This is clearly true for the common reference string model, since here $k_\mathsf{P}, k_\mathsf{V}$ are ε. For the designated-verifier setting of [LPWW20], our definition may at first seem different, but it isn't, because in [LPWW20], k_V is determined from crs by a key-generation algorithm, and doesn't depend on the trapdoor. (And meanwhile, $k_\mathsf{P} = \varepsilon$.) A distinguisher can thus compute $k_\mathsf{P}, k_\mathsf{V}$ from crs, making our definition equivalent to that of [LPWW20] in this case. The case where our definition is different from asking just for crs indistinguishability is when $k_\mathsf{P}, k_\mathsf{V}$ depend on the coins underlying $\mathrm{crs}, \mathrm{td}$. Dual-mode systems for this setting, however, do not seem to have been defined prior to our work.

4 A study in soundness

We study soundness transference, showing that whether or not it holds depends on exactly how soundness is defined. We start thus with definitions.

<u>SND-E and SND-P soundness.</u> Soundness for a relation R asks that it be hard to create a valid proof for $x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})$. We consider two ways to define this, namely the penalty style (SND-P) and the exclusion style (SND-E). These two styles, and their issues, were first formally considered in [BHK15] in the context of IND-CCA public-key encryption and KEMs. In the penalty style the adversary is penalized by the game when it submits a verification query where the claim is in the language, the game testing this and not allowing it to win in this case. In the exclusion style, the adversary is simply prohibited from making queries with claims in the language, meaning a claim of SND-E security quantifies only over the sub-class of adversaries that never make such queries (or make them with negligible probability).

Consider the games in Figure 3. A verification oracle $\textsc{Vf}$ is used to cover the designated-verifier setting. Game $\mathbf{G}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}$ includes the boxed code, so that when the adversary submits $x \in \mathsf{L}_\mathsf{R}(\mathrm{crs})$ to $\textsc{Vf}$, the oracle returns the output of the verification algorithm on the query without setting the win flag. Otherwise, the game returns the decision taken by $\mathsf{\Pi}.\mathsf{V}$ based on the instance x (now known not to be in $\mathsf{L}_\mathsf{R}(\mathrm{crs})$) and proof pf provided by the adversary, and only sets win if the verifier algorithm returns true in this case. From the transference perspective, the relevant fact is that the membership test will usually not be PT. We let $\mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A})]$ be the sndp-advantage of A. We say that $\mathsf{\Pi}$ is computational SND-P for R if for all PT A the function $\lambda \mapsto \mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A})$ is negligible; statistical SND-P if for all A, regardless of running time, the function $\lambda \mapsto \mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A})$ is negligible; perfect SND-P if for all A, regardless of running time, $\mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A}) = 0$. Saying just that $\mathsf{\Pi}$ is SND-P means it could be any of the three, meaning the default assumption is computational.

Game $\mathbf{G}^{\text{snd-e}}_{\mathsf{\Pi},\mathsf{R},\lambda}$ excludes the boxed code. If the adversary submits $x \in \mathsf{L}_\mathsf{R}(\mathrm{crs})$ to $\textsc{Vf}$, the game sets the flag bad but does nothing beyond that, so that, regardless of whether or not x is in $\mathsf{L}_{\mathsf{R}}(\mathrm{crs})$, the game's determination of whether or not the adversary wins is based entirely on the verifier's test at line 4. The benefit, from the transference perspective, is that the check is now PT. We let $\mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})]$ be the snde-advantage of A.

The flag bad does not influence the game outcome. It is there to allow us to make the following definition: We say that an adversary A is *membership conscious* if the function $\lambda \mapsto \Pr[\mathbf{G}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})$ sets $\mathsf{bad}]$ is negligible. Now we say that Π is computational SND-E for R if for all PT, membership-conscious A the function $\lambda \mapsto \mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})$ is negligible; statistical SND-E if for all membership-conscious A, regardless of running time, the function $\lambda \mapsto \mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})$ is negligible; perfect SND-E if for all membership-conscious A, regardless of running time, $\mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = 0$. Saying just that Π is SND-E means it could be any of the three, meaning the default assumption is computational.

In SND-E, there is no check as to whether or not the adversary is membership conscious. The quantification is simply restricted to ones that are.

<u>Relations between definitions.</u> To better understand the differences between the notions, we briefly study some relations between them. The following says that SND-P always implies SND-E:

Proposition 4.1 *Let Π be a proof system and R a relation. If Π is computational (respectively statistical, perfect) SND-P then it is also computational (respectively statistical, perfect) SND-E.*

The proof is simple. Let A be a membership-conscious adversary A that wins the SND-E game, violating SND-E. Then it also wins the SND-P game. (Its advantage as measured in the SND-P game can only be more than its advantage as measured in the SND-E game.) And since SND-P quantifies over all PT adversaries (whether membership conscious or not), we have an adversary violating SND-P.

Another observation is that an unbounded adversary can check membership in the language. Due to this, the two formulations of soundness coincide in the statistical and perfect cases:

Proposition 4.2 *Let Π be a proof system and R a relation. Then (1) Π is statistically SND-P for R iff it is statistically SND-E for R, and (2) Π is perfectly SND-P for R iff it is perfectly SND-E for R.*

The interesting case is the computational one. Here the two definitions do *not* coincide. We defer the proof of the following to Appendix A.

Proposition 4.3 *Assume there exists a group generator relative to which DDH is hard. Then there exists a proof system Π and relation R such that (1) Π is SND-E for R but (2) Π is not SND-P for R.*

We will see in Section 6 that SND-P is what works for applications, SND-E being too weak. Thus, it is desirable that SND-P transfer. Unfortunately, we show below that, in general, it does not. Later, we will show that SND-E does, however, transfer.

<u>Non-transference of SND-P.</u> We give a counter-example to show that SND-P does not, in general, transfer. The counter-example constructs an explicit relation R and dual-mode proof system $\mathsf{D\Pi}$ such that $\mathsf{D\Pi}$ satisfies mode indistinguishability and SND-P holds in mode 1, but we can give an attack showing it does not hold in mode 0. $\mathsf{D\Pi}$ is a conventional (common reference string model) system, meaning $k_{\mathsf{P}} = k_{\mathsf{V}} = \varepsilon$, which means the result holds also in the designated verifier and prover settings.

Theorem 4.4 *Assume there exists a group generator relative to which DDH is hard. Then there exists a dual-mode proof system $\mathsf{D\Pi}$ and relation R such that (1) $\mathsf{D\Pi}$ satisfies mode indistinguishability and (2) $\mathsf{D\Pi}_1$ satisfies SND-P for R, but (3) $\mathsf{D\Pi}_0$ does **not** satisfy SND-P for R.*

```
Relation $\mathsf{R}((p,\mathbb{G},g,A,B,C),X,w)$:
  Return $(g^w = A) \wedge (X \neq B^w) \wedge (X \in \mathbb{G})$

$\mathsf{D\Pi.C}(1^\lambda,\mu)$:
  $(p,\mathbb{G},g) \leftarrow\$ \mathsf{GG}(1^\lambda)$ ; $a,b \leftarrow\$ \mathbb{Z}_p$
  If $(\mu = 1)$ then $c \leftarrow\$ \mathbb{Z}_p$ else $c \leftarrow ab \mod p$
  $A \leftarrow g^a$ ; $B \leftarrow g^b$ ; $C \leftarrow g^c$ ; crs $\leftarrow (p,\mathbb{G},g,A,B,C)$
  td $\leftarrow \varepsilon$ ; $k_{\mathsf{P}} \leftarrow \varepsilon$ ; $k_{\mathsf{V}} \leftarrow \varepsilon$ ; Return (crs, td, $k_{\mathsf{P}}$, $k_{\mathsf{V}}$)

$\mathsf{D\Pi.P}(1^\lambda,\mathrm{crs},k_{\mathsf{P}},X,w)$:
  $(p,\mathbb{G},g,A,B,C) \leftarrow$ crs ; Return $\varepsilon$

$\mathsf{D\Pi.V}(1^\lambda,\mathrm{crs},k_{\mathsf{V}},X,\mathrm{pf})$:
  $(p,\mathbb{G},g,A,B,C) \leftarrow$ crs
  If $(X \notin \mathbb{G})$ then return false else return true

Adversary $\mathrm{A}_0$:
  $(\mathrm{crs},k_{\mathsf{P}}) \leftarrow\$ \mathbf{G}^{\text{snd-p}}_{\mathsf{D\Pi_0},\mathsf{R},\lambda}.\textsc{Init}$
  $(p,\mathbb{G},g,A,B,C) \leftarrow$ crs
  $\mathbf{G}^{\text{snd-p}}_{\mathsf{D\Pi_0},\mathsf{R},\lambda}.\textsc{Vf}(C,\varepsilon)$ ; $\mathbf{G}^{\text{snd-p}}_{\mathsf{D\Pi_0},\mathsf{R},\lambda}.\textsc{Fin}$

Adversary $\mathrm{A}_{\mathrm{ddh}}$:
  $(p,\mathbb{G},g,A,B,C) \leftarrow\$ \mathbf{G}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}.\textsc{Init}$ ; $\mathrm{A}^{\textsc{Init},\textsc{Fin}}$
$\textsc{Init}$:
  Return $((p,\mathbb{G},g,A,B,C),\varepsilon,\varepsilon)$
$\textsc{Fin}(b')$:
  $\mathbf{G}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}.\textsc{Fin}(1-b')$

Adversary $\mathrm{A}_{\mathrm{cdh}}$:
  $(p,\mathbb{G},g,A,B) \leftarrow\$ \mathbf{G}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}.\textsc{Init}$ ; $\mathrm{A}^{\textsc{Init},\textsc{Vf},\textsc{Fin}}$
$\textsc{Init}$:
  $c \leftarrow\$ \mathbb{Z}_p$ ; $C \leftarrow g^c$
  Return $((p,\mathbb{G},g,A,B,C),\varepsilon)$
$\textsc{Vf}(X,\mathrm{pf})$:
  $S \leftarrow S \cup \{X\}$
  If $(X \in \mathbb{G})$ then return true
  Return false
$\textsc{Fin}()$:
  $X \leftarrow\$ S$ ; $\mathbf{G}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}.\textsc{Fin}(X)$
```

Figure 4: Relation R, dual-mode proof system $\mathsf{D\Pi}$ and various adversaries for the proof of Theorem 4.4.

Proof of Theorem 4.4: Let GG be a group generator relative to which DDH is hard. Consider the relation R shown in Figure 4. Its first input is the CRS, which has the form $(p,\mathbb{G},g,A,B,C)$, where $(p,\mathbb{G},g) \in [\mathsf{GG}(1^\lambda)]$ and $A,B,C \in \mathbb{G}$. The second input X is the instance. The third input w is the witness, which we assume is in $\mathbb{Z}_p$. Recall that $\mathrm{dlog}_{\mathbb{G},g}(Y) \in \mathbb{Z}_p$ is the discrete logarithm of $Y \in \mathbb{G}$ to base g. Let $a,b,c \in \mathbb{Z}_p$ be such that $A = g^a$, $B = g^b$ and $C = g^c$. The relation tests three things, returning true iff all hold. The test that $g^w = A$ forces the witness w to be $a = \mathrm{dlog}_{\mathbb{G},g}(A)$, meaning it can have only one value. The third test requires X to be a group element, and the second test then says that X may be any group element *except* $B^w = g^{bw} = g^{ba}$. Recall that the language associated to R, $(p,\mathbb{G},g,A,B,C)$ is the set of all X for which there exists a witness w making $\mathsf{R}((p,\mathbb{G},g,A,B,C),X,w) = \mathsf{true}$, so we have $\mathsf{L_R}((p,\mathbb{G},g,A,B,C)) = \mathbb{G} \setminus \{g^{ab}\}$, meaning it is all group elements except g^{ab}. Since violating soundness requires submitting an $X \notin \mathsf{L_R}((p,\mathbb{G},g,A,B,C))$, this means that there is only one choice of X that potentially violates soundness, namely g^{ab}.

Now consider the dual-mode proof system $\mathsf{D\Pi}$ whose algorithms $\mathsf{D\Pi.C}$, $\mathsf{D\Pi.P}$, $\mathsf{D\Pi.V}$ are described in Figure 4. The CRS generator picks a group $\mathbb{G}$ and returns CRS $(p,\mathbb{G},g,A,B,C)$ such that, again letting $A = g^a$, $B = g^b$ and $C = g^c$, if $\mu = 0$ then (A,B,C) is a DH-tuple —meaning $C = g^{ab}$— and if $\mu = 1$ then A,B,C are uniform and independent group elements. Under the DDH assumption, the two CRSs are indistinguishable, while the proving and verification keys are identical in both modes since they are set to the empty string. Formally, this is claim (1) of the theorem statement, which we show by a reduction from the mode-indistinguishability of $\mathsf{D\Pi}$ to the DDH assumption for the group generator GG. For this, let A be a PT adversary. We construct the PT time adversary $\mathrm{A}_{\mathrm{ddh}}$ shown in Figure 4. For all $\lambda \in \mathbb{N}$ we have

$$\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\mathsf{R},\lambda}(\mathrm{A}) \leq \mathbf{Adv}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(\mathrm{A}_{\mathrm{ddh}}) ,$$

which justifies claim (1). The other algorithms of $\mathsf{D\Pi}$ perform trivially: The proof generator always returns the empty string as the proof, and the verifier algorithm always accepts.

Let $\mathsf{D\Pi}_0$ and $\mathsf{D\Pi}_1$ be the induced proof systems of $\mathsf{D\Pi}$. We show claim (3) of the theorem by an attack, namely an adversary violating SND-P for $\mathsf{D\Pi}_0$. The adversary A_0 is shown in Figure 4. It starts, at line 1, by calling the $\textsc{Init}$ oracle of its game $\mathbf{G}^{\text{snd-p}}_{\mathsf{D\Pi}_0,\mathsf{R},\lambda}$. The CRS $(p, \mathbb{G}, g^a, g^b, C)$ in this game is generated by $\mathsf{D\Pi}_0.\mathsf{C}$, and hence $C = g^{ab}$. This gives the adversary an instance outside the language $\mathsf{L}_{\mathsf{R}}((p, \mathbb{G}, g^a, g^b, C))$, namely $X = C$. It can now submit C to $\textsc{Vf}$ at line 3. What it submits as the proof does not matter (the choice made is the empty string) since $\mathsf{D\Pi}.\mathsf{V}$ always accepts as long as the statement is in the group $\mathbb{G}$; the challenge in violating SND-P was to find an instance outside the language. This $\textsc{Vf}$ query will set the win flag, and therefore this adversary will win the game when it calls $\textsc{Fin}$. We have $\mathbf{Adv}^{\text{snd-p}}_{\mathsf{D\Pi}_0,\mathsf{R},\lambda}(\mathrm{A}_0) = 1$, establishing claim (3) of the theorem.

It remains to show claim (2), namely that $\mathsf{D\Pi}_1$ does satisfy SND-P. This is true under the CDH assumption on GG, which is implied by the DDH assumption we have made, and is proved by reduction. Given a PT adversary A trying to win game $\mathbf{G}^{\text{snd-p}}_{\mathsf{D\Pi}_1,\mathsf{R},\lambda}$, we construct the PT cdh-adversary A_{cdh} shown in Figure 4. It is playing game $\mathbf{G}^{\text{cdh}}_{\mathsf{GG},\lambda}$. From the $\textsc{Init}$ oracle of that game, it obtains $(p, \mathbb{G}, g, A, B)$, and then runs A, simulating A's $\textsc{Init}, \textsc{Vf}, \textsc{Fin}$ oracles as shown. When A calls $\textsc{Init}$, our A_{cdh} can return a legitimate mode-0 CRS by itself picking C at random and returning $(p, \mathbb{G}, g, A, B, C)$. When A calls $\textsc{Vf}$ with an argument X (and a proof pf which does not matter) that, if it sets the win flag, will be g^{ab}, where $A = g^a$ and $B = g^b$. If only one $\textsc{Vf}$ query was allowed, A_{cdh} could call its own $\textsc{Fin}$ oracle with X to also win. However, since multiple $\textsc{Vf}$ queries can be made, the best A_{cdh} can do is to pick one of the statements queried to the $\textsc{Vf}$ by A at random to submit to its $\textsc{Fin}$ oracle. This adds a multiplicative factor of the number of $\textsc{Vf}$ queries made by A, say $q(\lambda)$, to the bound. For all $\lambda \in \mathbb{N}$ we have

$$\mathbf{Adv}^{\text{snd-p}}_{\mathsf{D\Pi}_0,\mathsf{R},\lambda}(\mathrm{A}) \leq q \cdot \mathbf{Adv}^{\text{cdh}}_{\mathsf{GG},\lambda}(\mathrm{A}_{\text{cdh}}) .$$

This completes the proof of claim (3) and thus of the Theorem. ∎

In Theorem 4.4, the SND-P in mode-1 is computational, not statistical or perfect. A good question is, if mode-1 has statistical or perfect SND-P, then does it transfer, meaning does mode-0 have computational SND-P? The difficulty of proving the answer is "yes" remains, namely that the PT mode-indistinguishability adversary still has to test membership in the language, which may not be PT. We do not see a way in which stronger SND-P in mode-1 helps the transfer. But, for this case, neither do we have a counter-example.

In Theorem 4.4, the relation R depends on the CRS. An interesting open question is whether one can prove a similar negative result for a relation which does not depend on the CRS.

5 Transference framework and positive results

Let $\mathsf{D\Pi}$ be a dual-mode proof system satisfying mode indistinguishability. Recall we say that a "property" P (for example, zero-knowledge, soundness, extractability) transfers, if, for any $\mu \in \{0, 1\}$, we have: If $\mathsf{D\Pi}_\mu$ satisfies P then $\mathsf{D\Pi}_{1-\mu}$ satisfies the computational counterpart of P. In this Section we want to give positive results, showing some properties P do transfer.

We could try to do this exhaustively for target properties $\mathrm{P}_1, \mathrm{P}_2, \ldots$: prove P_1 transfers; then prove P_2 transfers; and so on. This ad hoc approach has several drawbacks. First, proofs which at heart are very similar will be repeated. Second, proofs will be needed again for new or further properties that we do not consider. (Counting definitional variants in the literature, the number of properties of interest, namely the length of the list above, is rather large.) Third, we'd like a better theoretical understanding of what exactly are the attributes of a property that allow transference.

To address this, we give a framework to obtain positive transference results. We start by formalizing what we call a property specification PS. While the game defining P will pick the CRS and the proving and verification keys by running the CRS generator, PS will aim to see the adversary advantage in this game as a function of an external choice of CRS and keys, effectively pulling the choice of CRS and keys out of the game. We will then give a general result (the Transfer Theorem) saying that polynomial-time property specifications transfer successfully. To apply this to get a positive transfer result for some property P of interest, one then has to show that P can be captured by a polynomial time property specification PS. We will illustrate application by providing PS explicitly for a few choices of P. It will soon be easy to just look at the game defining P and see from it whether or not P can be cast as a polynomial-time PS, making it simple to see which properties transfer successfully.

When the property specification PS is not polynomial time, our Transfer Theorem does not apply. This does not necessarily mean the property fails to transfer, but is an indication in that direction. To show that a particular (non polynomial-time) property P fails to transfer, one can give a counter-example, as with Theorem 4.4.

<u>Property specifications.</u> A *property specification* PS is a function that, given a proof system Π, returns a triple ($\mathsf{PS.StI}$, $\mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}$, $\mathsf{PS.type}$). The first component $\mathsf{PS.StI}$ is an algorithm that we refer to as the *state initializer*, and, as the notation indicates, it does not depend on Π. The second component $\mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}$ is an algorithm that we refer to as the *oracle responder*. We require that it invokes the prover and verifier algorithms of Π as oracles, so that if two proof systems have the same prover and verifier algorithms, the corresponding oracle responders are identical. The final component $\mathsf{PS.type} \in \{\mathsf{dec}, \mathsf{ser}\}$ is a keyword (formally, just a bit), indicating the type of problem, decision or search.

The state initializer takes the unary security parameter and a tuple $(\mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}) \in [\Pi.\mathsf{C}(1^\lambda)]$ to return an initial state, $st \leftarrow_\$ \mathsf{PS.StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$. Then, given a string $\mathrm{Oname} \in \mathsf{PS.ONames} \subseteq \{0,1\}^*$ called an *oracle name*, another string Oarg called an *oracle argument*, and also given a current state *st*, the oracle responder returns a pair $(\mathrm{Orsp}, st) \leftarrow_\$ \mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$ consisting of an *oracle response* Orsp and an updated state. The finite set of oracle names $\mathsf{PS.ONames}$ (also defined by PS but not allowed to depend on Π) must contain the special name Fin, and it must be that the response Orsp is in the set $\{\mathsf{true}, \mathsf{false}\}$ whenever $(\mathrm{Orsp}, st) \leftarrow_\$ \mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Fin}, \mathrm{Oarg}, st)$.

A property specification PS is said to be polynomial time (PT) if for every proof system Π there is a polynomial p such that algorithms $\mathsf{PS.StI}$ and $\mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}$ run in time bounded by p applied to the lengths of their inputs.

We can run PS with a proof system Π, security parameter λ, input $(\mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}) \in [\Pi.\mathsf{C}(1^\lambda)]$ and an adversary A to get a boolean output, which is denoted $\mathsf{PS}[\Pi].\mathsf{Out}_{\mathrm{crs},\lambda}(\mathrm{A})$. This output is determined by the process on the left below, the right showing the oracle provided to A—

```
PS[Π].Out_{crs,k_P k_V,λ}(A)
    $st \leftarrow_\$ \mathsf{PS.StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$
    Run $\mathrm{A}^{\textsc{Or}}$
    Return $out$

Oracle Or(Oname, Oarg)   // $\mathrm{Oname} \in \mathsf{PS.ONames}$
    $(\mathrm{Orsp}, st) \leftarrow_\$ \mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$
    If (($\mathrm{Oname} = \mathrm{Fin}$) and ($out = \bot$)) then $out \leftarrow \mathrm{Orsp}$
    Return Orsp
```

Above, the execution initializes the state to $st \leftarrow \mathsf{PS.StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$. Then it runs A with access to the oracle Or shown on the right. Given the string naming an oracle, and an argument for it, Or provides the response as defined by $\mathsf{PS}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}$. The first time the oracle named Fin is called, the computed response is retained as *out*, and the latter becomes the output of the execution, namely the value returned as $\mathsf{PS}[\Pi].\mathsf{Out}_{\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}},\lambda}(\mathrm{A})$. We define the ps-advantage of A, depending on whether it is a search or decision problem, via

$$\mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\Pi],\lambda}(\mathrm{A}) = \begin{cases} \Pr\left[\mathsf{PS}[\Pi].\mathsf{Out}_{\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}},\lambda}(\mathrm{A}) \;:\; (\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}}) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^{\lambda})\right] & \text{if } \mathsf{PS}.\mathsf{type} = \mathsf{ser} \\ 2\cdot\Pr\left[\mathsf{PS}[\Pi].\mathsf{Out}_{\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}},\lambda}(\mathrm{A}) \;:\; (\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}}) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^{\lambda})\right] - 1 & \text{if } \mathsf{PS}.\mathsf{type} = \mathsf{dec}. \end{cases}$$

Here $\Pr\left[\mathsf{PS}[\Pi].\mathsf{Out}_{\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}},\lambda}(\mathrm{A}) \;:\; (\mathrm{crs},k_{\mathsf{P}},k_{\mathsf{V}}) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^{\lambda})\right]$ is the probability that the output of the property specification is true when the CRS and proving and verification keys are chosen at random according to $\Pi.\mathsf{C}$. We say that Π *satisfies* PS *for a class (set) of adversaries* $\mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}}$ if for every adversary $\mathrm{A} \in \mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}}$, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\Pi],\lambda}(\mathrm{A})$ is negligible. Parameterizing the definition by a class of adversaries allows us to cover restrictions like membership-consciousness, and to capture computational and statistical variants of a property.

THE SND-E AND SND-P PROPERTY SPECIFICATIONS. Before providing the Transfer Theorem we pause to illustrate property specifications by an example. We describe the property specifications $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}$ and $\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}$ corresponding to the SND-E and SND-P properties for R, respectively, in Figure 6. The state initializer algorithms are the same for both, setting the initial state *st* to the crs they are given as input, together with the security parameter. The oracle responder algorithms differ only at line 6, which is included for SND-P and excluded for SND-E. To each of the oracles $\textsc{Init}, \textsc{Vf}, \textsc{Fin}$ in the games of Figure 3, we associate a string naming it, these being $\mathrm{Init}, \mathrm{Vf}, \mathrm{Fin}$, respectively, so that $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}.\mathsf{ONames} = \mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}.\mathsf{ONames} = \{\mathrm{Init}, \mathrm{Vf}, \mathrm{Fin}\}$. Passing the name of an oracle, and arguments for it, to the oracle responder, results in the response of that oracle being returned. (Also returned is the state, which is updated here, but may not be in other property specifications.) The problem type is $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}.\mathsf{type} = \mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}.\mathsf{type} = \mathsf{ser}$, meaning both are search problems. As this shows, there is a quite direct connection between the games and the property specification, the key difference being that the latter has the CRS as input while the former pick it internally.

We connect the actual properties with their formal property specifications via the following, which says that, for $\mathrm{x} \in \{\mathrm{e}, \mathrm{p}\}$, the sndx-advantage is identical to the corresponding ps-advantage.

Proposition 5.1 *Let* R *be an NP-relation,* Π *a proof system and* A *an adversary. Then for all* $\lambda \in \mathbb{N}$ *we have:*

$$\mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}[\Pi],\lambda}(\mathrm{A}) \;\; \textit{and} \;\; \mathbf{Adv}^{\text{snd-p}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}[\Pi],\lambda}(\mathrm{A}) \,.$$

In the case of SND-E, Proposition 5.1 does not restrict to membership conscious adversaries, even though these are the ones of eventual interest; the claim of the Proposition is true for all adversaries.

The key difference between the two property specifications is in their running time. Property specification $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}$ is polynomial time. But property specification $\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}$ is only polynomial time if testing membership of x in $\mathsf{L}_{\mathsf{R}}(\mathrm{crs})$ can be done in time polynomial in the lengths of x and crs, which, for relations of interest, is usually *not* the case. Our Transfer Theorem applies to $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}$ but, due to its not in general being polynomial time, not to $\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}$.

TRANSFER THEOREM. Refer above for what it means for a proof system Π to satisfy a property specification PS for a class of adversaries $\mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}}$. The following says that when this is true in one mode of a dual-mode proof system $\mathsf{D}\Pi$, then its computational counterpart is true in the other mode. Below, we let $\mathcal{A}^{\mathrm{PT}}$ be the class of all polynomial-time adversaries; the intersection of $\mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}}$ with $\mathcal{A}^{\mathrm{PT}}$ in the conclusion of the Theorem captures that the transferred property is the computational counterpart of the original one.

```
Game $\mathrm{G}_d$   // $d \in \{0,1\}$
INIT():
  $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\$ \mathsf{D\Pi.C}(1^\lambda, d)$
  $st \leftarrow\$ \mathsf{PS.StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$
OR(Oname, Oarg):
  $(\mathrm{Orsp}, st) \leftarrow\$ \mathsf{PS[D\Pi.P, D\Pi.V].Or}(\mathrm{Oname}, \mathrm{Oarg})$
  If $((\mathrm{Oname} = \mathrm{Fin})$ and $(out = \bot))$ then
    $out \leftarrow \mathrm{Orsp}$
  Return Oname
FIN():
  Return $out$
```

```
Adversary $\mathrm{A}_{\mathrm{mode}}$
  $(\mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\$ \mathbf{G}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}.\mathrm{INIT}$
  $st \leftarrow \mathsf{PS.StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$
  Run $\mathrm{A}^{\mathrm{OR}}$
  If $out$ then $a \leftarrow 1$ else $a \leftarrow 0$
  Return $(a \oplus \mu)$
OR(Oname, Oarg):
  $(\mathrm{Orsp}, st) \leftarrow\$ \mathsf{PS[D\Pi.P, D\Pi.V].Or}(\mathrm{Oname}, \mathrm{Oarg})$
  If $((\mathrm{Oname} = \mathrm{Fin})$ and $(out = \bot))$ then
    $out \leftarrow \mathrm{Orsp}$
  Return Oname
```

Figure 5: Left: Games for the proof of Theorem 5.2. Right: Adversary for the proof of Theorem 5.2.

Theorem 5.2 *Let $\mathsf{D\Pi}$ be a dual-mode proof system that is mode indistinguishable. Let $\mu \in \{0,1\}$. Let PS be a polynomial-time property specification. Assume $\mathsf{D\Pi}_\mu$ satisfies PS for a class of adversaries $\mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}}$. Then $\mathsf{D\Pi}_{1-\mu}$ satisfies PS for the class of adversaries $\mathcal{A}^{\mathrm{ps}}_{\mathsf{PS}} \cap \mathcal{A}^{\mathrm{PT}}$.*

Proof of Theorem 5.2: Let A be a polynomial-time adversary. We build a polynomial-time adversary $\mathrm{A}_{\mathrm{mode}}$ such that for all $\lambda \in \mathbb{N}$ we have

$$\mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{1-\mu}],\lambda}(\mathrm{A}) \leq \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{\mu}],\lambda}(\mathrm{A}) + 2 \cdot \mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}}) \; . \tag{1}$$

The theorem follows.

To establish Equation (1), recall that, as per the definition of a property specification, the oracle responder algorithm depends only on the prover and verifier algorithms of the proof system it is given, invoking these as oracles. But, by the definition of the proof systems induced proof by a dual-mode proof system, for both $\mathsf{D\Pi}_0$ and $\mathsf{D\Pi}_1$, the prover algorithm is $\mathsf{D\Pi.P}$ and the verifier algorithm is $\mathsf{D\Pi.V}$. This means that the oracle responder algorithms corresponding to $\mathsf{D\Pi}_0$ and $\mathsf{D\Pi}_1$ are identical, both being $\mathsf{PS[D\Pi.P, D\Pi.V].Or}$.

With this, consider the the games G_d, defined, for $d \in \{0,1\}$, in Figure 5. At line 1, they generate the CRS and the proving and verification keys in mode d, and then run the state initializer (we use here the fact that it does not depend on the proof system) to get an initial state. In responding to OR queries at line 2, they use the oracle responder with prover and verifier algorithms set to those of the dual-mode proof system, as per the above. Then for both $d = 0$ and $d = 1$ we have

$$\mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_d],\lambda}(\mathrm{A}) = \begin{cases} \Pr[\mathrm{G}_d(\mathrm{A})] & \text{if } \mathsf{type} = \mathsf{ser} \\ 2\Pr[\mathrm{G}_d(\mathrm{A})] - 1 & \text{if } \mathsf{type} = \mathsf{dec}. \end{cases}$$

Thus if $\mathsf{type} = \mathsf{ser}$ we have

$$\begin{aligned} \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{1-\mu}],\lambda}(\mathrm{A}) &= \Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] = \Pr[\mathrm{G}_{\mu}(\mathrm{A})] + (\,\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})]\,) \\ &= \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{\mu}],\lambda}(\mathrm{A}) + (\,\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})]\,) \; . \end{aligned}$$

And if $\mathsf{type} = \mathsf{dec}$ we have

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{1-\mu}],\lambda}(\mathrm{A}) &= 2\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - 1 = 2\Pr[\mathrm{G}_{\mu}(\mathrm{A})] - 1 + 2\cdot(\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})])\\ &= \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}[\mathsf{D\Pi}_{\mu}],\lambda}(\mathrm{A}) + 2\cdot(\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})])\ .\end{aligned}$$

We build PT mode indistinguishability adversary $\mathrm{A}_{\mathrm{mode}}$ so that

$$\Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})] \le \mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}})\ .$$

The adversary $\mathrm{A}_{\mathrm{mode}}$ is shown at the bottom in Figure 5. At line 1 it obtains a CRS and proving and verification keys from its own INIT oracle. It then runs A and simulates the OR oracle of the latter as shown. Let b be the randomly chosen bit in $\mathbf{G}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}$. Then

$$\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}}) = \Pr[\,a\oplus\mu = 1 \mid b = 1\,] - \Pr[\,a\oplus\mu = 1 \mid b = 0\,]\ .$$

Let us consider the two cases depending on whether μ is 0 or 1.

$$\underline{\mu = 0}: \qquad \begin{aligned}\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}}) &= \Pr[\,a = 1 \mid b = 1\,] - \Pr[\,a = 1 \mid b = 0\,]\\ &= \Pr[\mathrm{G}_1(\mathrm{A})] - \Pr[\mathrm{G}_0(\mathrm{A})]\end{aligned}$$

$$\underline{\mu = 1}: \qquad \begin{aligned}\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}}) &= \Pr[\,a = 0 \mid b = 1\,] - \Pr[\,a = 0 \mid b = 0\,]\\ &= \Pr[\,a = 1 \mid b = 0\,] - \Pr[\,a = 1 \mid b = 1\,]\\ &= \Pr[\mathrm{G}_0(\mathrm{A})] - \Pr[\mathrm{G}_1(\mathrm{A})]\end{aligned}$$

We can combine the two cases together as follows:

$$\mathbf{Adv}^{\mathrm{mode}}_{\mathsf{D\Pi},\lambda}(\mathrm{A}_{\mathrm{mode}}) = \Pr[\mathrm{G}_{1-\mu}(\mathrm{A})] - \Pr[\mathrm{G}_{\mu}(\mathrm{A})]\ .$$

This completes the proof. ▮

<u>TRANSFERENCE OF SND-E.</u> We can apply this to conclude transference of SND-E as follows. Let $\mathcal{A}^{\mathrm{snde}}$ be the class of membership-conscious, PT adversaries. (Restricting, here, to PT only strengthens the result.) Let PS be $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}$. Then combining Theorem 5.2 with Proposition 5.1 says that if $\mathsf{D\Pi}_{\mu}$ is SND-E then so is $\mathsf{D\Pi}_{1-\mu}$.

This is however a simple case. For ZK, the property specification definition is more delicate, and some work will be needed to check that it obeys the conditions required by the definition of a property specification.

We now turn to establishing transference for other properties. We will give their definitions, and the property specifications, side by side.

<u>ZERO KNOWLEDGE.</u> The property specification allowing showing transference for ZK is more interesting, in part because it is a decision problem.

We formalize what is usually called adaptive zero knowledge, as the form most useful for applications. A *simulator* S is specifies a PT algorithm $\mathsf{S.C}$ (the simulation CRS-generator) and a PT algorithm $\mathsf{S.P}$ (the simulation proof-generator). Consider game $\mathbf{G}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}$ specified in Figure 7. ZK-adversary A can adaptively request proofs by supplying an instance and a valid witness for it. The proof is produced either by the honest prover using the witness, or by the proof simulator $\mathsf{S.P}$ using a simulation trapdoor td_0. The adversary outputs a guess b' as to whether the proofs were real or simulated. Let $\mathbf{Adv}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}) = 2\Pr[\mathbf{G}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}(\mathrm{A})] - 1$ be its zk-advantage relative to S.

We say that $\mathsf{\Pi}$ is computational ZK for R if there exists a simulator S such that for all PT A the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}(\mathrm{A})$ is negligible; statistical ZK for R if there exists a simulator S such that

```
State initializer $\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}.\mathsf{StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$ / $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}.\mathsf{StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$:
$st \leftarrow (1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$ ; Return $st$
$\boxed{\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)}$ / $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$:
$(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}, \mathsf{win}) \leftarrow st$
If $(\mathrm{Oname} = \mathrm{Init})$ then return $((1^\lambda, \mathrm{crs}, k_{\mathsf{P}}), st)$
If $(\mathrm{Oname} = \mathrm{Vf})$ then
    $(x, \mathrm{pf}) \leftarrow \mathrm{Oarg}$
    $\boxed{\text{If } (x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs})) \text{ then return } (\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}), st)}$
    If $(\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}))$ then $\mathsf{win} \leftarrow \mathsf{true}$ ; $st \leftarrow (1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}, \mathsf{win})$
    Return $(\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}), st)$
If $(\mathrm{Oname} = \mathrm{Fin})$ then return $(\mathsf{win}, st)$
```

Figure 6: Algorithms associated by the SND-P property specification $\mathsf{PS}^{\mathrm{sndp}}_{\mathsf{R}}$ and the SND-E property specification $\mathsf{PS}^{\mathrm{snde}}_{\mathsf{R}}$ to proof system Π, where R is an NP-relation. The boxed code is only included in the SND-P specification.

```
Game $\mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}$
INIT():
$(\mathrm{crs}_1, \mathrm{td}_1, k_{\mathsf{P}}, k_{\mathsf{V}1}) \leftarrow\$ \Pi.\mathsf{C}(1^\lambda)$
$(\mathrm{crs}_0, \mathrm{td}_0, k_{\mathsf{V}0}) \leftarrow\$ \mathsf{S}.\mathsf{C}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$ ; $\mathrm{crs} \leftarrow \mathrm{crs}_b$ ; $k_{\mathsf{V}} \leftarrow k_{\mathsf{V}b}$
Return $(1^\lambda, \mathrm{crs}, k_{\mathsf{V}})$
PF$(x, w)$:
If $(\neg\mathsf{R}(\mathrm{crs}, x, w))$ then return $\bot$
If $(b = 1)$ then $\mathrm{pf} \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w)$
Else $\mathrm{pf} \leftarrow\$ \mathsf{S}.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, \mathrm{td}_0, x)$
Return pf
FIN$(b')$:
Return $(b' = b)$
```

```
State initializer $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}.\mathsf{StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$:
$b \leftarrow\$ \{0,1\}$
If $(b = 0)$ then $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{V}}) \leftarrow\$ \mathsf{S}.\mathsf{C}(1^\lambda)$
$st \leftarrow (1^\lambda, \mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}, b)$ ; Return $st$
Oracle responder $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$:
$(1^\lambda, \mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}, b) \leftarrow st$
If $(\mathrm{Oname} = \mathrm{Init})$ then return $(\mathrm{crs}, k_{\mathsf{V}}, st)$
If $(\mathrm{Oname} = \mathrm{Pf})$ then
    $(x, w) \leftarrow \mathrm{Oarg}$
    If $(\neg\mathsf{R}(\mathrm{crs}, x, w))$ then return $(\bot, st)$
    If $(b = 1)$ then $\mathrm{pf} \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w)$
    Else $\mathrm{pf} \leftarrow\$ \mathsf{S}.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, \mathrm{td}, x)$
    Return $(\mathrm{pf}, st)$
If $(\mathrm{Oname} = \mathrm{Fin})$ then
    $b' \leftarrow \mathrm{Oarg}$ ; Return $((b = b'), st)$
```

Figure 7: Left: Game defining zero-knowledge (relative to simulator S) for proof system Π. Right: Algorithms associated by the ZK property specification $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}$ to proof system Π, where R is an NP-relation and S is a simulator.

for all A, regardless of running time, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A})$ is negligible; and perfect ZK for R if there exists a simulator S such that for all A, regardless of running time, $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}) = 0$. Saying just that Π is ZK means it could be any of the three, meaning the default assumption is computational.

We describe the property specification $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}$ corresponding to the ZK property for relation R and simulator S. Figure 7 shows the algorithms that it associates to a given proof system Π. While game $\mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S}}$ of Figure 7 picks the CRS and the proving and verification keys via $\Pi.\mathsf{C}$ when $b = 1$, state initializer $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}.\mathsf{StI}$ takes the CRS and the proving and verification keys as input and sets this CRS as the CRS when $b = 1$. If $b = 0$, this CRS is overwritten at line 2. The state is the 6-tuple at line 4. To each oracle INIT, PF, FIN in game $\mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S}}$ we associate a string naming it, these being Init, Pf, Fin, respectively, so that $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}.\mathsf{ONames} = \{\mathrm{Init}, \mathrm{Pf}, \mathrm{Fin}\}$. Passing the name of an oracle,

```
Game $\mathbf{G}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}$

INIT():
  $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\!\$\ \Pi.\mathsf{C}(1^\lambda)$
  $b \leftarrow\!\$\ \{0,1\}$
  Return $(1^\lambda, \mathrm{crs}, k_{\mathsf{V}})$
PF$(x, w_0, w_1)$:
  $d_0 \leftarrow \mathsf{R}(\mathrm{crs}, x, w_0))$
  $d_1 \leftarrow \mathsf{R}(\mathrm{crs}, x, w_1)$
  If $(\neg d_0$ or $\neg d_1)$ then return $\perp$
  $\mathrm{pf} \leftarrow\!\$\ \Pi.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w_b)$
  Return pf
FIN$(b')$:
  Return $(b' = b)$
```

```
State initializer $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}.\mathsf{StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$:
  $b \leftarrow\!\$\ \{0,1\}$ ; $st \leftarrow (1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}, b)$ ; return $st$
Oracle responder $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$:
  $(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}}, b) \leftarrow st$
  If (Oname = Init) then return $(\mathrm{crs}, k_{\mathsf{V}}, st)$
  If (Oname = Pf) then
    $(x, w_0, w_1) \leftarrow$ Oarg
    If $(\neg\mathsf{R}(\mathrm{crs}, x, w_0)$ or $\neg\mathsf{R}(\mathrm{crs}, x, w_1))$ then
      Return $(\perp, st)$
    $\mathrm{pf} \leftarrow\!\$\ \Pi.\mathsf{P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, w_b)$ ; return $(\mathrm{pf}, st)$
  If (Oname = Fin) then
    $b' \leftarrow$ Oarg ; Return $((b = b'), st)$
```

Figure 8: Left: Games defining witness indistinguishability of proof system Π. Right: Algorithms associated by the WI property specification $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}$ to proof system Π, where R is an NP-relation.

and arguments for it, to $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}$, results in the response of that oracle being returned. (Also returned is the state, which here is not updated, but may be in other property specifications.) The type is $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}.\mathsf{type} = \mathsf{dec}$, meaning this is a decision problem.

To connect the ZK property with the property specification $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}$, we see that the zk-advantage is identical to the ps-advantage:

Proposition 5.3 *Let* R *be an NP-relation,* S *a simulator,* Π *a proof system and* A *an adversary. Then for all* $\lambda \in \mathbb{N}$ *we have:*

$$\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}) = \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}[\Pi],\lambda}(\mathrm{A})\ .$$

From Figure 7, one can see that the ZK property specification $\mathsf{PS}^{\mathrm{zk}}_{\mathsf{R},\mathsf{S}}$ is polynomial time. Combining Theorem 5.2 with Proposition 5.3 thus says that the ZK property transfers.

<u>WITNESS INDISTINGUISHABILITY.</u> This asks that, knowing $x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs})$ and knowing two witnesses $w_0, w_1 \in \mathsf{R}(\mathrm{crs}, x)$, it is hard to tell under which of the two a proof has been computed [FS90]. Consider game $\mathbf{G}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}$ specified in Figure 8. Let $\mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = 2\Pr[\mathbf{G}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})] - 1$. We say that Π is computational WI for R if for all PT A the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})$ is negligible; statistical WI if for all A, regardless of running time, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A})$ is negligible; perfect WI if for all A, regardless of running time, $\lambda \mapsto \mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = 0$. Saying just that Π is WI means it could be any of the three, meaning the default assumption is computational.

We describe the algorithms of the property specification $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}$ corresponding to the WI property for the relation R in Figure 8. The type is $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}.\mathsf{type} = \mathsf{dec}$, meaning it is a decision problem. Connecting the WI property with its property specification $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}$, we claim that the wi-advantage is identical to the ps-advantage.

Proposition 5.4 *Let* R *be an NP-relation,* Π *a proof system and* A *an adversary. Then for all* $\lambda \in \mathbb{N}$ *we have:*

$$\mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\lambda}(\mathrm{A}) = \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}[\Pi],\lambda}(\mathrm{A})\ .$$

Since $\mathsf{PS}^{\mathrm{wi}}_{\mathsf{R}}$ is also PT, combining Theorem 5.2 with Proposition 5.4 thus says that the WI property transfers.

```
Game $\mathbf{G}^{\mathrm{xt}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}$

INIT():
1  $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow\!\!\$\ \mathsf{S.C}(1^\lambda)$
2  Return $(1^\lambda, \mathrm{crs}, k_{\mathsf{P}})$

VF$(x, \mathrm{pf})$:
3  Return $\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf})$

FIN$(x, \mathrm{pf})$:
4  If $(\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}) = \mathsf{false})$ then
       return $\mathsf{false}$
5  $w \leftarrow\!\!\$\ \mathsf{S.X}(1^\lambda, \mathrm{crs}, \mathrm{td}, x, \mathrm{pf})$
6  Return $\neg\mathsf{R}(\mathrm{crs}, x, w)$
```

```
State initializer $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}.\mathsf{StI}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, k_{\mathsf{V}})$:
1  $(\mathrm{crs}', \mathrm{td}, k'_{\mathsf{P}}, k'_{\mathsf{V}}) \leftarrow\!\!\$\ \mathsf{S.C}(1^\lambda)$
2  $st \leftarrow (1^\lambda, \mathrm{crs}', \mathrm{td}, k'_{\mathsf{P}}, k'_{\mathsf{V}})$ ; Return $st$
Oracle responder $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}[\Pi.\mathsf{P}, \Pi.\mathsf{V}].\mathsf{Or}(\mathrm{Oname}, \mathrm{Oarg}, st)$:
3  $(1^\lambda, \mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}}) \leftarrow st$
4  If $(\mathrm{Oname} = \mathrm{Init})$ then return $((1^\lambda, \mathrm{crs}, k_{\mathsf{P}}), st)$
5  If $(\mathrm{Oname} = \mathrm{Vf})$ then return $\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf})$
6  If $(\mathrm{Oname} = \mathrm{Fin})$ then
7      $(x, \mathrm{pf}) \leftarrow \mathrm{Oarg}$
8      If $(\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, k_{\mathsf{V}}, x, \mathrm{pf}) = \mathsf{false})$ then return $\mathsf{false}$
9      $w \leftarrow\!\!\$\ \mathsf{S.X}(1^\lambda, \mathrm{crs}, \mathrm{td}, x, \mathrm{pf})$
10     Return $((\mathsf{R}(\mathrm{crs}, x, w) = \mathsf{false}), st)$
```

Figure 9: Top: Game defining XT extractability of a proof system Π for a relation R. Bottom: Algorithms associated by the XT property specification $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}[\Pi]$ to proof system Π, where R is an NP-relation and S is a simulator.

XT EXTRACTABILITY. The notion of Π being a proof of knowledge [GMR89, BG93, DP92] for R requires that whenever a (potentially cheating) prover, modeled as the adversary, is able to produce a valid proof, there is an extractor that, based on a trapdoor underlying the common reference string, can extract the witness from the information available to the adversary. We generalize extractability for the different models we consider along the lines of [CC18] (which defined knowledge-extractability for DV-NIZKs). Our formalization is via game $\mathbf{G}^{\mathrm{xt}}$ specified in Figure 3. It is parameterized by an *extractor* S, an object that specifies algorithms $\mathsf{S.C}$ (the extraction-CRS generator) and $\mathsf{S.X}$ (the extraction witness-generator). Let $\mathbf{Adv}^{\mathrm{xt}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\mathrm{xt}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A})]$ be the xt-advantage of A.

The notion of Π being XT-secure would be that there exists a polynomial time extractor S such that $\lambda \mapsto \mathbf{Adv}^{\mathrm{xt}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A})$ is negligible for all polynomial time adversaries A.

THE XT PROPERTY SPECIFICATION. We describe the property specification $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}$ corresponding to the XT property for the relation R in Figure 9. The type is $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}.\mathsf{type} = \mathsf{ser}$, meaning it is a search problem. Connecting the XT property with its property specification $\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}$, we claim that the xt-advantage is identical to the ps-advantage.

Proposition 5.5 *Let* R *be an NP-relation,* Π *a proof system and* A *an adversary. Then for all* $\lambda \in \mathbb{N}$ *we have:*

$$\mathbf{Adv}^{\mathrm{xt}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}) = \mathbf{Adv}^{\mathrm{ps}}_{\mathsf{PS}^{\mathrm{xt}}_{\mathsf{R},\mathsf{S}}[\Pi],\lambda}(\mathrm{A}) \; .$$

6 SND in application: A test case

The properties that one would most like to successfully transfer are the ones of most utility in applications. Since whether or not soundness transfers depends on exactly how it is defined (recall that SND-E transfers and SND-P does not) we would like to see which of the two is needed by applications. To this end we examine here in detail one representative and canonical application of NIZKs that uses soundness, namely the construction of a digital signature scheme from [BG90]. We find that the type of soundness needed is SND-P. SND-E does not appear to suffice. And signatures do not seem like an anomaly in this regard; indeed it seems that applications usually

```
Game $\mathbf{G}^{\mathrm{uf}}_{\mathsf{DS},\lambda}$

INIT():
  $(sk, vk) \leftarrow\$ \mathsf{DS.K}(1^\lambda)$ ; Return $vk$
SIGN($m$):
  $\sigma \leftarrow\$ \mathsf{DS.S}(1^\lambda, sk, vk, m)$ ; $S \leftarrow S \cup \{m\}$ ; Return $\sigma$
FIN($m, \sigma$):
  vf $\leftarrow \mathsf{DS.V}(1^\lambda, vk, m, \sigma)$ ; return ( vf and $(m \notin S)$ )
```

```
Game $\mathbf{G}^{\mathrm{uf}}_{\mathsf{F},\lambda}$

INIT():
  $K \leftarrow\$ \{0,1\}^\lambda$
FN($m$):
  $S \leftarrow S \cup \{m\}$ ; Return $\mathsf{F}(1^\lambda, K, m)$
FIN($m, T$):
  vf $\leftarrow (\mathsf{F}(1^\lambda, K, m) = T)$ ; return (vf and $(m \notin S)$)
```

```
Game $\mathbf{G}^{\mathrm{bind}}_{\mathsf{CS},\lambda}$

INIT():
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; Return $cp$
FIN($K, d, K', d'$):
  If $(K = K')$ then return false
  Return $(\mathsf{CS.C}(1^\lambda, cp, K', d') = \mathsf{CS.C}(1^\lambda, cp, K, d))$
```

```
Game $\mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}$

INIT():
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; $b \leftarrow\$ \{0,1\}$ ; Return $cp$
CMT($K_0, K_1$):
  $d \leftarrow\$ \{0,1\}^\lambda$ ; $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K_b, d)$ ; return $c$
FIN($b'$):
  Return $(b' = b)$
```

Figure 10: Top Left: Game defining UF security of a signature scheme DS. Top Right: Game defining UF security for MAC F. Bottom: Games defining BIND and HIDE security for commitment scheme CS.

require SND-P, not SND-E. This makes the transference position for these two more of a concern.

We slightly strengthen the results of [BG90]. While they relied on statistical soundness, we only assume computational (else the question of SND-P versus SND-E is moot due to Proposition 4.2), and we use a MAC rather than a PRF. We also generalize the underlying NIZK proof system to be in the designated-prover model. We can then capture the original CRS model setting by requiring the proving key k_{P} to be the empty string.

DIGITAL SIGNATURES. A (digital) signature scheme DS specifies PT algorithms for key-generation, signing and verifying, as follows. Via $(sk, vk) \leftarrow\$ \mathsf{DS.K}(1^\lambda)$, the signer generates a secret signing key sk and public verification key vk. Via $\sigma \leftarrow\$ \mathsf{DS.S}(1^\lambda, sk, vk, m)$, the signer generates a signature of a message $m \in \{0,1\}^*$. Via vf $\leftarrow \mathsf{DS.V}(1^\lambda, vk, m, \sigma)$, the verifier deterministically generates a boolean decision as to the validity of σ. Correctness requires that $\mathsf{DS.V}(1^\lambda, vk, m, \sigma) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $\sigma \in [\mathsf{DS.S}(1^\lambda, sk, vk, m)]$, all $(sk, vk) \in [\mathsf{DS.K}(1^\lambda)]$ and all $m \in \{0,1\}^*$.

The security metric is unforgeability (UF) [GMR88]. The game is in Figure 10. We let $\mathbf{Adv}^{\mathrm{uf}}_{\mathsf{DS},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\mathrm{uf}}_{\mathsf{DS},\lambda}(\mathrm{A})]$ be the uf-advantage of adversary A. A signature scheme DS is said to be UF-secure if for al PT adversaries A, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{uf}}_{\mathsf{DS},\lambda}(\mathrm{A})$ is negligible.

BUILDING BLOCKS. We give definitions for the building blocks we need, namely MACs and commitment schemes.

A MAC is a PT deterministic algorithm F that takes 1^λ, a key $K \in \{0,1\}^\lambda$ and an input $m \in \{0,1\}^*$ to return an output $\mathsf{F}(1^\lambda, K, m) \in \{0,1\}^*$. The security metric is again unforgeability: A MAC is the symmetric analogue of a signature scheme. Consider the game $\mathbf{G}^{\mathrm{uf}}_{\mathsf{F}}$ in Figure 10. The uf-advantage of adversary A is $\mathbf{Adv}^{\mathrm{uf}}_{\mathsf{F},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\mathrm{uf}}_{\mathsf{F},\lambda}(\mathrm{A})]$. We say that F is UF-secure if for all PT adversaries A, the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{uf}}_{\mathsf{F},\lambda}(\mathrm{A})$ is negligible.

A commitment scheme CS specifies a PT parameter generation algorithm $\mathsf{CS.P}$ and a deterministic commitment algorithm $\mathsf{CS.C}$ such that $\mathsf{CS.C}(1^\lambda, cp, \cdot, \cdot)\colon \{0,1\}^* \times \{0,1\}^\lambda \to \{0,1\}^*$ for all $\lambda \in \mathbb{N}$ and for every $cp \in [\mathsf{CS.P}(1^\lambda)]$. The scheme is set up by generating parameters $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$. Then via $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$, one generates a commitment to string $K \in \{0,1\}^*$ with randomly-chosen de-commitment key $d \leftarrow\$ \{0,1\}^\lambda$. The bind advantage of an adversary A is

```
Relation $\mathsf{R}(\mathrm{crs}, (1^\lambda, cp, c, Y, m), (K, d))$:
  Return $(\mathsf{CS.C}(1^\lambda, cp, K, d) = c)$ and $(Y = \mathsf{F}(1^\lambda, K, m))$
Key-generation algorithm $\mathsf{DS.K}(1^\lambda)$:
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; $K \leftarrow\$ \{0,1\}^\lambda$ ; $d \leftarrow\$ \{0,1\}^\lambda$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$ ; $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, \varepsilon) \leftarrow\$ \mathsf{\Pi.C}(1^\lambda)$
  $sk \leftarrow (K, d, k_{\mathsf{P}})$ ; $vk \leftarrow (\mathrm{crs}, cp, c)$ ; return $(sk, vk)$
Signing algorithm $\mathsf{DS.S}(1^\lambda, sk, vk, m)$:
  $(\mathrm{crs}, cp, c) \leftarrow vk$ ; $(K, d, k_{\mathsf{P}}) \leftarrow sk$ ; $Y \leftarrow \mathsf{F}(1^\lambda, K, m)$
  $x \leftarrow (1^\lambda, cp, c, Y, m)$ ; $\mathrm{pf} \leftarrow\$ \mathsf{\Pi.P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, (K, d))$
  Return $(Y, \mathrm{pf})$
Verifying algorithm $\mathsf{DS.V}(1^\lambda, vk, m, \sigma)$:
  $(\mathrm{crs}, cp, c) \leftarrow vk$ ; $(Y, \mathrm{pf}) \leftarrow \sigma$ ; $x \leftarrow (1^\lambda, cp, c, Y, m)$
  Return $\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf})$
```

```
Adversary $\mathrm{A}_{\mathrm{sndp}}$:
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; $K \leftarrow\$ \{0,1\}^\lambda$
  $d \leftarrow\$ \{0,1\}^\lambda$ ; $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$
  $(\mathrm{crs}, k_{\mathsf{P}}) \leftarrow\$ \mathbf{G}^{\mathrm{snd\text{-}p}}_{\mathsf{\Pi},\mathsf{R},\lambda}.\textsc{Init}$ ; $vk \leftarrow (\mathrm{crs}, cp, c)$
  $\mathrm{A}_{\mathsf{DS}}^{\textsc{Init},\textsc{Sign},\textsc{Fin}}$
$\textsc{Init}$:
  Return $vk$
$\textsc{Sign}(m)$:
  $Y \leftarrow \mathsf{F}(K, m)$ ; $x \leftarrow (1^\lambda, cp, c, Y, m)$
  $\mathrm{pf} \leftarrow\$ \mathsf{\Pi.P}(1^\lambda, \mathrm{crs}, k_{\mathsf{P}}, x, (K, d))$
  Return $(Y, \mathrm{pf})$
$\textsc{Fin}(m, \sigma)$:
  $(Y, \mathrm{pf}) \leftarrow \sigma$ ; $\mathbf{G}^{\mathrm{snd\text{-}p}}_{\mathsf{\Pi},\mathsf{R},\lambda}.\textsc{Vf}((1^\lambda, cp, c, Y, m), \mathrm{pf})$
  $\mathbf{G}^{\mathrm{snd\text{-}p}}_{\mathsf{\Pi},\mathsf{R},\lambda}.\textsc{Fin}()$
```

Figure 11: Left: Signature scheme DS. Right: SND-P adversary for Theorem 6.1.

$\mathbf{Adv}^{\mathrm{bind}}_{\mathsf{CS},\lambda}(\mathrm{A}) = \Pr[\mathbf{G}^{\mathrm{bind}}_{\mathsf{CS},\lambda}(\mathrm{A})]$, where the game is in Figure 10. It is required that the d, d' queried by the adversary to $\textsc{Fin}$ are in $\{0,1\}^\lambda$. We require perfect binding, namely that $\mathbf{Adv}^{\mathrm{bind}}_{\mathsf{CS},\lambda}(\mathrm{A}) = 0$ for all adversaries A, regardless of their running time. The hide advantage of an adversary A is $\mathbf{Adv}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(\mathrm{A}) = 2\Pr[\mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(\mathrm{A})] - 1$, where the game is in Figure 10. We say that CS is hiding if for all PT A the function $\lambda \mapsto \mathbf{Adv}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(\mathrm{A})$ is negligible. (The hiding requirement is computational.)

CONSTRUCTION. With MAC F and commitment scheme CS as above, let R be the claim validator of Figure 11. Then let Π be a proof system that satisfies completeness, SND-P for R and ZK for R, and let DS be the signature scheme whose algorithms are shown in Figure 11. Theorem 6.1 says that DS is UF secure.

Theorem 6.1 *Let F be a UF-secure MAC. Let CS be a commitment scheme that is perfectly binding and (computationally) hiding. Let R be the relation of Figure 11. Let Π be a proof system that is SND-P and ZK for R. Let DS be the signature scheme whose algorithms are shown in Figure 11. Then DS is UF-secure.*

Proof of Theorem 6.1: Let A_{DS} be a polynomial time adversary. Then we construct PT adversaries $\mathrm{A}_{\mathrm{sndp}}, \mathrm{A}_{\mathrm{zk}}, \mathrm{A}_{\mathrm{hide}}, \mathrm{A}_{\mathrm{uf}}$ (shown explicitly in Figures 11 and 13) such that

$$\mathbf{Adv}^{\mathrm{uf}}_{\mathsf{DS},\lambda}(\mathrm{A}_{\mathsf{DS}}) \le \mathbf{Adv}^{\mathrm{snd\text{-}p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(\mathrm{A}_{\mathrm{sndp}}) + \mathbf{Adv}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}_{\mathrm{zk}}) + \mathbf{Adv}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(\mathrm{A}_{\mathrm{hide}}) + \mathbf{Adv}^{\mathrm{uf}}_{\mathsf{F}}(\mathrm{A}_{\mathrm{uf}}) .$$

The theorem then follows.

Consider the games of Figure 12. Games $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$ differ only in one line, the ones used, respectively, in these games, being lines 8,9,10, which change how the game decides whether or not the adversary's forgery attempt should let it win the game. We have

$$\begin{aligned}\mathbf{Adv}^{\mathrm{uf}}_{\mathsf{DS}}(\mathrm{A}_{\mathsf{DS}}) &= \Pr[\mathrm{G}_0(\mathrm{A}_{\mathsf{DS}})] \\ &= \Pr[\mathrm{G}_1(\mathrm{A}_{\mathsf{DS}})] + (\Pr[\mathrm{G}_0(\mathrm{A}_{\mathsf{DS}})] - \Pr[\mathrm{G}_1(\mathrm{A}_{\mathsf{DS}})]) .\end{aligned}$$

```
Games $G_0, G_1, G_2$
Init():
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; $K \leftarrow\$ \{0,1\}^\lambda$ ; $d \leftarrow\$ \{0,1\}^\lambda$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$ ; $(\mathrm{crs}, \mathrm{td}, \varepsilon, \varepsilon) \leftarrow\$ \mathsf{\Pi.C}(1^\lambda)$
  $vk \leftarrow (\mathrm{crs}, cp, c)$ ; return $vk$
Sign($m$):
  $Y \leftarrow \mathsf{F}(1^\lambda, K, m)$
  $\mathrm{pf} \leftarrow\$ \mathsf{\Pi.P}(1^\lambda, \mathrm{crs}, \varepsilon, (1^\lambda, cp, c, Y, m), (K, d))$
  $\sigma \leftarrow (Y, \mathrm{pf})$ ; $S \leftarrow S \cup \{m\}$ ; Return $\sigma$
Fin($m, \sigma$):
  $(Y, \mathrm{pf}) \leftarrow \sigma$
  If $(m \in S)$ then return false
  $\mathrm{vf} \leftarrow \mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, (1^\lambda, cp, c, Y, m), \mathrm{pf})$   // Game $G_0$
  $\mathrm{vf} \leftarrow \exists (K', d') : (\mathsf{CS.C}(1^\lambda, cp, K', d') = c) \wedge (\mathsf{F}(1^\lambda, K', m) = Y)$
    // Game $G_1$
  $\mathrm{vf} \leftarrow (\mathsf{F}(1^\lambda, K, m) = Y)$   // Game $G_2$
  Return vf

Games $G_3, G_4$
Init():
  $cp \leftarrow\$ \mathsf{CS.P}(1^\lambda)$ ; $K, K' \leftarrow\$ \{0,1\}^\lambda$
  $d \leftarrow\$ \{0,1\}^\lambda$ ; $(\mathrm{crs}, \mathrm{td}, \varepsilon) \leftarrow\$ \mathsf{S.C}(1^\lambda)$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$   // Game $G_3$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K', d)$   // Game $G_4$
  $vk \leftarrow (\mathrm{crs}, cp, c)$ ; return $vk$
Sign($m$):
  $Y \leftarrow \mathsf{F}(1^\lambda, K, m)$
  $\mathrm{pf} \leftarrow\$ \mathsf{S.P}(1^\lambda, \mathrm{crs}, \mathrm{td}, \varepsilon, (1^\lambda, cp, c, Y, m))$
  $\sigma \leftarrow (Y, \mathrm{pf})$ ; $S \leftarrow S \cup \{m\}$ ; Return $\sigma$
Fin($m, \sigma$):
  $(Y, \mathrm{pf}) \leftarrow \sigma$
  If $(m \in S)$ then return false
  $\mathrm{vf} \leftarrow (\mathsf{F}(1^\lambda, K, m) = Y)$ ; Return vf
```

Figure 12: Games for proof of Theorem 6.1.

We claim to have designed the Figure 11 adversary A_{sndp} so that

$$\Pr[G_0(A_{\mathsf{DS}})] - \Pr[G_1(A_{\mathsf{DS}})] \leq \mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(A_{\mathrm{sndp}}) \, .$$

Notice first that the condition setting vf in G_1 (line 8) is exactly the test for membership in $\mathsf{L}_{\mathsf{R}}(\mathrm{crs})$. For brevity, we let x denote the $(1^\lambda, cp, c, Y, m)$ tuple corresponding to the (m, σ) pair queried to the Fin procedure by the adversary A_{DS} in games G_0 and G_1. Then

$$\Pr[G_0(A_{\mathsf{DS}})] = \Pr[\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf})] \text{ and } \Pr[G_1(A_{\mathsf{DS}})] = \Pr[x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs})] \, .$$

This gives us (let $\chi = \Pr[G_0(A_{\mathsf{DS}})] - \Pr[G_1(A_{\mathsf{DS}})]$)

$$\begin{aligned}\chi &= \Pr[\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf})] - \Pr[x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs})] \\ &= \Pr[\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) \ \wedge \ (x \notin \mathsf{L}_{\mathsf{R}}(\mathrm{crs}))] \\ &\quad + \Pr[\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) \ \wedge \ (x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs}))] - \Pr[(x \in \mathsf{L}_{\mathsf{R}}(\mathrm{crs}))] \\ &\leq \Pr[\mathsf{\Pi.V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) \ \wedge \ (x \notin \mathsf{L}_{\mathsf{R}}(\mathrm{crs}))] \\ &= \mathbf{Adv}^{\text{snd-p}}_{\mathsf{\Pi},\mathsf{R},\lambda}(A_{\mathrm{sndp}})\end{aligned}$$

We now show by explicitly providing the adversaries A_{zk}, A_{hide}, and A_{uf}, that

$$\Pr[G_1(A_{\mathsf{DS}})] \leq \mathbf{Adv}^{\mathrm{zk}}_{\mathsf{\Pi},\mathsf{R},\mathsf{S},\lambda}(A_{\mathrm{zk}}) + \mathbf{Adv}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(A_{\mathrm{hide}}) + \mathbf{Adv}^{\mathrm{uf}}_{\mathsf{F}}(A_{\mathrm{uf}}) \, . \tag{2}$$

We have

$$\Pr[G_1(A_{\mathsf{DS}})] = \Pr[G_2(A_{\mathsf{DS}})] + (\Pr[G_1(A_{\mathsf{DS}})] - \Pr[G_2(A_{\mathsf{DS}})]) \, .$$

The assumption that CS is perfectly binding implies that

$$\Pr[G_1(A_{\mathsf{DS}})] = \Pr[G_2(A_{\mathsf{DS}})] \, .$$

```
Adversary $\mathrm{A}_{\mathrm{zk}}$:
  $cp \leftarrow\!\!\$\ \mathsf{CS.P}(1^\lambda)$ ; $K \leftarrow\!\!\$\ \{0,1\}^\lambda$ ; $d \leftarrow\!\!\$\ \{0,1\}^\lambda$ ; $(\mathrm{crs}, \varepsilon) \leftarrow\!\!\$\ \mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}.\textsc{Init}$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K, d)$ ; $vk \leftarrow (\mathrm{crs}, cp, c)$ ; $\mathrm{A}_{\mathrm{ds}}^{\textsc{Init},\textsc{Sign},\textsc{Fin}}$
$\textsc{Init}$:
  Return $vk$
$\textsc{Sign}(m)$:
  $Y \leftarrow \mathsf{F}(1^\lambda, K, m)$ ; $\mathrm{pf} \leftarrow\!\!\$\ \mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}.\textsc{Pf}((1^\lambda, cp, c, Y, m), (K, d))$
  $\sigma \leftarrow (Y, \mathrm{pf})$ ; $S \leftarrow S \cup \{m\}$ ; Return $\sigma$
$\textsc{Fin}(m, \sigma)$:
  $(Y, \mathrm{pf}) \leftarrow \sigma$
  If $(m \in S)$ then $\mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}.\textsc{Fin}(0)$
  If $(\mathsf{F}(1^\lambda, K, m) = Y)$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
  $\mathbf{G}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}.\textsc{Fin}(b')$
```

```
Adversary $\mathrm{A}_{\mathrm{hide}}$:
  $cp \leftarrow\!\!\$\ \mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}.\textsc{Init}$ ; $K, K' \leftarrow\!\!\$\ \{0,1\}^\lambda$
  $c \leftarrow\!\!\$\ \mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}.\mathrm{CMT}(K, K')$ ; $(\mathrm{crs}, \mathrm{td}, \varepsilon) \leftarrow\!\!\$\ \mathsf{S.C}(1^\lambda)$
  $vk \leftarrow (\mathrm{crs}, cp, c)$ ; $\mathrm{A}_{\mathrm{ds}}^{\textsc{Init},\textsc{Sign},\textsc{Fin}}$
$\textsc{Init}$:
  Return $vk$
$\textsc{Sign}(m)$:
  $S \leftarrow S \cup \{m\}$ ; $Y \leftarrow \mathsf{F}(1^\lambda, K, m)$ ; $x \leftarrow (1^\lambda, cp, c, Y, m)$
  $\mathrm{pf} \leftarrow\!\!\$\ \mathsf{S.P}(1^\lambda, \mathrm{crs}, \mathrm{td}, \varepsilon, x)$ ; $\sigma \leftarrow (Y, \mathrm{pf})$ ; return $\sigma$
$\textsc{Fin}(m, \sigma)$:
  $(Y, \mathrm{pf}) \leftarrow \sigma$
  If $(m \in S)$ then $\mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}.\textsc{Fin}(0)$
  If $((\mathsf{F}(1^\lambda, K, m) = Y))$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
  $\mathbf{G}^{\mathrm{hide}}_{\mathsf{CS},\lambda}.\textsc{Fin}(b')$
```

```
Adversary $\mathrm{A}_{\mathrm{uf}}$:
  $cp \leftarrow\!\!\$\ \mathsf{CS.P}(1^\lambda)$ ; $K' \leftarrow\!\!\$\ \{0,1\}^\lambda$
  $d \leftarrow\!\!\$\ \{0,1\}^\lambda$ ; $(\mathrm{crs}, \mathrm{td}, \varepsilon) \leftarrow\!\!\$\ \mathsf{S.C}(1^\lambda)$
  $c \leftarrow \mathsf{CS.C}(1^\lambda, cp, K', d)$ ; $vk \leftarrow (\mathrm{crs}, cp, c)$
  $\mathbf{G}^{\mathrm{uf}}_{\mathsf{F},\lambda}.\textsc{Init}$ ; $\mathrm{A}_{\mathrm{ds}}^{\textsc{Init},\textsc{Sign},\textsc{Fin}}$
$\textsc{Init}$:
  Return $vk$
$\textsc{Sign}(m)$:
  $Y \leftarrow \mathbf{G}^{\mathrm{uf}}_{\mathsf{F},\lambda}.\mathrm{FN}(m)$ ; $x \leftarrow (1^\lambda, cp, c, Y, m)$
  $\mathrm{pf} \leftarrow\!\!\$\ \mathsf{S.P}(1^\lambda, \mathrm{crs}, \mathrm{td}, \varepsilon, x)$ ; return $(Y, \mathrm{pf})$
$\textsc{Fin}(m, \sigma)$:
  $(Y, \mathrm{pf}) \leftarrow \sigma$ ; $\mathbf{G}^{\mathrm{uf}}_{\mathsf{F},\lambda}.\textsc{Fin}(m, Y)$
```

Figure 13: Adversaries for Theorem 6.1.

Next we have

$$\Pr[\mathrm{G}_2(\mathrm{A}_{\mathsf{DS}})] = \Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})] + (\Pr[\mathrm{G}_2(\mathrm{A}_{\mathsf{DS}})] - \Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})])\ .$$

We claim to have designed the Figure 13 adversary A_{zk} so that

$$\Pr[\mathrm{G}_2(\mathrm{A}_{\mathsf{DS}})] - \Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})] \le \mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{S},\lambda}(\mathrm{A}_{\mathrm{zk}})\ .$$

Next we have

$$\Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})] = \Pr[\mathrm{G}_4(\mathrm{A}_{\mathsf{DS}})] + (\Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})] - \Pr[\mathrm{G}_4(\mathrm{A}_{\mathsf{DS}})])\ .$$

We claim to have designed the Figure 13 adversaries $\mathrm{A}_{\mathrm{hide}}$ and A_{uf} so that

$$\Pr[\mathrm{G}_3(\mathrm{A}_{\mathsf{DS}})] - \Pr[\mathrm{G}_4(\mathrm{A}_{\mathsf{DS}})] \le \mathbf{Adv}^{\mathrm{hide}}_{\mathsf{CS},\lambda}(\mathrm{A}_{\mathrm{hide}})\ ,$$

$$\Pr[\mathrm{G}_4(\mathrm{A}_{\mathsf{DS}})] \le \mathbf{Adv}^{\mathrm{uf}}_{\mathsf{F}}(\mathrm{A}_{\mathrm{uf}})\ .$$

Putting these together gives us the inequality (2). This completes the proof. ▌

The question that we now discuss is whether the SND-E soundness notion will suffice for this application. Recall that SND-E requires adversaries to be membership-conscious. However, in a reduction from an adversary to an adversary against SND-E, there is no clear way to ensure that membership-consciousness holds, which indicates that SND-E might not be sufficient for this application.

References

[AFH$^+$16] Martin R. Albrecht, Pooya Farshim, Dennis Hofheinz, Enrique Larraia, and Kenneth G. Paterson. Multilinear maps from obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 446–473. Springer, Heidelberg, January 2016. (Cited on page 3, 4, 9, 30, 31.)

[BCG$^+$19] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 489–518. Springer, Heidelberg, August 2019. (Cited on page 4.)

[BCGI18] Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 896–912. ACM Press, October 2018. (Cited on page 4.)

[BDSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM Journal on Computing*, 20(6):1084–1118, 1991. (Cited on page 3, 4, 7.)

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988. (Cited on page 3, 4, 7.)

[BFM90] Manuel Blum, Paul Feldman, and Silvio Micali. Proving security against chosen cyphertext attacks. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 256–268. Springer, Heidelberg, August 1990. (Cited on page 29.)

[BG90] Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 194–211. Springer, Heidelberg, August 1990. (Cited on page 4, 5, 19, 20.)

[BG93] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 390–420. Springer, Heidelberg, August 1993. (Cited on page 19.)

[BHK15] Mihir Bellare, Dennis Hofheinz, and Eike Kiltz. Subtleties in the definition of IND-CCA: When and how should challenge decryption be disallowed? *Journal of Cryptology*, 28(1):29–48, January 2015. (Cited on page 5, 9.)

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. (Cited on page 6.)

[CC18] Pyrros Chaidos and Geoffroy Couteau. Efficient designated-verifier non-interactive zero-knowledge proofs of knowledge. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 193–221. Springer, Heidelberg, April / May 2018. (Cited on page 19.)

[CCH$^+$19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019. (Cited on page 30, 31.)

[CDG$^+$17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1825–1842. ACM Press, October / November 2017. (Cited on page 4, 5.)

[Dam00] Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 418–430. Springer, Heidelberg, May 2000. (Cited on page 7, 29.)

[DFN06] Ivan Damgård, Nelly Fazio, and Antonio Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 41–59. Springer, Heidelberg, March 2006. (Cited on page 4, 7, 29.)

[DMP90] Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Non-interactive zero-knowledge with preprocessing. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 269–282. Springer, Heidelberg, August 1990. (Cited on page 7, 29.)

[DP92] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction (extended abstract). In *33rd FOCS*, pages 427–436. IEEE Computer Society Press, October 1992. (Cited on page 19.)

[ES02] Edith Elkind and Amit Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/042, 2002. http://eprint.iacr.org/2002/042. (Cited on page 4, 7, 8, 29.)

[FF00] Marc Fischlin and Roger Fischlin. Efficient non-malleable commitment schemes. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 413–431. Springer, Heidelberg, August 2000. (Cited on page 7, 29.)

[FHHL18] Pooya Farshim, Julia Hesse, Dennis Hofheinz, and Enrique Larraia. Graded encoding schemes from obfuscation. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 371–400. Springer, Heidelberg, March 2018. (Cited on page 31.)

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd ACM STOC*, pages 416–426. ACM Press, May 1990. (Cited on page 18.)

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988. (Cited on page 20.)

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. (Cited on page 19.)

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994. (Cited on page 29.)

[GOS06a] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 97–111. Springer, Heidelberg, August 2006. (Cited on page 3, 30, 31.)

[GOS06b] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 339–358. Springer, Heidelberg, May / June 2006. (Cited on page 3, 6, 30, 31.)

[GOS12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *Journal of the ACM (JACM)*, 59(3):1–35, 2012. (Cited on page 3, 4.)

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, December 2006. (Cited on page 4, 7, 30, 31.)

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, April 2008. (Cited on page 30, 31.)

[HHNR17] Gunnar Hartung, Max Hoffmann, Matthias Nagel, and Andy Rupp. BBA+: Improving the security and applicability of privacy-preserving point collection. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1925–1942. ACM Press, October / November 2017. (Cited on page 30, 31.)

[HU19] Dennis Hofheinz and Bogdan Ursu. Dual-mode NIZKs from obfuscation. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 311–341. Springer, Heidelberg, December 2019. (Cited on page 3, 4, 9, 30, 31.)

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 143–154. Springer, Heidelberg, May 1996. (Cited on page 29.)

[KNYY19] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Designated verifier/prover and preprocessing NIZKs from Diffie-Hellman assumptions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 622–651. Springer, Heidelberg, May 2019. (Cited on page 4, 7, 8, 29.)

[KW18] Sam Kim and David J. Wu. Multi-theorem preprocessing NIZKs from lattices. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 733–765. Springer, Heidelberg, August 2018. (Cited on page 4, 7, 8, 29.)

[LPWW20] Benoît Libert, Alain Passelègue, Hoeteck Wee, and David J. Wu. New constructions of statistical NIZKs: Dual-mode DV-NIZKs and more. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 410–441. Springer, Heidelberg, May 2020. (Cited on page 3, 4, 8, 9, 30, 31.)

[LQR$^+$19] Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier NIZKs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 670–700. Springer, Heidelberg, August 2019. (Cited on page 8.)

[Ps05] Rafael Pass and Abhi shelat. Unconditional characterizations of non-interactive zero-knowledge. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 118–134. Springer, Heidelberg, August 2005. (Cited on page 7, 29.)

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Heidelberg, August 2019. (Cited on page 30, 31.)

[PsV06] Rafael Pass, abhi shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 271–289. Springer, Heidelberg, August 2006. (Cited on page 4, 7, 8, 29.)

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008. (Cited on page 29.)

[QRW19] Willy Quach, Ron D. Rothblum, and Daniel Wichs. Reusable designated-verifier NIZKs for all NP from CDH. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 593–621. Springer, Heidelberg, May 2019. (Cited on page 8.)

```
Relation $\mathsf{R}((p, \mathbb{G}, g, A, B, C), X, w)$:
 1  Return $(g^w = A) \wedge (X \neq B^w) \wedge (X \in \mathbb{G})$

$\Pi.\mathsf{C}(1^\lambda)$:
 1  $(p, \mathbb{G}, g) \leftarrow\!\!\$\ \mathsf{GG}(1^\lambda)$ ; $\mu \leftarrow\!\!\$\ \{0,1\}$ ; $a, b \leftarrow\!\!\$\ \mathbb{Z}_p$
 2  If $(\mu = 0)$ then $c \leftarrow\!\!\$\ \mathbb{Z}_p$ else $c \leftarrow ab \mod p$
 3  $A \leftarrow g^a$ ; $B \leftarrow g^b$ ; $C \leftarrow g^c$ ; $\mathrm{crs} \leftarrow (p, \mathbb{G}, g, A, B, C)$
 4  $\mathrm{td} \leftarrow \varepsilon$ ; $k_{\mathsf{P}} \leftarrow \varepsilon$ ; $k_{\mathsf{V}} \leftarrow \varepsilon$ ; return $(\mathrm{crs}, \mathrm{td}, k_{\mathsf{P}}, k_{\mathsf{V}})$
$\Pi.\mathsf{P}(1^\lambda, (p, \mathbb{G}, g, A, B, C), k_{\mathsf{P}}, X, w)$:
 5  Return $\varepsilon$

$\Pi.\mathsf{V}(1^\lambda, (p, \mathbb{G}, g, A, B, C), k_{\mathsf{V}}, X, \mathrm{pf})$:
 6  If $(X \notin \mathbb{G})$ then return false
 7  Return true

Adversary $\mathrm{A}_{\text{snd-p}}$:
 1  $(\mathrm{crs}, k_{\mathsf{P}}) \leftarrow\!\!\$\ \mathbf{G}^{\text{snd-p}}_{\Pi,\mathsf{R},\lambda}.\textsc{Init}$
 2  $(p, \mathbb{G}, g, A, B, C) \leftarrow \mathrm{crs}$
 3  $\mathbf{G}^{\text{snd-p}}_{\Pi,\mathsf{R},\lambda}.\textsc{Vf}(C, \varepsilon)$ ; $\mathbf{G}^{\text{snd-p}}_{\Pi,\mathsf{R},\lambda}.\textsc{Fin}$
```

```
Games $\mathrm{G}_0$ - $\mathrm{G}_8$

$\textsc{Init}()$:
 1  $(p, \mathbb{G}, g) \leftarrow\!\!\$\ \mathsf{GG}(1^\lambda)$
 2  $\mu \leftarrow\!\!\$\ \{0,1\}$ ; $a, b \leftarrow\!\!\$\ \mathbb{Z}_p$
 3  If $(\mu = 0)$ then $c \leftarrow\!\!\$\ \mathbb{Z}_p$
 4  Else $c \leftarrow ab \mod p$
 5  $A \leftarrow g^a$ ; $B \leftarrow g^b$ ; $C \leftarrow g^c$
 6  Return $((p, \mathbb{G}, g, A, B, C), \varepsilon)$

$\textsc{Vf}(X, \mathrm{pf})$:
 8  If $(X \in \mathbb{G} \setminus \{g^{ab}\})$ then bad $\leftarrow$ true
 9  $\mathrm{vf} \leftarrow (X \in \mathbb{G})$ ; $u_1 \leftarrow (X = C)$ ; $u_2 \leftarrow (\mu = 0)$
10  If vf then win $\leftarrow$ true   // $\mathrm{G}_0$
11  If $(\mathrm{vf} \wedge u_1 \wedge u_2)$ then win $\leftarrow$ true   // $\mathrm{G}_1$
12  If $(\mathrm{vf} \wedge u_1 \wedge \neg u_2)$ then win $\leftarrow$ true   // $\mathrm{G}_2$
13  If $(\mathrm{vf} \wedge \neg u_1 \wedge u_2)$ then win $\leftarrow$ true   // $\mathrm{G}_3$
14  If $(\mathrm{vf} \wedge \neg u_1 \wedge \neg u_2)$ then win $\leftarrow$ true   // $\mathrm{G}_4$
15  If $(\mathrm{vf} \wedge u_1 \wedge u_2 \wedge \neg\mathsf{bad})$ then win $\leftarrow$ true   // $\mathrm{G}_5$
16  If $(\mathrm{vf} \wedge u_1 \wedge u_2 \wedge \mathsf{bad})$ then win $\leftarrow$ true   // $\mathrm{G}_6$
17  If $(\mathrm{vf} \wedge \neg u_1 \wedge u_2 \wedge \neg\mathsf{bad})$ then win $\leftarrow$ true   // $\mathrm{G}_7$
18  If $(\mathrm{vf} \wedge \neg u_1 \wedge u_2 \wedge \mathsf{bad})$ then win $\leftarrow$ true   // $\mathrm{G}_8$
19  If vf then return true
20  Return false

$\textsc{Fin}()$:
 7  Return win
```

Figure 14: Top: Relation R, proof system Π, and adversary $\mathrm{A}_{\text{snd-p}}$ for the proof of Proposition 4.3. Bottom: Games for the proof of Proposition 4.3.

```
Adversary $A^1_{\text{ddh}}$:
1 $(p, \mathbb{G}, g, A, B, C) \leftarrow\$ \mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{INIT}$ ; $A^{\text{INIT},\text{VF},\text{FIN}}$
INIT():
2 Return $((p, \mathbb{G}, g, A, B, C), \varepsilon)$
VF($X$, pf):
3 If $((X \in \mathbb{G}) \wedge (X = C))$ then win $\leftarrow$ true
4 Return $(X \in \mathbb{G})$
FIN():
5 If win then $\mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{FIN}(1)$ else $\mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{FIN}(0)$
```

```
Adversary $A^2_{\text{ddh}}$:
1 $(p, \mathbb{G}, g, A, B, C) \leftarrow\$ \mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{INIT}$ ; $A^{\text{INIT},\text{VF},\text{FIN}}$
INIT():
2 Return $((p, \mathbb{G}, g, A, B, C), \varepsilon)$
VF($X$, pf):
3 If $((X \in \mathbb{G}) \wedge (X \neq C))$ then win $\leftarrow$ true
4 Return $(X \in \mathbb{G})$
FIN():
5 If win then $\mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{FIN}(1)$ else $\mathbf{G}^{\text{ddh}}_{\mathsf{GG},\lambda}.\text{FIN}(0)$
```

```
Adversary $A^1_{\text{cdh}}$:
1 $(p, \mathbb{G}, g, A, B) \leftarrow\$ \mathbf{G}^{\text{cdh}}_{\mathsf{GG},\lambda}.\text{INIT}$
2 $c \leftarrow\$ \mathbb{Z}_p$ ; $C \leftarrow g^c$ ; $A^{\text{INIT},\text{VF},\text{FIN}}$
INIT():
3 Return $((p, \mathbb{G}, g, A, B, C), \varepsilon)$
VF($X$, pf):
4 If $((X \in \mathbb{G}) \wedge (X = C))$ then $S \leftarrow S \cup \{X\}$
5 Return $(X \in \mathbb{G})$
FIN():
6 $X \leftarrow\$ S$ ; $\mathbf{G}^{\text{cdh}}_{\mathsf{GG},\lambda}.\text{FIN}(X)$
```

```
Adversary $A^2_{\text{cdh}}$:
1 $(p, \mathbb{G}, g, A, B) \leftarrow\$ \mathbf{G}^{\text{cdh}}_{\mathsf{GG},\lambda}.\text{INIT}$
2 $c \leftarrow\$ \mathbb{Z}_p$ ; $C \leftarrow g^c$ ; $A^{\text{INIT},\text{VF},\text{FIN}}$
INIT():
3 Return $((p, \mathbb{G}, g, A, B, C), \varepsilon)$
VF($X$, pf):
4 If $((X \in \mathbb{G}) \wedge (X \neq C))$ then $S \leftarrow S \cup \{X\}$
5 Return $(X \in \mathbb{G})$
FIN():
6 $X \leftarrow\$ S$ ; $\mathbf{G}^{\text{cdh}}_{\mathsf{GG},\lambda}.\text{FIN}(X)$
```

Figure 15: More adversaries for the proof of Proposition 4.3.

A Proof of Proposition 4.3

In this appendix, we provide a counterexample to prove the separation between SND-E and SND-P in the computational setting.

Proof of Proposition 4.3: Consider a proof system Π for the relation R, both of which are described in Figure 14. We show that (1) Π is SND-E for R but (2) Π is not SND-P for R, assuming DDH is hard relative to the group generator.

We provide an adversary strategy to show (2) in Figure 14. When $\mu = 1$, the adversary wins the SND-P game, and when $\mu = 0$, it wins the SND-P game with negligible probability. Therefore, we can conclude that:

$$\mathbf{Adv}^{\text{snd-p}}_{\Pi,\mathsf{R},\lambda}(A_{\text{snd-p}}) \geq \frac{1}{2}$$

We now show (1) via a sequence of games, described in Figure 14. First, notice that $\mathbf{Adv}^{\text{snd-e}}_{\Pi,\mathsf{R},\lambda}(A) = \Pr[\,G_0(A)\,]$. Further, we have

$$\Pr[\,G_0(A)\,] = \Pr[\,G_1(A)\,] + \Pr[\,G_2(A)\,] + \Pr[\,G_3(A)\,] + \Pr[\,G_4(A)\,]$$

We can construct adversaries A^1_{ddh} and A^2_{ddh} against the DDH game (described in Figure 15) such that

$$\Pr[\,G_2(A)\,] - \Pr[\,G_1(A)\,] \leq \mathbf{Adv}^{\text{ddh}}_{\mathsf{GG},\lambda}(A^1_{\text{ddh}})$$
$$\Pr[\,G_4(A)\,] - \Pr[\,G_3(A)\,] \leq \mathbf{Adv}^{\text{ddh}}_{\mathsf{GG},\lambda}(A^2_{\text{ddh}})$$

We can further write

$$\Pr[\,G_1(A)\,] = \Pr[\,G_5(A)\,] + \Pr[\,G_6(A)\,]$$
$$\Pr[\,G_3(A)\,] = \Pr[\,G_7(A)\,] + \Pr[\,G_8(A)\,]$$

Now, since the adversaries all must be membership-conscious (i.e. $A \in \mathcal{A}^{\mathrm{snde}}$), we know that

$$\Pr[\,G_6(A)\,] \leq \mathrm{negl}(\lambda) \qquad\qquad \Pr[\,G_8(A)\,] \leq \mathrm{negl}(\lambda)$$

We also construct adversaries A^1_{cdh} and A^2_{cdh} against the CDH game in Figure 15 such that

$$\Pr[\,G_5(A)\,] \leq \mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(A^1_{\mathrm{cdh}}) \qquad\qquad \Pr[\,G_7(A)\,] \leq \mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(A^2_{\mathrm{cdh}})$$

Putting all this together, we get

$$\begin{aligned}\mathbf{Adv}^{\mathrm{snd\text{-}e}}_{\mathsf{\Pi},\mathsf{R},\lambda}(A) \leq \mathbf{Adv}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(A^1_{\mathrm{ddh}}) + \mathbf{Adv}^{\mathrm{ddh}}_{\mathsf{GG},\lambda}(A^2_{\mathrm{ddh}})&\\ + 2\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(A^1_{\mathrm{cdh}}) + 2\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{GG},\lambda}(A^2_{\mathrm{cdh}}) + \mathrm{negl}(\lambda)&\end{aligned}$$

This shows that the scheme is SND-E secure assuming the hardness of the DDH and CDH assumptions relative to the group generator. ▌

B Related Work

It has been shown that non-interactive zero-knowledge proof systems for languages outside of **BPP** cannot exist in the plain model [GO94]. The existing literature instead considers a variety of different models under which they provide constructions achieving non-interactive zero knowledge.

THE COMMON REFERENCE STRING (CRS) MODEL. This widely used model is also known as the auxiliary string model [Dam00] or the public-parameter model [FF00, Ps05]. In this model, there is assumed to exist a trusted party that generates a common reference string which is available to both the prover and the verifier. The common random string model [BFM90] can be considered to be a special case of the CRS model, in which the common string generated by the trusted party is sampled from a uniform distribution.

THE PREPROCESSING MODEL [DMP90]. In this model, there is assumed to be an initial, statement-independent trusted setup (preprocessing) phase, that results in the generation of a proving key (which will be needed to generate proofs) and a verification key (which is needed to verify proofs). Soundness is now required to hold even against a prover with oracle access to the verifier (but no access to the verification key), while zero-knowledge is required to hold against a verifier with oracle access to the prover (but no access to the proving key).

THE DESIGNATED VERIFIER AND DESIGNATED PROVER MODELS. The designated verifier (DV) model (introduced in [JSI96] for interactive proofs and in [ES02, PsV06, DFN06] for non-interactive proofs) and the designated prover (DP) model [KW18, KNYY19] can be studied as special cases of the preprocessing model. In the DV model, the proving key is considered to be empty, so that any party can generate a proof of a statement, but verification can only be done by the party with the secret verification key. Analogously, in the DP model, the verification key is considered to be empty, so that any party can verify a proof of a statement, but proof generation can only be done by the party with the secret proving key.

DUAL-MODE NIZKS. The abstraction of the dual-mode cryptosystem was first considered in [PVW08] for the setting of oblivious transfer. A similar technique named "parameter switching" was used in [GOS06b] in the context of non-interactive zero-knowledge. Though this technique was used in multiple constructions [Gro06, GOS06a, GOS06b, GS08], the term "dual-mode NIZK" was first used in [AFH+16] as one of the building blocks for multilinear maps. Dual-mode NIZKs have been constructed from obfuscation in [HU19], while [LPWW20] constructs dual-mode NIZKs in the designated verifier model from different assumptions. The works of [CCH+19, PS19] also construct NIZK systems that can be used in two modes, but do not explicitly consider the dual-mode abstraction.

We now examine the definitions of soundness in papers that use the dual-mode within our notation to see whether they transfer. Let R be a relation and Π a proof system.

[GOS06b]. They call their definition non-adaptive computational soundness. It requires that for all non-uniform PT adversaries A and all $x \notin \mathsf{L}_\mathsf{R}$ (the language here does not depend on the CRS)—

$$\Pr\left[(\mathrm{crs}, \varepsilon, \varepsilon, \varepsilon) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^\lambda)\ ;\ \mathrm{pf} \leftarrow\!\!\$\; \mathrm{A}(x, \mathrm{crs}) : \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false}\right] \approx 1 .$$

The "≈ 1" above is shorthand for saying there is a negligible function ν such that the probability on the left is $\geq 1 - \nu(\lambda)$. Now there is some ambiguity in the definition, namely that it is not clear how ν is quantified. Does it depend on A? On x? The meaningful choice here is to assume the authors meant it to depend on A but not on x, so that the definition becomes that for all non-uniform PT adversaries A there exists a negligible function ν such that for all $x \notin \mathsf{L}_\mathsf{R}$ and all λ—

$$\Pr\left[(\mathrm{crs}, \varepsilon, \varepsilon, \varepsilon) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^\lambda)\ ;\ \mathrm{pf} \leftarrow\!\!\$\; \mathrm{A}(x, \mathrm{crs}) : \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{true}\right] \leq \nu(\lambda) .$$

This notion of soundness does transfer. The proof relies, however, crucially on non-uniformity; mode indistinguishability must be assumed for non-uniform adversaries. This, however, is indeed the setting of the paper, so we conclude that soundness as per [GOS06b] transfers successfully, supporting the claim made (without explicit proof) in the paper.

However, this non-adaptive definition of soundness is not well suited (too weak) for some applications, because it does not allow x to depend on the CRS. This arises for example in the application to signatures we discussed in Section 6 . Also the definition does not seem written to allow dependence of the language on the CRS, yet in their system to prove that a ciphertext encrypts a bit, the language does depend on the CRS.

[GOS06a, Gro06, GS08, HHNR17]. All of these works use equivalent definitions for (perfect) soundness for a non-interactive proof system Π for relation R. They require that for all non-uniform adversaries A and all λ—

$$\Pr\left[(\mathrm{crs}, \varepsilon, \varepsilon, \varepsilon) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^\lambda)\ ;\ (x, \mathrm{pf}) \leftarrow\!\!\$\; \mathrm{A}(\mathrm{crs}) : \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false} \text{ if } x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})\right] = 1 .$$

The computational variant would presumably be that for all non-uniform PT adversaries A there exists a negligible function ν such that for all λ—

$$\Pr\left[\begin{matrix}(\mathrm{crs}, \varepsilon, \varepsilon, \varepsilon) \leftarrow\!\!\$\; \Pi.\mathsf{C}(1^\lambda); \\ (x, \mathrm{pf}) \leftarrow\!\!\$\; \mathrm{A}(\mathrm{crs})\end{matrix} : \Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false} \text{ if } x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})\right] \geq 1 - \nu(\lambda) .$$

Again, there is some ambiguity, namely as to the meaning of the "if." We would expect that what is written after the colon, here "$\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false}$ if $x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})$," is an event in the probability space described by what precedes the colon, and in this light have trouble understanding the "if." Our best interpretation was to view the "if" as an implication. That is, "$\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false}$ if $x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})$" becomes "$(x \notin \mathsf{L}_\mathsf{R}(\mathrm{crs})) \implies (\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false})$," which in turn becomes "$(\Pi.\mathsf{V}(1^\lambda, \mathrm{crs}, \varepsilon, x, \mathrm{pf}) = \mathsf{false})$ or $(x \in \mathsf{L}_\mathsf{R}(\mathrm{crs}))$." The condition above now becomes

$$\Pr\left[\begin{array}{l}(\mathrm{crs},\varepsilon,\varepsilon,\varepsilon) \leftarrow_{\$} \mathsf{\Pi.C}(1^\lambda);\\ (x,\mathrm{pf}) \leftarrow_{\$} \mathrm{A}(\mathrm{crs})\end{array} : (\mathsf{\Pi.V}(1^\lambda,\mathrm{crs},\varepsilon,x,\mathrm{pf})=\mathsf{true}) \wedge (x \notin \mathsf{L_R}(\mathrm{crs}))\right] \leq \nu(\lambda) \ .$$

This definition is equivalent to the SND-P notion of soundness we give in Figure 3. As we have shown in Theorem 4.4, this notion of soundness need not in general transfer, so that the soundness definition of [GOS06a, Gro06, GS08, HHNR17] also fails in general to transfer. Note Theorem 4.4 holds in both the uniform and non-uniform settings; unlike for the [GOS06b] definition discussed above, non-uniformity does not seem aid transfer here. All this is with the caveat that we may be mis-interpreting the "if."

On the other hand, the soundness in this definition is adaptive, making it good for applications, as we indicate via Section 6 . The pattern we see is that weak (less application-enabling) soundness transfers and strong (more application-enabling) soundness does not transfer.

[AFH+16, FHHL18]. This work uses a definition of perfect soundness for a non-interactive proof system. Their definition implies that for all λ—

$$\Pr\left[\begin{array}{l}(\mathrm{crs},\mathrm{td},\varepsilon,\varepsilon) \leftarrow_{\$} \mathsf{\Pi.C}(1^\lambda);\\ (x,\mathrm{pf}) \leftarrow_{\$} \mathrm{A}(\mathrm{crs})\end{array} : (\mathsf{\Pi.V}(1^\lambda,\mathrm{crs},\varepsilon,x,\mathrm{pf})=\mathsf{true}) \wedge (x \notin \mathsf{L_R}(\mathrm{crs}))\right] = 0 \ .$$

This corresponds to the SND-P notion of soundness, which, as we have seen, does not in general transfer. However, the definition (and its computational variant, which is not defined in the paper) is adaptive, which is application-enabling.

[HU19]. Their requirement of (statistical) soundness is that for all non-uniform adversaries A there exists a negligible function ν such that—

$$\Pr\left[\begin{array}{l}(\mathrm{crs},\varepsilon,\varepsilon,\varepsilon) \leftarrow_{\$} \mathsf{\Pi.C}(1^\lambda);\\ (x,\mathrm{pf}) \leftarrow_{\$} \mathrm{A}(\mathrm{crs})\end{array} : (\mathsf{\Pi.V}(1^\lambda,\mathrm{crs},\varepsilon,x,\mathrm{pf})=\mathsf{true}) \wedge (x \notin \mathsf{L_R}(\mathrm{crs}))\right] \leq \nu(\lambda) \ .$$

The computational version is presumably that for all PT non-uniform adversaries there exists a negligible function ν such that for all λ the same equation above holds. This definition is equivalent to the SND-P notion of soundness we define in Figure 3. As we have shown in Theorem 4.4, this notion of soundness does not in general transfer. However it is a strong, application-enabling definition.

[CCH+19, PS19] These works consider two definitions of soundness, in the statistical case and in the computational case. The statistical soundness definition is the stronger, adaptive version, and corresponds to the SND-P notion of soundness in Figure 3. However, computational soundness has a weaker, non-adaptive definition (corresponding to the SND-E notion). This corresponds implicitly to our result that the transference would only hold for the weaker SND-E notion but not for the SND-P notion.

[LPWW20]. Note that this work, unlike the previous works, is in the designated-verifier model. It too, defines two forms of soundness, an adaptive version, and a non-adaptive version and points out the difficulty in arguing what corresponds to the transference of the SND-P notion. It therefore uses the weaker non-adaptive version for computational soundness.