NIZKs with an Untrusted CRS: Security in the Face of Parameter Subversion

Mihir Bellare[1] Georg Fuchsbauer[2] Alessandra Scafuro[3]

February 2016

Abstract

Motivated by the subversion of "trusted" public parameters in mass-surveillance activities, this paper studies the security of NIZKs in the presence of a maliciously chosen common reference string. We provide definitions for subversion soundness, subversion witness indistinguishability and subversion zero knowledge. We then provide both negative and positive results, showing that certain combinations of goals are unachievable but giving protocols to achieve other combinations.

[1] 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-1228890, NSF grant CNS-1526801, ERC Project ERCC FP7/615074 and a gift from Microsoft corporation. This work was done in part while visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

[2] Inria, Ecole Normale Supérieure, CNRS and PSL Research University, Paris, France. Email: `georg.fuchsbauer@ens.fr`. URL: `http://www.di.ens.fr/~fuchsbau/`.

[3] Computer Science Departments, Boston University and Northeastern University. Email: `scafuro@bu.edu`. URL: `http://cs-people.bu.edu/scafuro/`. Supported in part by NSF grants CNS-1347350, CNS-1413964, CNS-1012798 and CNS-1414119. This work was done in part while visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

Contents

1 Introduction **3**

2 Notation **8**

3 Security of NIZKs under CRS subversion **8**
- 3.1 NP relations and NI systems 8
- 3.2 Notions for honest CRS: SND, WI and ZK 9
- 3.3 Notions for subverted CRS: S-SND, S-WI and S-ZK 9
- 3.4 2-move protocols 11

4 Negative result: ZK and S-SND are not compatible **11**

5 Positive results **14**
- 5.1 Soundness and subversion ZK 14
- 5.2 Subversion SND and subversion WI 23
- 5.3 Soundness, ZK and subversion WI 24

A Relation of NI systems to 2-move protocols **28**

B Additional details for the proof of Theorem 5.2 **29**

C Proof sketch for Theorem 5.1 **31**

D Complete relations **32**

1 Introduction

The summer of 2013 brought shocking news of mass surveillance being conducted by the NSA and its counter-parts in other countries. The documents revealed new ways in which the adversary compromises security, ways not covered by standard models and definitions in cryptography. This opens up a new research agenda, namely to formalize security goals that defend against these novel attacks, and study the achievability of these goals. This agenda is being pursued along several fronts. The front we pursue here is *parameter subversion*, namely the compromise of security by the malicious creation of supposedly trusted public parameters for cryptographic systems. The representative example is the Dual EC random number generator (RNG).

DUAL EC. Dual EC is an NSA-designed, elliptic-curve based RNG, standardized as NIST SP 800-90 and ANSI X9.82. BLN [12] say that its story is "one of the most interesting in modern cryptography." The RNG includes two points P, Q on an elliptic curve that function as public parameters for the algorithm. At the Crypto 2007 rump session, Shumow and Ferguson noted that anyone who knew the discrete logarithm of P to base Q, meaning a scalar s such that $P = sQ$, could predict generator outputs. In a Wired Magazine article the same year, Schneier warned against Dual EC because it "just might contain a backdoor for the NSA." The NSA's response was that they had "generated P, Q in a secure, classified way." But the Snowden revelations (documents from project Bullrun and SIGINT) show that Dual EC was part of a systematic NSA effort to subvert standards. And in 2014, CNEGLRBMSF [21] showed the practical effectiveness of the subversion by demonstrating how the backdoor could be exploited to break TLS.

Two things are remarkable. The first is that the "trusted" public parameters were in fact subverted. The second is the effort put into ensuring that the subverted parameters were standardized and used. NSA-based pressure and lobbying not only lead to Dual EC remaining a US standard but even to its being in an international standard, ISO 18031:2005. In 2013 Reuters reported that the NSA paid RSA corporation \$10 million to make Dual EC the default method for random number generation in their BSafe library.

CRYPTOGRAPHY RESISTANT TO PARAMETER SUBVERSION. The lesson to take away is that a cryptographic system that relies on public parameters assumed to have been honestly generated, say by some "trusted" party, is at great practical risk from the possibility that the parameters were in fact maliciously generated with intent to subvert security of their use. We suggest that in response we should develop cryptography that is resistant to parameter subversion. This means that it should provide its usual security with trusted parameters, but retain as much security as possible when the parameters are maliciously generated.

Parameters arise in many places in cryptography, but a prominent one that springs to mind are non-interactive zero-knowledge (NIZK) systems, where the common reference string (CRS) is assumed to be honestly generated. NIZKs are not only important in their own right but used in a wide variety of applications, so their security under parameter subversion has far-reaching effects. This paper provides a treatment of resistance to parameter subversion for NIZKs, with definitions, negative results and positive results.

NIZKS. Non-interactive zero-knowledge systems originate with BFM [15] and BDMP [14] and have since seen an explosion in constructions and applications. The Groth-Sahai framework for efficient NIZKs [40] is widely utilized and we are seeing not only efficient NIZKs but also their implementation in systems [40, 35, 10, 28, 11]. Structure-preserving cryptography [1, 2, 36] was developed to allow these NIZKs to be used for efficient applications.

The NIZK model postulates a common reference string (CRS) that has been honestly generated according to some distribution. The pragmatics of how this is done receives little explicit attention.

Some early works talk of using digits of π and others speak whimsically of "a random string in the sky," but for the most part the understanding is that a trusted party will generate, and make public, the CRS. In light of the above, however, we must be concerned that the CRS is in fact maliciously generated. This is the issue addressed by our work.

An immediate avenue of attack that may come to mind is the following. NIZKs require for security that there is a simulator that generates a simulated CRS (indistinguishable from the honest one) together with a trapdoor allowing the simulator to generate proofs without knowing the witness. What if the subvertor generates the CRS via the simulator, so that it knows the trapdoor? Since this CRS is indistinguishable from an honestly generated one, the subversion will not be detected. Now, what does the subvertor gain? This seems to depend on the particular system and its properties. For example, the subvertor may be able to generate proofs of *false* statements and violate soundness. In some cases the trapdoor permits extraction of witnesses from honest proofs, in which case the subvertor would be able to violate zero knowledge. What we see here is that features built into the standard notions and constructions of NIZKs turn out to be potential liabilities in the face of subversion. Put another way, current NIZKs have the possibility of subversion effectively built into the security requirement because the simulator works by "subverting" the CRS.

Two remarks with regard to the above. (1) First, if it is unclear what is going on, or what conclusion to draw, there is a good reason, namely that we are trying to think or talk about what subversion does in the absence of a clear understanding of the subversion-resistance goal, effectively jumping the gun. To be able to effectively assess security we first need precise definitions of the new goal(s) underlying resistance to CRS subversion. Providing such definitions is the first contribution of this paper. (2) Second, while the above discussion may lead one to be pessimistic, we will see that in fact a surprising amount of security can be retained even under a maliciously generated CRS.

NIZK SECURITY, NOW. To discuss the new goals in subversion-resistant NIZKs we first back up to recall the standard goals in the current model where the CRS is trusted and assumed honestly generated. We distinguish three standard goals for a non-interactive (NI) system $\mathsf{\Pi}$ relative to an $\mathbf{NP}$ relation R defining the language $L(\mathsf{R}) \in \mathbf{NP}$. The formalizations are recalled in Section 3.

SND: (Soundness) It is hard for an adversary, given an honestly generated *crs*, to find an $x \notin L(\mathsf{R})$ together with a valid proof π (meaning one that the verification algorithm $\mathsf{\Pi.V}$ accepts) for x relative to *crs*.

WI: (Witness indistinguishability) Assuming *crs* is honestly generated, an adversary can't tell under which of two valid witnesses an honest proof (meaning one generated by the prover algorithm $\mathsf{\Pi.P}$ under *crs*) for an instance x was created, and moreover this holds even for multiple, adaptively chosen instances depending on *crs*.

ZK: (Zero-knowledge) There is a simulator $\mathsf{\Pi.Sim.crs}$ returning a simulated CRS crs_0 and associated trapdoor *std*, and an accomplice simulator $\mathsf{\Pi.Sim.pf}$ taking an instance $x \in L(\mathsf{R})$ and *std* and returning a proof, such that an adversary given crs_b cannot tell whether a proof it receives was created honestly (with the honest prover algorithm, an honest crs_1 and a witness; the $b = 1$ case) or via $\mathsf{\Pi.Sim.pf}$ (the $b = 0$ case). Moreover this holds even for multiple, adaptively chosen instances depending on crs_b.

NIZK SECURITY UNDER SUBVERSION. The key change in our model is that the adversary generates the CRS. It can retain, via its coins r, some kind of "backdoor" related to this CRS. In Section 3 we formalize the following goals:

	Standard			Subversion resistant			Achievable?
	SND	ZK	WI	S-SND	S-ZK	S-WI	
N		•		•			✗ Thm. 4.2
P1	•	•	•		•	•	✓ Thm. 5.2
P2	•		•	•		•	✓ Thm. 5.5
P3	•	•	•			•	✓ Thm. 5.6

$$\begin{tikzcd}
\text{S-SND} \arrow[d] & \text{S-ZK} \arrow[r] \arrow[d] & \text{S-WI} \arrow[d] \\
\text{SND} & \text{ZK} \arrow[r] & \text{WI}
\end{tikzcd}$$

Figure 1: **Left:** Achievability chart showing our negative result **N** and positive results **P1**, **P2**, **P3**. In a row we refer to simultaneously achieving all selected notions. **Right:** Relations.

S-SND: (Subversion soundness) It is hard for the adversary to generate a (malicious) CRS *crs* together with an instance $x \notin L(\mathsf{R})$ and a valid proof π for x relative to *crs*. (The goal of the subvertor here is to create a CRS that allows it to give proofs of false statements.)

S-WI: (Subversion witness indistinguishability) Even if the adversary creates *crs* maliciously and retains the corresponding coins r, it can't tell under which of two valid witnesses an honest proof (meaning one generated by the prover algorithm $\Pi.\mathsf{P}$ under the subverted *crs*) for an instance x was created, and moreover this holds even for multiple, adaptively chosen instances depending on *crs*.

S-ZK: (Subversion zero knowledge) For any adversary X creating a malicious CRS crs_1 using coins r_1, there is a simulator $\mathsf{S.crs}$ returning not only a simulated CRS crs_0 and associated trapdoor *std* *but also simulated coins* r_0, and an accomplice simulator $\mathsf{S.pf}$ taking an instance $x \in L(\mathsf{R})$ and *std* and returning a proof, such that an adversary A given crs_b, r_b cannot tell whether a proof it receives was created honestly (with the honest prover algorithm, crs_1 and a witness; the $b = 1$ case) or via $\mathsf{S.pf}$ (the $b = 0$ case). Moreover this holds even for multiple, adaptively chosen instances depending on crs_b, r_b.

The right side of Figure 1 may help situate the notions. It shows the obvious relations: S-X implies X, that ZK implies WI and that S-ZK implies S-WI.

<u>Achievability.</u> Is subversion resistance achievable? This question first needs to be meaningfully posed. The subversion-resistance goals are easy to achieve *in isolation*. For example, S-SND is achieved for any **NP** relation by having the prover send the witness, but this is not ZK. S-ZK is achieved by having the prover send the empty string as the proof and having the verifier always accept, but this is not SND. Such trivial constructions are un-interesting. The interesting question is whether meaningful combinations of the goals are simultaneously achievable. A pragmatic viewpoint is that we already have systems achieving SND+WI+ZK. We want to "upgrade" these to get some resistance to subversion. While retaining SND, WI and ZK, what can be added from the list S-SND, S-WI, S-ZK? Can we have them all? Are things so bad that we can have none? We will be able to completely categorize what is achievable and what is not and will see that the truth is somewhere between these extremes and on the whole the news is perhaps more positive than we might have expected. Our core results are summarized in the table on the left side of Figure 1. In any row, we are considering simultaneously achieving the notions indicated by the bullets. The last column indicates whether or not it is possible. We now discuss these results, beginning with the negative result of the first row.

<u>Negative result.</u> We first ask whether we can achieve S-SND (soundness for a malicious CRS) while retaining what we have now, namely SND, WI and ZK. Result **N** (the first row of Figure 1)

indicates that we cannot. It says that there is no NI system that achieves both ZK and S-SND. (More precisely, it is only possible for trivial **NP**-relations, i.e., where verifiers can check if $x \in L(\mathsf{R})$ themselves.) We stress that ZK here is the standard notion where the CRS is honest. We are not asking for S-ZK but only to retain ZK. The proof of Theorem 4.2 establishing this uses the paradigm of GO [32] of using the simulator to break soundness.

Positive results. Figure 1 lists three positive results that we discuss in turn:

P1: The most desirable target is S-ZK. By result **N** we cannot get it in combination with S-SND. The next best thing would be to get it in combination with SND. We show in Theorem 5.2 that this is possible. Since S-ZK implies all of ZK, S-WI and WI, this yields result **P1** of the table of Figure 1, showing we can simultaneously achieve all notions but S-SND. Theorem 5.2 is based on a knowledge-of-exponent assumption (KEA) in a group equipped with a bilinear map. The assumption is certainly strong, but (1) this is to be expected since our goal implies certain forms of 2-move interactive ZK that have themselves only been achieved under extractability assumptions [13], (2) similar assumptions have been made before [35], and (3) unlike other knowledge assumptions [13], our assumption is not ruled out assuming indistinguishability obfuscation. See the overview at the start of Section 5.1 for a high-level description of the ideas of our construction.

P2: The question left open by **P1** is whether there is some meaningful way to achieve S-SND. (It is the one item missing in row **P1**.) We know from result **N** that we cannot do this in combination with ZK. Result **P2** of the table of Figure 1 says that we can do the best possible given this limitation. Namely we can simultaneously achieve both S-SND and S-WI (and thus SND and WI). Theorem 5.5 establishing this is under a standard assumption, namely the decision-linear assumption (DLin). It follows easily from the existence of a SND and WI NI system with trivial CRS under DLin [38] and the observation (Lemma 5.4) that any such system is obviously also S-SND and S-WI.

P3: Result **P3** of the Figure 1 represents "hedging." The system has the desired properties (SND, WI, ZK) under an honest CRS. When the CRS is maliciously chosen, it does not break completely; it retains witness indistinguishability in the form of S-WI. In practice this offers quite a bit of protection. Our hedging construction combines a PRG with a zap. (Recall that the latter is a 2-move witness-indistinguishable interactive protocol [27].)

Result **P3** may seem redundant; isn't it implied by **P1**? (Indeed it selects a strict subset of the notions selected by **P1**.) The difference is that while **P1** uses strong (extractability) assumptions, **P3** is established in Theorem 5.6 under the minimal assumption that some SND+WI+ZK NI system exists. That is, our hedging adds no extra assumptions. This is because a zap can be built from any SND+ZK NI system [27].

Discussion and related work. There is a natural connection between NI systems and 2-move interactive protocols in which NI system $\mathsf{\Pi}$ corresponds to the protocol $\mathsf{2MV}$ in which the verifier first sends the CRS and the prover sends the proof in the second move (cf. Figure 14). We can then think of the following correspondence of notions for $\mathsf{\Pi}$ and $\mathsf{2MV}$: S-WI $\leftrightarrow$ ZAP; ZK $\leftrightarrow$ honest-verifier ZK; S-ZK $\leftrightarrow$ Full (cheating verifier) ZK. This analogy provides intuition and insight and opens up connections we exploit for both positive and negative results, but one must be wary that the analogy is not fully accurate. ZK for $\mathsf{\Pi}$ is not identical to honest-verifier ZK for $\mathsf{2MV}$. In particular the simulation requirement for ZK for $\mathsf{\Pi}$ is stronger because the simulated CRS (first move in $\mathsf{2MV}$) must be produced upfront without knowing the instance, and then the simulator must be able to adaptively produce simulated proofs for multiple instances.

S-ZK for $\mathsf{\Pi}$ is similarly not the same as (full) ZK for $\mathsf{2MV}$. The connection is further complicated by there being many variant notions of ZK for 2-move protocols. The Π notions imply some of these, but do not imply others. One thing on which the connection gives perspective is the difficulty of achieving S-ZK. Many forms of 2-move ZK are impossible [32, 4]. This does not make S-ZK impossible because S-ZK does not imply these particular forms of interactive ZK. (Indeed the definition of S-ZK was chosen to make this true.) A form of 2-move ZK has been achieved by BCPR [13] using extractability assumptions. This form of 2-move ZK is implied by S-ZK (but not vice-versa), indicating that it would be hard to achieve S-ZK without extractability assumptions. The strength of the assumptions we need for result **P1** is thus expected. For more details on the relation between NI systems and 2-move protocols see Appendix A.

The broad question we have asked is, which combinations of the six notions SND, WI, ZK, S-SND, S-WI, S-ZK are simultaneously achievable? Figure 1 looks at four combinations. But there are in principle 2^6 combinations about which one could ask. In Table 1 in Appendix D we go systematically over *all* combinations and evaluate achievability. We are able to give the answer in all cases. Briefly, Figure 1 covers the interesting cases, which is why we have focused on those for the body of the paper, and other cases are dealt with relatively easily in Appendix D.

Resistance of NIZKs to parameter subversion may not be of *immediate* practical relevance but we believe it is an important long-term consideration for this technology. The foundational tradition has always had as its stated goal to model and capture realistic, practical attacks and then investigate theoretically whether or not security can be achieved. Parameter subversion is such a realistic attack not previously considered, and it leads us to revisit the foundations of NIZKs to bring it into the picture. We are seeing large efforts in the creation of efficient NIZKs and their implementation in systems towards eventual applications [40, 35, 10, 28, 11, 9]. For security, parameter subversion must be kept in mind from the start.

We have been selective rather than exhaustive with regard to which notions to consider in this setting, focusing on the basic soundness, witness indistinguishability and zero knowledge. There are many other notions in this area that could be considered including robustness, simulation soundness and extractability [23, 34, 37, 25] but it seems fairly apparent that these stronger notions will be subject to commensurately strong negative results with regard to security under CRS subversion. For example, extractability asks that the simulator can create a CRS such that, with a trapdoor it withholds, it can extract the witness from a valid proof. But if so, a subvertor can create the CRS like the simulator so that it has the trapdoor and can also extract the witness.

A standard suggestion to protect against CRS subversion is to generate the CRS via a multi-party computation protocol so that no particular party controls the outcome. This is pursued in [9]. The effectiveness and practicality of this solution are not very clear. What parties would perform this task, and why can we trust *any* of them? The Snowden revelations indicate for example that corporations cooperate with the NSA toward subversion, either willingly or due to court orders. NIZKs with built-in resistance to subversion, as we define and achieve, provide greater protection.

One might note that in some applications, such as the use of NIZKs for signatures [5, 20, 25] and IND-CCA encryption [42, 26], the user can pick their own CRS and be confident of its quality. However this blows up key sizes and increases system complexity. It would be more convenient if there were a single, global CRS, in which case resistance to subversion matters.

CPs [19] study UC-secure computation in a model where the CRS is drawn from a distribution that is adversarially chosen subject to several restrictions, including that it has high min-entropy and is efficiently sampleable via an algorithm known to the simulator. They do not consider NIZKs, and in their model the CRS is not chosen fully maliciously, with no restrictions, as in our model.

Algorithm-substitution attacks, studied in [7, 3], are another form of subversion. They go back to the broader framework of kleptography [47, 48]. Back-doored blockciphers were studied in [45, 43, 44]. DGGJR [24] provide a formal treatment of back-dooring of PRGs in response to the Dual EC debacle.

2 Notation

The empty string is denoted by ε. If x is a (binary) string then $|x|$ is its length. If S is a finite set then $|S|$ denotes its size and $s \leftarrow\!\!\$\; S$ denotes picking an element uniformly from S and assigning it to s. We denote by $\lambda \in \mathbb{N}$ the security parameter and by 1^λ its unary representation.

Algorithms are randomized unless otherwise indicated. "PT" stands for "polynomial time", whether for randomized or deterministic algorithms. By $y \leftarrow A(x_1, \ldots; r)$ we denote the operation of running algorithm A on inputs $x_1, \ldots$ and coins r and letting y denote the output. By $y \leftarrow\!\!\$\; A(x_1, \ldots)$, we denote the operation of letting $y \leftarrow A(x_1, \ldots; r)$ for random r. We denote by $[A(x_1, \ldots)]$ the set of points that have positive probability of being output by A on inputs $x_1, \ldots$. Adversaries are algorithms. Complexity is uniform throughout: scheme algorithms and adversaries are Turing Machines, not circuit families.

For our security definitions and some proofs we use the code-based game playing framework of [8]. A game G (e.g. Figure 2) usually depends on some scheme and executes one or more adversaries. It defines oracles for the adversaries as procedures. The game eventually returns a boolean. We let Pr[G] denote the probability that G returns true.

3 Security of NIZKs under CRS subversion

We start by recalling and discussing standard notions of NIZK security in the setting used until now where the CRS is trusted. Then we turn to formulating new notions of NIZK security in the setting where the CRS is subverted. We start with the syntax.

3.1 NP relations and NI systems

NP relations. Proofs pertain to membership in an **NP** language defined by an **NP** relation, and we begin with the latter. Suppose $\mathsf{R}\colon \{0,1\}^* \times \{0,1\}^* \to \{\mathsf{true}, \mathsf{false}\}$. For $x \in \{0,1\}^*$ we let $\mathsf{R}(x) = \{\, w : \mathsf{R}(x, w) = \mathsf{true} \,\}$ be the *witness set* of x. We say that R is 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 $\mathsf{R}(x)$ has length at most $\mathsf{R.wl}(|x|)$ for all $x \in \{0,1\}^*$. We let $L(\mathsf{R}) = \{\, x : \mathsf{R}(x) \neq \emptyset \,\}$ be the *language* associated to R. The fact that R is an **NP** relation means that $L(\mathsf{R}) \in \mathbf{NP}$. We now go on to security properties, first giving formal definitions and then discussions.

NI systems. A non-interactive (NI) system specifies the syntax of the proof system. We can then consider various security attributes, including soundness, zero knowledge and witness indistinguishability. Formally, a NI system Π for R specifies the following PT algorithms. Via $crs \leftarrow\!\!\$\; \Pi.\mathsf{Pg}(1^\lambda)$ one generates a common reference string crs. Via $\pi \leftarrow\!\!\$\; \Pi.\mathsf{P}(1^\lambda, crs, x, w)$ the honest prover, given x and $w \in \mathsf{R}(x)$, generates a proof π that $x \in L(\mathsf{R})$. Via $d \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$ a verifier can produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ indicating whether π is a valid proof that $x \in L(\mathsf{R})$. We require (perfect) completeness, namely $\Pi.\mathsf{V}(1^\lambda, crs, x, \Pi.\mathsf{P}(1^\lambda, crs, x, w)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $crs \in [\Pi.\mathsf{Pg}(\lambda)]$, all $x \in L(\mathsf{R})$ and all $w \in \mathsf{R}(x)$. We also require that $\Pi.\mathsf{V}$ returns false if any of its arguments is $\perp$.

3.2 Notions for honest CRS: SND, WI and ZK

Soundness. Soundness asks that it be hard to create a valid proof for $x \notin L(\mathsf{R})$. Formally, we say that Π is sound for R, abbreviated SND, if $\mathbf{Adv}^{\mathrm{snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) = \Pr[\mathrm{SND}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)]$ and game SND is specified in Figure 2. This is a computational soundness requirement as opposed to a statistical one, as is sufficient for applications.

WI. This notion [29] requires that a PT adversary, which chooses two witnesses, cannot tell which one was used to create a proof. Formally, we say that Π is witness-indistinguishable (WI) for R, if $\mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{wi}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) = 2\Pr[\mathrm{WI}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)] - 1$ and game WI is specified in Figure 2. In this game, an adversary A can request a proof for x under one of two witnesses w_0, w_1. It is returned an honestly generated proof under w_b where b is the challenge bit. It can adaptively request and obtain many such proofs before outputting a guess b' for b. The game returns true if this guess is correct.

ZK. We say that Π is zero-knowledge for R, abbreviated ZK, if Π specifies additional PT algorithms $\Pi.\mathsf{Sim.crs}$ and $\Pi.\mathsf{Sim.pf}$ such that $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) = 2\Pr[\mathrm{ZK}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)] - 1$ and game ZK is specified in Figure 2. Adversary A can adaptively request proofs by suppling 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 $\Pi.\mathsf{Sim.pf}$ using a trapdoor std. The adversary outputs a guess b' as to whether the proofs were real or simulated.

Discussion. The classical definitions of soundness and zero knowledge for proof systems [33] were in what we will call the complexity-theoretic style. The soundness condition said that for all $x \notin L(\mathsf{R})$, the probability that a dishonest prover could convince the honest verifier to accept was low. Zero knowledge, similarly, looked at distributions associated to a fixed $x \in L(\mathsf{R})$ and then at ensembles over x. The first definition for NIZK was similar [14]. But over time, NIZK definitions have adapted to what we call a cryptographic style [23, 39]. This is the style we use because it seems more prevalent now and it works better for applications. Here x is not quantified but chosen by an adversary. The definitions directly capture proofs for multiple, related statements. All adversaries are PT, meaning all metrics are computational.

One consequence of the complexity-theoretic style was a need for non-uniform complexity for adversaries and assumptions [33, 31]. Goldreich [30] made a case for uniform complexity. The cryptographic style we adopt is in this vein, and in our setting all complexity (adversaries, algorithms, assumptions) is uniform.

3.3 Notions for subverted CRS: S-SND, S-WI and S-ZK

A core assumption in NIZKs is that the CRS is honestly generated. In light of subversion of parameters in other contexts as part of the mass-surveillance revelations, we ask what would happen if the CRS were maliciously generated. We will define subversion-resistance analogues S-SND, S-WI and S-ZK of the SND, WI, ZK goals above. The key difference is that the CRS is selected by an adversary rather than via the CRS-generation algorithm $\Pi.\mathsf{Pg}$ prescribed by Π.

Subversion soundness. Subversion soundness asks that if a subvertor creates a CRS in any way it likes, it will still be unable to prove false statements under that CRS. Formally, we say that Π is subversion-sound for R, abbreviated S-SND, if $\mathbf{Adv}^{\mathrm{s\text{-}snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{s\text{-}snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) = \Pr[\text{S-SND}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)]$ and game S-SND is specified in Figure 2. Compared to the honest-CRS game SND to the left of it, the adversary now not only generates x and π, but itself supplies crs, modeling a malicious choice of the latter.

```
Game SND_{Π,R,A}(λ)
    $crs \leftarrow\$ \Pi.\mathsf{Pg}(1^\lambda)$
    $(x,\pi) \leftarrow\$ \mathsf{A}(1^\lambda, crs)$
    Return $(x \notin L(\mathsf{R}) \ \wedge\ \Pi.\mathsf{V}(1^\lambda, crs, x, \pi))$
```

```
Game S-SND_{Π,R,A}(λ)
    $(crs, x, \pi) \leftarrow\$ \mathsf{A}(1^\lambda)$
    Return $(x \notin L(\mathsf{R})$ and $\Pi.\mathsf{V}(1^\lambda, crs, x, \pi))$
```

```
Game WI_{Π,R,A}(λ)
    $b \leftarrow\$ \{0,1\}$
    $crs \leftarrow\$ \Pi.\mathsf{Pg}(1^\lambda)$
    $b' \leftarrow\$ \mathsf{A}^{\textsc{Prove}}(1^\lambda, crs)$
    Return $(b = b')$

Prove(x, w_0, w_1)
    If $\mathsf{R}(x, w_0) = \mathsf{false}$ or $\mathsf{R}(x, w_1) = \mathsf{false}$
        then Return $\perp$
    $\pi \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, crs, x, w_b)$
    Return $\pi$
```

```
Game S-WI_{Π,R,A}(λ)
    $b \leftarrow\$ \{0,1\}$
    $(crs, st) \leftarrow\$ \mathsf{A}(1^\lambda)$
    $b' \leftarrow\$ \mathsf{A}^{\textsc{Prove}}(1^\lambda, crs, st)$
    Return $(b = b')$

Prove(x, w_0, w_1)
    If $\mathsf{R}(x, w_0) = \mathsf{false}$ or $\mathsf{R}(x, w_1) = \mathsf{false}$
        then Return $\perp$
    $\pi \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, crs, x, w_b)$
    Return $\pi$
```

```
Game ZK_{Π,R,A}(λ)
    $b \leftarrow\$ \{0,1\}$
    $crs_1 \leftarrow\$ \Pi.\mathsf{Pg}(1^\lambda)$
    $(crs_0, std) \leftarrow\$ \Pi.\mathsf{Sim.crs}(1^\lambda)$
    $b' \leftarrow\$ \mathsf{A}^{\textsc{Prove}}(1^\lambda, crs_b)$
    Return $(b = b')$

Prove(x, w)
    If $\mathsf{R}(x, w) = \mathsf{false}$ then Return $\perp$
    If $b = 1$ then $\pi \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, crs_1, x, w)$
    Else $\pi \leftarrow\$ \Pi.\mathsf{Sim.pf}(1^\lambda, crs_0, std, x)$
    Return $\pi$
```

```
Game S-ZK_{Π,R,X,S,A}(λ)
    $b \leftarrow\$ \{0,1\}$
    $r_1 \leftarrow\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$ ; $crs_1 \leftarrow \mathsf{X}(1^\lambda; r_1)$
    $(crs_0, r_0, std) \leftarrow\$ \mathsf{S.crs}(1^\lambda)$
    $b' \leftarrow\$ \mathsf{A}^{\textsc{Prove}}(1^\lambda, crs_b, r_b)$
    Return $(b = b')$

Prove(x, w)
    If $\mathsf{R}(x, w) = \mathsf{false}$ then Return $\perp$
    If $b = 1$ then $\pi \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, crs_1, x, w)$
    Else $\pi \leftarrow\$ \mathsf{S.pf}(1^\lambda, crs_0, std, x)$
    Return $\pi$
```

Figure 2: Games defining standard (left) and subversion (right) security of NI system Π. Top to bottom: Soundness, witness indistinguishability, zero knowledge.

Subversion WI. Subversion WI asks that if a subvertor creates a CRS in any way it likes then it will still be unable to tell which of two witnesses was used to create a proof, even given both witnesses. Formally, we say that Π is subversion witness-indistinguishable (abbreviated S-WI) for R if $\mathbf{Adv}^{\text{s-wi}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{s-wi}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) = 2\Pr[\text{S-WI}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)] - 1$ and game S-WI is specified in Figure 2. Compared to the honest-CRS game WI, the CRS *crs* is now generated by the adversary in a first stage, along with state information *st* passed to the adversary in its second stage. In the latter, via its Prove oracle, it adaptively obtains proofs for instances of its choice under a challenge witness, and outputs a guess b' for the challenge b. The state can contain the coins of A or any trapdoor associated to *crs* that A chooses to put there, and it can use this in its second-stage distinguishing task.

Subversion ZK. Subversion ZK asks that for any CRS subvertor X creating a CRS in any way it likes there is a simulator able to produce the full view of the CRS subvertor, including its coins and proofs corresponding to adaptively chosen instances, without knowing the witnesses. Formally, a simulator S for X specifies PT algorithms $\mathsf{S.crs}$ and $\mathsf{S.pf}$. Now consider game S-ZK of Figure 2 associated to $\Pi, \mathsf{R}, \mathsf{X}, \mathsf{S}$ and an adversary A. We let $\mathbf{Adv}^{\text{s-zk}}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) = 2\Pr[\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda)] - 1$.

We say that Π is subversion zero-knowledge (abbreviated S-ZK) for R if for all PT CRS subvertors X there is a PT simulator S such that for all PT A the function $\mathbf{Adv}^{\text{s-zk}}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\cdot)$ is negligible.

In this game, if the challenge bit b is 1 then the CRS crs_1 is generated via X with the coins r_1 made explicit. Otherwise, if $b = 0$, the first stage $\mathsf{S.crs}$ of the simulator is run to produce simulated versions crs_0, r_0 not only of the CRS but *also of the coins of* X. Alongside, $\mathsf{S.crs}$ produces a simulation trapdoor std as in ZK to allow its second stage to simulate proofs. Now, A gets to request its Prove oracle for proofs of instances of its choice. If $b = 1$, these are produced by the honest prover with the given witness; but if $b = 0$, they are produced via the second stage $\mathsf{S.pf}$ of the simulator using the simulation trapdoor std and no witness. Adversary A produces its guess b' and wins of $b' = b$.

The definition reflects that X here is like a cheating verifier in classical ZK [33]. The simulator thus needs to produce its coins as well as the transcript of its interaction with its oracle. But also, to reflect the ZK requirement of *non-interactive* systems above, more is required, namely that the simulator must first produce the simulated CRS and coins, and then, in its second stage, be able to produce simulated proofs. The definition is thus quite demanding. Note that the simulator can depend (in a non-blackbox way) on X, but not on A. The latter is important to ensure that S-ZK implies ZK.

3.4 2-move protocols

We will have many occasions to refer to and use 2-move interactive protocols, so we fix a syntax for them. A 2-move protocol $\mathsf{2MV}$ for **NP** relation R specifies PT algorithms $\mathsf{2MV.V}$, $\mathsf{2MV.P}$, $\mathsf{2MV.D}$. Via $(m_1, st) \leftarrow\!\!\$\ \mathsf{2MV.V}(1^\lambda, x)$ the honest verifier generates the first move message m_1 on input x, retaining associated state information st. Via $m_2 \leftarrow\!\!\$\ \mathsf{2MV.P}(1^\lambda, x, w, m_1)$ the honest prover generates a reply computed from x, a witness $w \in \mathsf{R}(x)$ and the first move message m_1. Deterministic decision algorithm $\mathsf{2MV.D}$ takes x, m_1, m_2, st and returns a boolean decision. Security notions will be discussed as needed. The relation of NI systems to 2-move protocols is discussed in Appendix A.

4 Negative result: ZK and S-SND are not compatible

All the different forms of subversion security (S-SND, S-WI, S-ZK) are easy to achieve in isolation, meaning if nothing else is required. For example sending the witness as the proof achieves S-SND (but this is not ZK). Having the verification algorithm always accept and sending the empty string as the proof achieves S-ZK (but this is not SND). These kinds of results are not interesting. We want to study the simultaneous achievability of meaningful combinations of the notions, meaning some kind of soundness together with some kind of zero knowledge or witness indistinguishability.

We already have NI systems that are SND+ZK and we do not want to degrade this. If now the CRS is subverted, what more can we have without losing the initial properties? The first question we ask is, can we up the ante for soundness, meaning add S-SND? That is, we want subversion soundness while retaining ZK. We will show that this is not possible.

An impossibility result in this domain means no NI system satisfying the conditions exists unless the relation R is trivial. Roughly, trivial means that the verification algorithm can decide membership in $L(\mathsf{R})$ on its own. Impossibility results of this type begin with Goldreich and Oren (GO) [32]. Their definition of R being trivial was simple, namely that it is in **BPP**. This will not suffice here, so we begin with a more precise definition of relation triviality and an explanation of why it is needed.

```
$\underline{\text{GAME } \mathrm{DEC}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\lambda)}$
$(x, w) \leftarrow\!\!\$\ \mathsf{IG}(1^\lambda)$; $d_1 \leftarrow \mathsf{R}(x, w)$
If $(x \in L(\mathsf{R})$ and $d_1 = \mathsf{false})$ then return $\mathsf{false}$
$d_0 \leftarrow\!\!\$\ \mathsf{M}(1^\lambda, x)$ ; return $(d_0 \neq d_1)$
```

Figure 3: Game defining language triviality

<u>RELATION TRIVIALITY.</u> The definition of a relation R being trivial if $L(\mathsf{R}) \in \mathbf{BPP}$ works when the formulations of ZK and soundness being used are in the complexity-theoretic style, meaning the conditions refer to universally quantified inputs. As discussed in Section 3.2 however, our formulations, following modern treatments of NI systems in the literature, are in the cryptographic style, which is better suited for applications. Here the only instances that come into play are those that can be generated by PT algorithms, and the only positive instances that come into play are those generated with witnesses. In this setting, **BPP** will not work as a definition of triviality because membership in standard complexity classes like **BPP** refers to arbitrary inputs, not merely ones that one can generate in PT. For our purposes we thus give a definition of a language (actually an **NP** relation) being trivial, which can be seen as defining a cryptographic version of **BPP**.

Let R be an **NP** relation. An *instance generator* is a PT algorithm that on input 1^λ returns a pair (x, w). Here x is a challenge instance that may or may not be in $L(\mathsf{R})$, and w should be in $\mathsf{R}(x)$ if $x \in L(\mathsf{R})$. Let M be an algorithm (decision procedure) taking $1^\lambda, x$ and returning a boolean representing whether or not it thinks x is in $L(\mathsf{R})$. Consider game DEC of Figure 3 associated to $\mathsf{IG}, \mathsf{R}, \mathsf{M}$ and let $\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\lambda) = \Pr[\mathrm{DEC}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\lambda)]$. We say that algorithm M decides R if for every PT IG the function $\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\cdot)$ is negligible. We say that R is trivial if there is a PT algorithm M that decides R. Intuitively, in game DEC, think of IG as an adversary trying to make M fail. The game returns true when IG succeeds, meaning M returns the wrong decision. A technical point is that if IG generates a positive instance x, the game forces it to lose if the witness w is not valid. Thus we are asking that M is able to decide membership in PT for instances that can be efficiently generated with valid witnesses if the instance is positive. But this does not mean it can decide membership on all instances. Thus if $L(\mathsf{R}) \in \mathbf{BPP}$ then R is certainly trivial, but the converse need not be true.

The following proposition formally states the result that motivates this definition, namely that for trivial relations, achieving all goals is itself trivial:

Proposition 4.1 *Let* R *be a trivial* **NP***-relation. Then there is a NI system* Π *for* R *that is S-SND, S-ZK and S-WI.*

Proof: Let M be a PT algorithm that decides R. Let $\Pi.\mathsf{Pg}$ return the empty string ε. Let $\Pi.\mathsf{P}$ return ε as the proof. Let $\Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$ return $\mathsf{M}(1^\lambda, x)$. Given A we can construct IG such that $\mathbf{Adv}^{\text{s-snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot) \leq \mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\cdot)$, which establishes S-SND. S-ZK is trivial since an honest proof is always the empty string, and S-WI is implied by S-ZK. We omit the details. ▌

Prop. 4.1 validates our triviality definition and shows why it is needed. Now we turn to the interesting question, namely what happens for non-trivial relations.

<u>RESULT.</u> We show that ZK and subversion soundness (S-SND) cannot co-exist, meaning only trivial relations will have NI systems with both attributes. We stress that we are not asking here for subversion ZK but just plain ZK.

```
GAMES $G_0$, $\boxed{G_1}$
    $(x, w) \leftarrow\$ \mathsf{IG}(1^\lambda)$ ; $d_1 \leftarrow \mathsf{R}(x, w)$
    $(crs, std) \leftarrow\$ \Pi.\mathsf{Sim.crs}(1^\lambda)$
    $\pi \leftarrow\$ \Pi.\mathsf{Sim.pf}(1^\lambda, crs, std, x)$
    $d_0 \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$
    $b \leftarrow ((x \notin L(\mathsf{R})) \wedge (d_0 = \mathsf{true}))$
    $\boxed{b \leftarrow ((d_1 = \mathsf{true}) \wedge (d_0 = \mathsf{false}))}$
    Return $b$

GAME $G_2$
    $(x, w) \leftarrow\$ \mathsf{IG}(1^\lambda)$ ; $d_1 \leftarrow \mathsf{R}(x, w)$
    $crs \leftarrow\$ \Pi.\mathsf{Pg}(1^\lambda)$
    $\pi \leftarrow\$ \Pi.\mathsf{P}(1^\lambda, crs, x, w)$
    $d_0 \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$
    $b \leftarrow ((d_1 = \mathsf{true}) \wedge (d_0 = \mathsf{false}))$
    Return $b$
```

Figure 4: Games for proof of Theorem 4.2

Theorem 4.2 *Let Π be a NI system satisfying zero knowledge (ZK) and subversion soundness (S-SND) for an* **NP** *relation* R. *Then* R *is trivial.*

The proof follows the basic paradigm of GO [32]. We use the simulator to build a cheating prover that violates soundness. In our case this works if soundness holds relative to a simulated CRS, but S-SND guarantees this.

Proof of Theorem 4.2: Define the following decision procedure M:

```
Algorithm $\mathsf{M}(1^\lambda, x)$
    $(crs_0, std_0) \leftarrow\$ \Pi.\mathsf{Sim.crs}(1^\lambda)$; $\pi \leftarrow\$ \Pi.\mathsf{Sim.pf}(1^\lambda, crs_0, std_0, x)$
    Return $\Pi.\mathsf{V}(1^\lambda, crs_0, x, \pi)$
```

Thus, to decide if $x \in L(\mathsf{R})$, algorithm M runs the simulator to get a simulated CRS and simulation trapdoor, uses the latter to generate a simulated proof, and decides that $x \in L(\mathsf{R})$ if this proof is valid. Let IG be any PT instance generator. We will show below that $\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\cdot)$ is negligible. This shows that R is trivial.

To show $\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\cdot)$ is negligible, below we will define PT adversaries A, B such that

$$\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\lambda) \leq \mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) + \mathbf{Adv}^{\mathrm{s\text{-}snd}}_{\Pi,\mathsf{R},\mathsf{B}}(\lambda) \tag{1}$$

for all $\lambda \in \mathbb{N}$. By assumption, Π satisfies ZK and S-SND for R, so the functions $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ and $\mathbf{Adv}^{\mathrm{s\text{-}snd}}_{\Pi,\mathsf{R},\mathsf{B}}(\cdot)$ are both negligible. Thus Eq. (1) implies that $\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\cdot)$ is negligible, as desired.

Consider games G_0, G_1, G_2 of Figure 4. Game G_0 is defined ignoring the box, while game $\boxed{G_1}$ includes it. Games G_0 and G_1 split up the decision process depending on whether or not $x \in L(\mathsf{R})$. Game G_2 switches to a real CRS and proofs, which it can do since the instance generator provided a witness.

Game DEC returns true if $d_1 = \mathsf{true}$ and $d_0 = \mathsf{false}$; if $d_1 = \mathsf{false}$ and $d_0 = \mathsf{true}$ we must also have $x \notin L(\mathsf{R})$, which however implies $d_1 = \mathsf{false}$. We thus have

$$\mathbf{Adv}^{\mathrm{dec}}_{\mathsf{IG},\mathsf{R},\mathsf{M}}(\lambda) = \Pr[G_0] + \Pr[G_1] = \Pr[G_0] + \Pr[G_2] + (\Pr[G_1] - \Pr[G_2]) \ . \tag{2}$$

Notice that by completeness of Π we have

$$\Pr[G_2] = 0 \ . \tag{3}$$

Now we specify the adversaries A, B as follows:

```
Adversary $\mathsf{A}^{\textsc{Prove}}(1^\lambda, crs)$
$(x, w) \leftarrow\!\!\$\ \mathsf{IG}(1^\lambda)$ ; $d_1 \leftarrow \mathsf{R}(x, w)$
$\pi \leftarrow\!\!\$\ \textsc{Prove}(x, w)$ ; $d_0 \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$
If $((d_1 = \mathsf{true}) \wedge (d_0 = \mathsf{false}))$ then return $b' \leftarrow 0$
Else return $b' \leftarrow 1$
```

```
Adversary $\mathsf{B}(1^\lambda)$
$(x, w) \leftarrow\!\!\$\ \mathsf{IG}(1^\lambda)$
$(crs, std) \leftarrow\!\!\$\ \Pi.\mathsf{Sim.crs}(1^\lambda)$
$\pi \leftarrow\!\!\$\ \Pi.\mathsf{Sim.pf}(1^\lambda, crs, std, x)$
Return $(crs, x, \pi)$
```

Then we have

$$\Pr[\mathrm{G}_0] \leq \mathbf{Adv}^{\text{s-snd}}_{\Pi,\mathsf{R},\mathsf{B}}(\lambda) \tag{4}$$

$$\Pr[\mathrm{G}_1] - \Pr[\mathrm{G}_2] \leq \mathbf{Adv}^{\text{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) \ . \tag{5}$$

Putting together Eqs. (2), (3), (4) and (5) we get Eq. (1). ▮

5 Positive results

We already have NI systems that are SND+ZK, or SND+WI. We ask, if the CRS is subverted, what more can we have without losing the initial properties?

Can we add S-ZK? In Sect. 5.1 we answer positively to this question (result **P1**), showing a protocol that is SND+ZK+S-ZK under a knowledge-of-exponent assumption (KEA) in a group equipped with a bilinear map. In light of negative result **N**, this is the best we can achieve if we want to retain the ZK property in presence of CRS subversion.

Can we add S-SND? In light of **N**, we know that we cannot have S-SND and any form of ZK together. The best we can achieve while retaining S-SND is S-WI. In Sect. 5.2 we show that there exist NI systems that are S-SND+S-WI (result **P2**).

Result **P1** provides S-ZK but requires KEA. A natural question is, if we relax the requirement of S-ZK and aim to retain S-WI, can we achieve it from weaker assumptions? In Sect. 5.3 we show that there exists a NI system that is SND, ZK and S-WI under the weaker assumption that one-way functions and zaps exist.

5.1 Soundness and subversion ZK

<u>Overview.</u> To achieve S-ZK, a simulator must be able to simulate proofs under a CRS output by a subvertor. As opposed to ZK, the simulator thus cannot embed a trapdoor in the CRS, nor can it extract one from the subvertor by rewinding, as there is no interaction with it. We will instead rely on a knowledge assumption, stating that an algorithm can only produce a certain output if it knows underlying information. This is formalized by requiring that there exists an extractor that extracts the information from the algorithm. We will use this information as the simulation trapdoor, which we can extract from a subvertor computing a CRS. For soundness, a minimal requirement is that it is hard for the adversary to obtain the trapdoor from an honestly generated CRS.

The knowledge-of-exponent assumption (KEA) for a group $\mathbb{G}$, generated by g, states that from any algorithm which given a random element $h \leftarrow\!\!\$\ \mathbb{G}$ returns a pair of the form (g^s, h^s) one can efficiently extract s. A possible approach for a NI system is to define the CRS as a pair (g^s, h^s), for random s, and define a proof for $x \in L$ to prove that either $x \in L$ or one knows the value s in the CRS. By extracting s, the simulator in the S-ZK game can simulate proofs, while the adversary in the soundness game must supposedly use a witness for x, since it does not know s.

There are two problems with this approach: who chooses the group $\mathbb{G}$ and who chooses the element h used to prove knowledge of s? We address the first problem by letting the group $\mathbb{G}$ be part of the scheme specification. As for the choice of h, it cannot be chosen at CRS setup, since if the subvertor knows $\eta = \log_g h$, it can produce a CRS *without* knowing s by randomly picking $S \leftarrow_{\$} \mathbb{G}$ and setting $S' \leftarrow S^{\eta}$. Fixing h and letting it also be also part of the scheme description is problematic, since again, what guarantees that the subvertor does not know its logarithm and thereby break KEA? We overcome this problem by defining a new type of KEA, stating that in order to produce elements $(h = g^{\eta}, g^s, h^s)$, one has to *either* know s *or* η. As tuples of this form are Diffie-Hellman tuples, we call the assumption DH-KEA.

We define a CRS as a tuple $(g^{s_0}, g^{s_1}, g^{s_0 s_1})$, and let a proof for a statement x prove that either there is a witness for x or one knows s_0 or s_1. We prove knowledge by adding a ciphertext C, and use a perfectly sound witness-indistinguishable NI proof ζ with trivial CRS (a.k.a. a non-interactive zap) to prove that either $x \in L$ or C encrypts s_0 or s_1. (Using linear encryption for C and the NI system by GOS [38], both IND-CPA of C, as well as WI of ζ, follow from the decision-linear assumption (Dlin) [16].)

The sketched scheme is ZK since by encrypting the trapdoor s_0 (or s_1) proofs can be simulated, and by IND-CPA of C and WI of ζ they are indistinguishable from real ones. But we defined the CRS to allow even more: by DH-KEA, from a CRS subvertor we can *extract* either s_0 or s_1, which should yield S-ZK. Not quite, since the subvertor could simply output random group elements (S_0, S_1, S_2), from which we cannot extract. Since the GOS NI system requires a *bilinear* group, we can use its pairing to check CRS well-formedness. The prove (and verification) algorithm can then reject a malformed CRS, which together with simulatability under a well-formed CRS yields S-ZK.

Soundness intuitively holds because, by soundness of ζ, a proof for a wrong statement must contain an encryption of s_0 or s_1, which should be infeasible to obtain from an honestly generated CRS if computing discrete logarithms (DL) is hard. (Given a DL challenge S, one can randomly set S_0 or S_1 to S and with probability $\frac{1}{2}$, the proof contains an encryption of $\log S$.) To formally prove soundness, the reduction must recover s from C. We could include in the CRS a public key under which C is to be encrypted: the reduction sets up the CRS, knows the decryption key and can obtain s. Alas, this would break S-ZK: an adversary that created the CRS could also decrypt C and thereby distinguish real proofs from simulated ones.

We therefore include the linear-encryption key $pk = (g^u, g^v)$ in the proof rather than the CRS. But how would the soundness reduction then retrieve s? Could we use KEA again? Since we can only extract one of two possible logarithms, we do the following. The proof contains *two* public keys $pk_0 = (g^{u_0}, g^{v_0})$ and $pk_1 = (g^{u_1}, g^{v_1})$ and s is encrypted under both of them. Additionally, the proof contains elements $g^{u_0 u_1}, g^{u_0 v_1}, g^{v_0 u_1}, g^{v_0 v_1}$, whose consistency can be verified via the pairing. By DH-KEA, there exists an extractor which from $(g^{u_0}, g^{u_1}, g^{u_0 u_1})$ extracts either u_0 or u_1, another extractor that from $(g^{u_0}, g^{v_1}, g^{u_0 v_1})$ extracts u_0 or v_1, and so on. Together these four extractors either yield (u_0, v_0) or (u_1, v_1), thus one of the secret keys corresponding to pk_0 and pk_1. This way the soundness reduction can extract the value s encrypted in a proof for a false statement. At the same time we show that S-ZK still holds.

In our actual scheme we use the CDH assumption (defined below and implied by DLin) instead of DL. The reason is that CDH solutions are group elements, which can be efficiently encrypted using linear encryption. The trapdoor is then solution to a CDH instance in the CRS. Besides 14 group elements, the most costly component of our proofs is the GOS NI proof ζ. It uses a circuit representation of the NP relation R and shows that (a) either $\mathsf{R}(x, w)$ for some w, or (b) the simulation trapdoor was encrypted (see Eq. (6)). The GOS system [38] was further developed by Groth and Sahai [40] yielding very efficient proofs for algebraic statements, and we could replace GOS by GS. As the clause (b) that we added has precisely this algebraic form, the overhead for turning a proof that is merely WI into one that is S-ZK would be quite modest.

Discussion. Our scheme specification includes the bilinear group, so one might ask whether we

```
GAME $\mathrm{KE}_{\mathsf{dGG},\mathsf{M},\mathsf{E}}(\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$h_0, h_1 \leftarrow\!\!\$\ \mathbb{G};\ r \leftarrow\!\!\$\ \{0,1\}^{\mathsf{M.rl}(\lambda)}$
$(S_0, S_1, S_2) \leftarrow \mathsf{M}(1^\lambda, h_0, h_1; r)$
$s \leftarrow\!\!\$\ \mathsf{E}(1^\lambda, h_0, h_1, r)$
Return $\big(\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2) \wedge g^s \neq S_0 \wedge g^s \neq S_1\big)$
```

```
GAME $\mathrm{CDH}_{\mathsf{dGG},\mathsf{A}}(\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$s, t \leftarrow\!\!\$\ \mathbb{Z}_p;\ C \leftarrow\!\!\$\ \mathsf{A}(1^\lambda, g^s, g^t)$
Return $(C = g^{st})$
```

```
GAME $\mathrm{DLin}_{\mathsf{dGG},\mathsf{A}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\};\ (p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$u, v, s, t, \xi \leftarrow\!\!\$\ \mathbb{Z}_p;\ b' \leftarrow\!\!\$\ \mathsf{A}(1^\lambda, g^u, g^v, g^{us}, g^{vt}, g^{s+t+b\cdot\xi})$
Return $(b = b')$
```

Figure 5: Games defining the knowledge-of-exponent assumption (left), the CDH assumption (top right) and the DLin assumption (bottom right)

have not just shifted the subversion risk from the CRS to the choice of the group. Since the group generation algorithm is deterministic and public, anyone can run the algorithm to re-obtain the group; moreover, different entities can implement it independently if they think that some standardized implementation was subverted, as a check.

With the CRS, the situation is different. There is no easy way to check that it was properly generated, at least without compromising security. Perhaps a vocabulary that speaks to this is that the group is *reproducible*, whereas the CRS is not. Someone is trusted to produce it and one cannot easily check that they did it honestly.

Still, one must ask whether the algorithms used allow embedding of backdoors. Here we must look at the specific algorithms. Thus, while one could use a bilinear group in which the discrete-log problem is easy, leading to an insecure scheme, we know it is possible to publicly specify good algorithms. The specifications, given for example in research papers, may be used by anyone to re-produce the results of the algorithms with some faith that there are no backdoors, in the case (as here) that these algorithms are deterministic.

Speaking broadly, we cannot (and do not claim to) prevent all possible subversion. This is not possible. Our goal is to put in defenses that make the most obvious paths harder, one of which is subversion of the CRS.

BILINEAR GROUPS. Our construction is based on bilinear groups for which we introduce a new type of knowledge-of-exponent assumption. A bilinear-group generator GGen is a PT algorithm that takes input a security parameter 1^λ and outputs a description of a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, where p is a prime of length λ, $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order p, g generates $\mathbb{G}$ and $\mathbf{e}\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map that is non-degenerate (i.e. $\langle \mathbf{e}(g, g) \rangle = \mathbb{G}_T$).

While in the cryptographic literature bilinear groups are often assumed to be probabilistically generated, real-world pairing-based schemes are defined for groups that are fixed for every λ. We reflect this by defining the group generator as a deterministic PT algorithm dGG. An advantage of doing so is that every entity in the scheme can compute the group from the security parameter and no party must be trusted with generating the group.

KEA. The knowledge-of-exponent assumption (KEA) [22, 41, 6] in a group $\mathbb{G}$ states that an algorithm M that is given two random generators g, h of $\mathbb{G}$ and outputs (g^c, h^c) must know c. This is formalized by requiring that there exists an extractor for M which when given M's coins outputs c. Generalizations of KEA were used in the bilinear-group setting in [35]. We introduce a new type of KEA in bilinear groups, which we call DH-KEA, where we assume that if M outputs a Diffie-Hellman (DH) tuple g^s, g^t, g^{st} then it must either know s or t. This should also be the case when M is given two additional random generators h_0, h_1. We note that while an adversary may produce one group element without knowing its discrete logarithm by hashing into the elliptic curve [17, 46, 18], it seems hard to produce a DH tuple without knowing at least one of the logarithms.

Formally, let $\mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{M},\mathsf{E}}(\lambda) = \Pr[\mathrm{KE}_{\mathsf{dGG},\mathsf{M},\mathsf{E}}(\lambda)]$, where game KE is defined in Figure 5. The DH-KEA assumption holds for dGG if for every PT M there exists a PT E s.t. $\mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{M},\mathsf{E}}(\cdot)$ is negligible.

We note that due to deterministic group generation the assumption does not hold for non-uniform machines M, as their advice for inputs 1^λ could simply be a DH tuple (S_0, S_1, S_2) w.r.t. the group output by $\mathsf{dGG}(1^\lambda)$. However, we follow Goldreich [30] and only consider uniform machines. As a sanity check, we show that DH-KEA holds in the generic-group model. To reflect hashing into elliptic curves, we provide the adversary with an additional generic operation: it can create new group elements without knowing their discrete log. In Appendix C we show the following.

Theorem 5.1 *DH-KEA, as defined above, holds in the generic-group model with hashing into the group.*

CDH. The computational Diffie-Hellman assumption in a group $\mathbb{G}$ states that given g^s and g^t for a random s, t, it should be hard to compute g^{st}. Formally, the CDH assumption holds for dGG if $\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{dGG},\mathsf{A}}(\cdot)$ is negligible for all PT adversary A, where $\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{dGG},\mathsf{A}}(\lambda) = \Pr[\mathrm{CDH}_{\mathsf{dGG},\mathsf{A}}(\lambda)]$ and game CDH is specified in Figure 5.

DLin. The decision linear assumption [16] in a group $\mathbb{G}$ states that given $(g^u, g^v, g^{us}, g^{vt})$ for random u, v, s, t, the element g^{s+t} is indistinguishable from a random group element. Formally, the DLin assumption holds for dGG if $\mathbf{Adv}^{\mathrm{dlin}}_{\mathsf{dGG},\mathsf{A}}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{dlin}}_{\mathsf{dGG},\mathsf{A}}(\lambda) = 2\Pr[\mathrm{DLin}_{\mathsf{dGG},\mathsf{A}}(\lambda)] - 1$ and game DLin is defined in Figure 5.

We will make use of the fact that DLin is self-reducible. This means that given a tuple (U, V, S, T, X) one can produce a new tuple (U', V', S', T', X') so that if the original tuple was linear then the new tuple is so too, but with fresh u, v, s and t; if X is random then (U', V', S', T', X') are all independently random as well. In particular, consider the following algorithm that takes input a DLin challenge $(U, V, S, T, X) \in \mathbb{G}^5$.

```
Algorithm $\mathsf{Rnd}(1^\lambda, (U, V, S, T, X))$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$; $z, a, b, c, d \leftarrow_\$ \mathbb{Z}_p$
$U' \leftarrow U^c$; $V' \leftarrow V^d$; $S' \leftarrow S^{cz} U^{ca}$; $T' \leftarrow T^{dz} V^{db}$; $X' \leftarrow X^z g^a g^b$
Return $(U', V', S', T', X')$
```

Let s, t, ξ be such that $S = U^s, T = V^t, X = g^\xi$. Define $s' := sz + a$ and $t' := tz + b$ and note that they are both uniformly random. We have $S' = (U')^{s'}$, $T' = (V')^{t'}$ and $X' = g^{\xi z + a + b} = g^{(\xi - s - t)z + sz + tz + a + b} = g^{(\xi - s - t)z + s' + t'}$. Thus, if the original challenge was a linear tuple (i.e., $\xi = s + t$) then the new tuple is also linear with new randomness uc, vd, s', t', whereas otherwise (i.e., $\xi - s - t \neq 0$) U', V', S', T' and X' are independently random.

THE SCHEME. Our S-ZK scheme is based on a bilinear-group generator dGG, for which we define *linear commitments* to messages $M \in \mathbb{G}$ as follows:

```
$\mathsf{Ln.C}(M; (\vec{u}, \vec{t}))$
Return $\vec{C} = (g^{u_0}, g^{u_1}, g^{u_0 t_0}, g^{u_1 t_1}, g^{t_0 + t_1} \cdot M)$
```

```
$\mathsf{Ln.D}(\vec{u}, (C_2, C_3, C_4))$
Return $M \leftarrow C_4 \cdot C_2^{-1/u_0} \cdot C_3^{-1/u_1}$
```

Commitments are hiding under DLin. Since (C_2, C_3, C_4) is a linear encryption under public key (C_0, C_1), the logarithms of the latter let one recover the message via $\mathsf{Ln.D}$.

```
$\underline{\Pi.\mathsf{Pg}(1^\lambda)}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$t, s_0, s_1 \leftarrow_\$ \mathbb{Z}_p$; $h \leftarrow g^t$
$S_0 \leftarrow g^{s_0}$; $S_1 \leftarrow g^{s_1}$; $S_2 \leftarrow g^{s_0 s_1}$
Return $crs \leftarrow (S_0, S_1, S_2, h)$

$\underline{\Pi.\mathsf{V}(1^\lambda, (S_0, S_1, S_2, h), x, \pi)}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g, h_1, h_2) \leftarrow \mathsf{dGG}(1^\lambda)$
Parse $(\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta) \leftarrow \pi$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then return false
For $i, j = 0, 1$:
    If $\mathbf{e}(C_{0,i}, C_{1,j}) \neq \mathbf{e}(g, D_{i,j})$, return false
Return $\mathsf{Z.V}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), \zeta)$

$\underline{\Pi.\mathsf{P}(1^\lambda, (S_0, S_1, S_2, h), x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$, return $\perp$
$C_{0,0}, \dots, C_{0,4}, C_{1,2}, C_{1,3}, C_{1,4} \leftarrow_\$ \mathbb{G}$
$u_0, u_1 \leftarrow_\$ \mathbb{Z}_p$
$C_{1,0} \leftarrow g^{u_0}$; $C_{1,1} \leftarrow g^{u_1}$
For $i, j = 0, 1$: $D_{i,j} \leftarrow C_{0,i}^{u_j}$
$\zeta \leftarrow_\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (w, \perp))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

Figure 6: NIZK scheme $\Pi[\mathsf{R}, \mathsf{dGG}]$ satisfying SND and S-ZK

We also use a statistically sound NI system with trivial CRS (also called "non-interactive zap" by GOS [38]) $\mathsf{Z} = (\mathsf{Z.P}, \mathsf{Z.V})$ for the following relation:

```
$\underline{\mathsf{R}_Z((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), ((w, (s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)))}$          (6)
If $\mathsf{R}(x, w) = 1$, return 1
If ($g^s = S_0$ or $g^s = S_1$), $\vec{C}_0 = \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$ and $\vec{C}_1 = \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$, return 1
Return 0
```

The NI proof system Z can for example be instantiated by the construction from [38], which does not require a CRS, is perfectly sound and WI under the DLin assumption. Our NIZK system $\Pi[\mathsf{R}, \mathsf{dGG}]$ is given in Figure 6.

Theorem 5.2 *Let* R *be an* **NP** *relation and let* dGG *be a bilinear-group generator. Then* $\Pi[\mathsf{R}, \mathsf{dGG}]$*, defined in Figure 6, satisfies (1) soundness under DH-KEA and CDH; and (2) subversion zero knowledge under DH-KEA and DLin.*

We start with some intuition before giving the proof.

Soundness. Assume an adversary A outputs a proof $\pi = (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$ for a false statement. Since there does not exist a witness w, by statistical soundness of the proof ζ, R_Z must return 1 in the second line in Eq. (6), meaning $\vec{C}_0$ and $\vec{C}_1$ are commitments to either $h^{\log S_0}$ or $h^{\log S_1}$; intuitively, the adversary has thus broken the CDH assumption either for challenge (S_0, h) or (S_1, h).

To make this formal, we construct an algorithm B that on input (g^s, h) outputs h^s with probability close to $\frac{1}{2}$. We first construct four machines $\mathsf{M}_{i,j}$, $0 \leq i, j \leq 1$ that are given given (S, h), set $S_b \leftarrow S$ for a random b, complete this to a CRS, on which they run A; when A returns π, $\mathsf{M}_{i,j}$ outputs $(C_{0,i}, C_{1,j}, D_{i,j})$. By DH-KEA there exist four extractors $\mathsf{E}_{i,j}$ which on input (S, h) and $\mathsf{M}_{i,j}$'s coins (which include A's coins) return either $u_{0,i} = \log C_{0,i}$ or $u_{1,j} = \log C_{1,j}$.

Using $\mathsf{M}_{0,0}, \dots, \mathsf{M}_{1,1}$, we define B: given a CDH challenge (S, h), it picks coins $\bar{r}$ and uses $\bar{r}$ to pick $b \leftarrow_\$ \{0, 1\}$, $s' \leftarrow_\$ \mathbb{Z}_p$ and coins r for A; it sets $S_b \leftarrow S$, $S_{1-b} \leftarrow g^{s'}$ and $S_2 \leftarrow S^{s'}$ and runs A on input (S_0, S_1, S_2, h) and coins r to get π containing $(\vec{C}_0, \vec{C}_1)$; it then runs all $\mathsf{E}_{i,j}$ on input $(S, h, \bar{r})$, which each returns either $u_{0,i} = \log C_{0,i}$ or $u_{1,j} = \log C_{1,j}$. This implies that for some i, B obtains both $u_{i,0}$ and $u_{i,1}$. Using this, B recovers $T \leftarrow \mathsf{Ln.D}((u_{i,0}, u_{i,1}), (C_{i,2}, C_{i,3}, C_{i,4}))$, which it outputs.

```
Game SND_{Π,R,A}(λ)
  $(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$; $t, s_0, s_1 \leftarrow\$ \mathbb{Z}_p$
  $h \leftarrow g^t$; $S_0 \leftarrow g^{s_0}$; $S_1 \leftarrow g^{s_1}$; $S_2 \leftarrow g^{s_0 s_1}$
  $(x, (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)) \leftarrow\$ \mathsf{A}(1^\lambda, (S_0, S_1, S_2, h))$
  Return true if all of the following hold:
    – $x \notin L(\mathsf{R})$
    – $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
    – For all $i, j = 0, 1 : \mathbf{e}(C_{0,i}, C_{1,j}) = \mathbf{e}(g, D_{i,j})$
    – $\mathsf{Z.V}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), \zeta)$
  Return false

Algorithm $\mathsf{M}_{i,j}(1^\lambda, S, h; (b, s', r))$
  $(p, \mathbb{G}, \mathbb{G}_T, e, g, h_1, h_2) \leftarrow \mathsf{dGG}(1^\lambda)$
  $S_b \leftarrow S$; $S_{1-b} \leftarrow g^{s'}$; $S_2 \leftarrow S^{s'}$
  $(x, (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta))$
      $\leftarrow \mathsf{A}(1^\lambda, (S_0, S_1, S_2, h); r)$
  Return $(C_{0,i}, C_{1,j}, D_{i,j})$
```

Figure 7: Soundness game for $\Pi[\mathsf{R}, \mathsf{dGG}]$ and algorithm $\mathsf{M}_{i,j}$

By soundness of ζ, we have either $T = h^{\log S_0}$ or $T = h^{\log S_1}$. Since A has no information on where the challenge S was embedded, B solves CDH with probability $\frac{1}{2}$.

Subversion zero knowledge. By DH-KEA, for every X that outputs a CRS of the form $(g^{s_0}, g^{s_1}, g^{s_0 s_1}, h)$ there exists an algorithm E that extracts either s_0 or s_1. To show S-ZK we first construct a simulator S. Its first part S.crs picks r, runs $crs \leftarrow \mathsf{X}(1^\lambda, r)$ and sets $s \leftarrow\$ \mathsf{E}(1^\lambda, r)$ if crs is correctly formed and $s \leftarrow \perp$ otherwise, and outputs crs, r and the trapdoor $std \leftarrow s$. It is immediate that crs_1 output by X on coins r_1 is indistinguishable from crs_0, r_0 output by S.crs.

We next construct a proof simulator S.pf for statements x under $crs = (S_0, S_1, S_2, h)$ using trapdoor s. Like Π.P it returns $\perp$ if crs is not correctly formed. It chooses $\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1$ and defines $\vec{C}_0$ and $\vec{C}_1$ as commitments to h^s and computes the corresponding elements $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$. Since either $g^s = S_0$ or $g^s = S_1$, S.pf has thus a witness for the statement $(x, S_0, S_1, h, \vec{C}_0, \vec{C}_1) \in \mathsf{R}_Z$, which it uses to compute a proof ζ. The simulator outputs $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$, which we now argue is indistinguishable from a proof output by Π.P under DLin by a series of game hops.

We first note that when constructing ζ, instead of witness $(s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)$ we could use w; this is indistinguishable under WI, which for the GOS system follows from DLin. In the next game hop, we replace $\vec{C}_0$ by a random quintuple and construct the $D_{i,j}$'s as in Π.P; this is indistinguishable under DLin. In the final game hop we replace $\vec{C}_1$ by a random quintuple. This is also reduced to DLin using the fact that we can compute the $D_{i,j}$'s using the logarithms of $\vec{C}_0$. The result is a proof π that is distributed like one output by Π.P.

Proof of Theorem 5.2: Soundness. Let A be a PT adversary breaking soundness. We write out the game and define four algorithms $\mathsf{M}_{i,j}$ for $0 \le i, j \le 1$ in Figure 7.

By the DH-KEA assumption (defined by game KE in Figure 5) applied to each $\mathsf{M}_{i,j}$, there exist PT extractors $\mathsf{E}_{i,j}$ which with with overwhelming probability extract either $\log C_{0,i}$ or $\log C_{1,j}$, that is,

$$\text{For all } 0 \le i, j \le 1 : \mathbf{Adv}^{\text{ke}}_{\mathsf{dGG}, \mathsf{M}_{i,j}, \mathsf{E}_{i,j}}(\cdot) \text{ is negligible} . \tag{7}$$

Consider games G_1, G_2, G_3 and G_4 in Figure 8, where games G_1 and G_3 ignore the boxes in its description, while $\boxed{\mathrm{G}_2}$ and $\boxed{\mathrm{G}_4}$ include the boxes.

Game G_1 differs from $\mathrm{SND}_{\Pi,\mathsf{R},\mathsf{A}}$ in how the CRS is computed. As the CRS is distributed identically in both games, we have

$$\Pr[\mathrm{SND}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)] = \Pr[\mathrm{G}_1(\lambda)] . \tag{8}$$

Since G_1 and G_2 only differ when for some i, j: $C_{0,i} \neq g^{v_{i,j}}$ and $C_{1,j} \neq g^{v_{i,j}}$, while $\mathbf{e}(C_{0,i}, C_{1,j}) =$

```
GAME $G_1$ and $\boxed{G_2}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$S, h \leftarrow\!\!\$\; \mathbb{G}$
$b \leftarrow\!\!\$\; \{0,1\}$; $s' \leftarrow\!\!\$\; \mathbb{Z}_p$
$S_b \leftarrow S$; $S_{1-b} \leftarrow g^{s'}$; $S_2 \leftarrow S^{s'}$
$r \leftarrow\!\!\$\; \{0,1\}^{\mathsf{A.rl}(\lambda)}$
$(x, (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)) \leftarrow \mathsf{A}(1^\lambda, (S_0, S_1, S_2); r)$
For $i, j = 0, 1$:
    $v_{i,j} \leftarrow\!\!\$\; \mathsf{E}_{i,j}(1^\lambda, S, h, (b, s', r))$
Return true if the following hold:
 – $x \notin L(\mathsf{R})$
 – $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
 – For all $i, j = 0, 1$:
        $\mathbf{e}(C_{0,i}, C_{1,j}) = \mathbf{e}(g, D_{i,j})$
        $\boxed{C_{0,i} = g^{v_{i,j}} \text{ or } C_{1,j} = g^{v_{i,j}}}$
 – $\mathsf{Z.V}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), \zeta)$
Return false
```

```
GAMES $G_3$ and $\boxed{G_4}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$S, h \leftarrow\!\!\$\; \mathbb{G}$; $b \leftarrow\!\!\$\; \{0,1\}$; $s' \leftarrow\!\!\$\; \mathbb{Z}_p$
$S_b \leftarrow S$; $S_{1-b} \leftarrow g^{s'}$; $S_2 \leftarrow S^{s'}$
$r \leftarrow\!\!\$\; \{0,1\}^{\mathsf{A.rl}(\lambda)}$
$(x, (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)) \leftarrow \mathsf{A}(1^\lambda, (S_0, S_1, S_2); r)$
For $i, j = 0, 1$: $v_{i,j} \leftarrow\!\!\$\; \mathsf{E}_{i,j}(1^\lambda, S, h, (b, s', r))$

If $(\exists\, j : C_{0,0} = g^{v_{0,j}})$ and $(\exists\, j : C_{0,1} = g^{v_{1,j}})$        (I)
    $T \leftarrow \mathsf{Ln.D}((v_{0,j}, v_{1,j}), (C_{0,2}, C_{0,3}, C_{0,4}))$
If $(\exists\, i : C_{1,0} = g^{v_{i,0}})$ and $(\exists\, i : C_{1,1} = g^{v_{i,1}})$        (II)
    $T \leftarrow \mathsf{Ln.D}((v_{i,0}, v_{i,1}), (C_{1,2}, C_{1,3}, C_{1,4}))$
Else return false                                                                           (III)
Return true if the following hold:
 – $x \notin L(\mathsf{R})$ and $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
 – For all $i, j = 0, 1$:
       $\mathbf{e}(C_{0,i}, C_{1,j}) = \mathbf{e}(g, D_{i,j})$
 – $\mathsf{Z.V}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), \zeta)$
$\boxed{– \mathbf{e}(S_0, h) = \mathbf{e}(g, T) \text{ or } \mathbf{e}(S_1, h) = \mathbf{e}(g, T)}$
Return false
```

Figure 8: Hybrid games in the proof of soundness of $\Pi[\mathsf{R}, \mathsf{dGG}]$

$\mathbf{e}(g, D_{i,j})$ (that is, $\mathsf{E}_{i,j}$ failed), we have

$$\Pr[G_1(\lambda)] - \Pr[G_2(\lambda)] \leq \sum_{i,j=0}^{1} \Pr[\mathrm{KE}_{\mathsf{dGG},\mathsf{M}_{i,j},\mathsf{E}_{i,j}}(\lambda)] \ . \tag{9}$$

We now argue that whenever G_2 returns true then so does G_3. The differences are the box in G_2 and lines (I), (II) and (III) in G_3. Suppose G_2 returns true. Then we have (1a) $g^{v_{0,0}} = C_{0,0}$ or (1b) $g^{v_{0,0}} = C_{1,0}$; (2a) $g^{v_{1,0}} = C_{0,1}$ or (2b) $g^{v_{1,0}} = C_{1,0}$; and (3a) $g^{v_{1,1}} = C_{0,1}$ or (3b) $g^{v_{1,1}} = C_{1,1}$. Suppose we have (1a): if (2a) holds then clause (I) in G_3 is satisfied; otherwise (2b) must hold. If (3a) holds then again (I) in G_3 is satisfied; if (3b) holds then, since we have (2b), clause (II) in G_3 is satisfied. Case (1b) is dealt with analogously. We thus obtain:

$$\Pr[G_3] \geq \Pr[G_2] \ . \tag{10}$$

Game G_4 returns false if T is not of the expected form. Games G_3 and G_4 thus differ when (a1) the logarithms of $(C_{0,0}, C_{0,1})$ or (a2) those of $(C_{1,0}, C_{1,1})$ were extracted (otherwise both games return false), moreover (b) $x \notin L(\mathsf{R})$ and (c) $\mathsf{Z.V}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), \zeta)$, while (d) $(g^t \neq S_0$ and $g^t \neq S_1)$, with t such that $T = h^t$. Suppose (e) there exist $(s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)$ such that:

$$g^s = S_0 \vee g^s = S_1 \ , \tag{11}$$

$$\vec{C}_0 = \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0)) \qquad \text{and} \qquad \vec{C}_1 = \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1)) \ . \tag{12}$$

If (a1) holds then by correctness of linear encryption and Eq. (12), we get that the result of decryption T satisfies $T = h^s$. This, together with Eq. (11) however contradicts (d). Analogously, we get a contradiction if we have (a2). Therefore, (e) does not hold, and together with (b) this yields $(x, S_0, S_1, h, \vec{C}_0, \vec{C}_1) \notin L(\mathsf{R}_Z)$, as defined in Eq. (6). Together with (c), this contradicts soundness of Z.

```
Algorithm $\mathsf{S.crs}(1^\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r \leftarrow\!\!\$\ \{0,1\}^{\mathsf{X.rl}(\lambda)}$
$(S_0, S_1, S_2, h) \leftarrow \mathsf{X}(1^\lambda; r)$
If $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
    then $s \leftarrow\!\!\$\ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r)$
Else $s \leftarrow \perp$
Return $((S_0, S_1, S_2, h), r, s)$
```

```
Algorithm $\mathsf{S.pf}(1^\lambda, (S_0, S_1, S_2, h), s, x)$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g, h_1, h_2) \leftarrow \mathsf{dGG}(1^\lambda)$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ or $s = \perp$ then return $\perp$
$\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1 \leftarrow\!\!\$\ \mathbb{Z}_p^2$
$\vec{C}_0 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$; $\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
For $i, j = 0, 1$: $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$
$\zeta \leftarrow\!\!\$\ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (\perp, (s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

Figure 9: Simulator for S-ZK

Letting A_Z runs the game and output the proof ζ together with its statement (formally defined in Figure 15 in Appendix B), we have thus shown that

$$\Pr[\mathrm{G}_3(\lambda)] - \Pr[\mathrm{G}_4(\lambda)] \leq \Pr[\mathrm{SND}_{\mathsf{Z},\mathsf{R}_Z,\mathsf{A}_Z}(\lambda)] \ . \tag{13}$$

Finally, note that since A's view is independent of the bit b, if G_4 returns true then $\mathbf{e}(S_b, h) = \mathbf{e}(g, T)$ with probability $\frac{1}{2}$. We can thus construct a CDH adversary B (formally specified in Figure 15 in Appendix B) that given (S, h) simulates G_4 and outputs T, which with probability $\frac{1}{2}\Pr[\mathrm{G}_4(\lambda)]$ is a CDH solution for (S, h), thus

$$\tfrac{1}{2}\Pr[\mathrm{G}_4(\lambda)] \leq \Pr[\mathrm{CDH}_{\mathsf{dGG},\mathsf{B}}(\lambda)] \ . \tag{14}$$

Eqs. (8), (9), (10), (13) and (14) together yield

$$\begin{aligned}\mathbf{Adv}^{\mathrm{snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) &= \Pr[\mathrm{G}_1(\lambda)] - \Pr[\mathrm{G}_2(\lambda)] + \Pr[\mathrm{G}_2(\lambda)] - \Pr[\mathrm{G}_3(\lambda)] + \\ &\qquad\qquad \Pr[\mathrm{G}_3(\lambda)] - \Pr[\mathrm{G}_4(\lambda)] + \Pr[\mathrm{G}_4(\lambda)] \\ &\leq \sum_{i,j=0}^{1} \mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{M}_{i,j},\mathsf{E}_{i,j}}(\lambda) + \mathbf{Adv}^{\mathrm{snd}}_{\mathsf{Z},\mathsf{R}_Z,\mathsf{A}_Z}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{dGG},\mathsf{B}}(\lambda) \ ,\end{aligned}$$

which by Eq. (7), the fact that Z is perfectly sound and assuming CDH is hard yields that $\mathbf{Adv}^{\mathrm{snd}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible, as desired.

Subversion zero knowledge. Let X be a CRS subvertor that outputs $(\vec{S}, h)$. Define $\mathsf{X}'(1^\lambda; r)$ that runs $(\vec{S}, h) \leftarrow \mathsf{X}(1^\lambda; r)$ and returns $\vec{S}$. By DH-KEA there exists a PT algorithm $\mathsf{E}_{\mathsf{X}'}$ that if $S_0 = g^{s_0}$, $S_1 = g^{s_1}$ and $S_2 = g^{s_0 s_1}$ for some s_0, s_1 then with overwhelming probability $\mathsf{E}_{\mathsf{X}'}$ extracts s_0 or s_1, that is,

$$\mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},2,\mathsf{X}',\mathsf{E}_{\mathsf{X}'}}(\cdot) \ \text{ is negligible}. \tag{15}$$

Using $\mathsf{E}_{\mathsf{X}'}$ we define a simulator S in Figure 9.

Let A be an arbitrary PT adversary for S-ZK. Writing out game S-ZK for $\Pi, \mathsf{X}, \mathsf{S}$ and A, we obtain the game in Figure 10. (Note that in case $b = 1$ the values $\vec{C}_0 = (g^{u_{0,0}}, g^{u_{0,1}}, g^{u_{0,0}t_{0,0}}, g^{u_{0,1}t_{0,1}}, g^{t_{0,0}+t_{0,1}} \cdot M_0)$ (and likewise $\vec{C}_1$) are random quintuples, so $\Pi.\mathsf{P}$ is correctly simulated. Moreover note that line $(**)$ is redundant, as $s = \perp$ only if $\mathbf{e}(S_{0,0}, S_{0,1}) \neq \mathbf{e}(g, S_{0,2})$; but if so then for $b = 0$ Prove returns $\perp$ before line $(**)$.)

Observe that r_0 and r_1 in $\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}$ are distributed identically and that for a fixed value $b \in \{0,1\}$ the values $r_{1-b}, \vec{S}_{1-b}$ and h_{1-b} are not used anywhere. We can therefore replace every occurrence of $r_0, \vec{S}_0, h_0$ and $r_1, \vec{S}_1, h_1$ by values $r, \vec{S}, h$, respectively.

```
$\underline{\text{GAME S-ZK}_{\Pi,R,X,S,A}(\lambda)}$
$b \leftarrow\!\$ \{0,1\}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r_1 \leftarrow\!\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$
$(\vec{S}_1, h_1) \leftarrow \mathsf{X}(1^\lambda; r_1)$
$r_0 \leftarrow\!\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$
$(\vec{S}_0, h_0) \leftarrow \mathsf{X}(1^\lambda; r_0)$
If $\mathbf{e}(S_{0,0}, S_{0,1}) = \mathbf{e}(g, S_{0,2})$
    $s \leftarrow\!\$ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r_0)$
Else $s \leftarrow \perp$
$b' \leftarrow\!\$ \mathsf{A}^{\text{PROVE}}(1^\lambda, (\vec{S}_b, h_b), r_b)$
Return $(b' = b)$

$\underline{\text{PROVE}(x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
If $\mathbf{e}(S_{b,0}, S_{b,1}) \neq \mathbf{e}(g, S_{b,2})$ then return $\perp$
$\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1 \leftarrow\!\$ \mathbb{Z}_p^2$
For $i, j = 0, 1$: $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$
If $b = 1$ then   // simulate $\Pi.\mathsf{P}$
    $M_0, M_1 \leftarrow\!\$ \mathbb{G}$; $\vec{C}_0 \leftarrow \mathsf{Ln.C}(M_0; (\vec{u}_0, \vec{t}_0))$
    $\vec{C}_1 \leftarrow \mathsf{Ln.C}(M_1; (\vec{u}_1, \vec{t}_1))$
    $\zeta \leftarrow\!\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (w, \perp))$
Else   // simulate $\mathsf{S.pf}$
    If $s = \perp$ then return $\perp$                                   $(**)$
    $\vec{C}_0 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$; $\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
    $\zeta \leftarrow\!\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (\perp, (s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

Figure 10: S-ZK game for $\Pi[\mathsf{R}, \mathsf{dGG}]$

```
$\underline{\text{GAMES G}_0(\lambda), \boxed{\text{G}_1(\lambda)}}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r \leftarrow\!\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$ ; $(\vec{S}, h) \leftarrow \mathsf{X}(1^\lambda; r)$
If $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$: $s \leftarrow\!\$ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r)$
    $\boxed{\text{If } g^s \neq S_0 \text{ and } g^s \neq S_1 \text{ return false}}$
Else $s \leftarrow \perp$
$b' \leftarrow\!\$ \mathsf{A}^{\text{PROVE}}(1^\lambda, (\vec{S}, h), r)$; return $(b' = 1)$

$\underline{\text{PROVE}(x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then return $\perp$
$\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1 \leftarrow\!\$ \mathbb{Z}_p^2$
For $i, j = 0, 1$: $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$
$\vec{C}_0 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$
$\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
$\zeta \leftarrow\!\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (\perp, (s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

```
$\underline{\text{GAMES G}_2(\lambda), \boxed{\text{G}_3(\lambda)}, \boxed{\boxed{\text{G}_4(\lambda)}}}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r \leftarrow\!\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$ ; $(\vec{S}, h) \leftarrow \mathsf{X}(1^\lambda; r)$
If $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$: $s \leftarrow\!\$ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r)$
    If $g^s \neq S_0$ and $g^s \neq S_1$ return $\mathsf{false}$              $(*)$
Else $s \leftarrow \perp$
$b' \leftarrow\!\$ \mathsf{A}^{\text{PROVE}}(1^\lambda, (\vec{S}, h), r)$; return $(b' = 1)$

$\underline{\text{PROVE}(x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then return $\perp$
$\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1 \leftarrow\!\$ \mathbb{Z}_p^2$
For $i, j = 0, 1$: $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$
$\vec{C}_0 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$
$\boxed{M_0 \leftarrow\!\$ \mathbb{G};\ \vec{C}_0 \leftarrow \mathsf{Ln.C}(M_0; (\vec{u}_0, \vec{t}_0))}$
$\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
$\boxed{\boxed{M_1 \leftarrow\!\$ \mathbb{G};\ \vec{C}_1 \leftarrow \mathsf{Ln.C}(M_1; (\vec{u}_1, \vec{t}_1))}}$
$\zeta \leftarrow\!\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (w, \perp))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

Figure 11: Hybrid games in the proof of S-ZK of $\Pi[\mathsf{R}, \mathsf{dGG}]$

In order to show that the cases $b = 0$ and $b = 1$ are indistinguishable, we define a sequence of hybrid games $\text{G}_0, \ldots, \text{G}_4$ given in Figure 11, where $\boxed{\text{G}_3}$ includes the first box and only $\boxed{\boxed{\text{G}_4}}$ includes the double box. The first game G_0 is game S-ZK$_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}$ with the value b fixed to 0, but returning $(b' = 1)$ instead of $(b' = 0)$. We thus have

$$\Pr[\text{G}_0(\lambda)] = 1 - \Pr[\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) \mid b = 0] \tag{16}$$

Game G_1 differs from G_0 if and only if $\mathsf{E}_{\mathsf{X}'}$ fails to extract s_0 or s_1 when X outputs a valid CRS, that is, X' outputs $(g^{s_0}, g^{s_1}, g^{s_0 s_1})$. We have

$$\Pr[\mathrm{G}_1(\lambda)] - \Pr[\mathrm{G}_0(\lambda)] \leq \mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{X}',\mathsf{E}_{\mathsf{X}'}}(\lambda) \ . \tag{17}$$

We define one more game G_5, which is defined as G_4 but without the line $(*)$. Observe that G_5 is actually the original game $\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}$ with b set to 1, so we have:

$$\Pr[\mathrm{G}_5(\lambda)] = \Pr[\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) \mid b = 1] \ . \tag{18}$$

Since game G_4 differs from G_5 only when $\mathsf{E}_{\mathsf{X}'}$ fails (line $(*)$), we have

$$\Pr[\mathrm{G}_5(\lambda)] - \Pr[\mathrm{G}_4(\lambda)] \leq \mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{X}',\mathsf{E}_{\mathsf{X}'}}(\lambda) \ . \tag{19}$$

Game G_2 differs from game G_1 only in which witness is used to compute ζ; games G_2 and G_3 differ in whether $\vec{C}_0$ is an encryption of h^s or random; games G_3 and G_4 differ in the same way for $\vec{C}_1$. In Appendix B we show the following:

Claim 5.3 *There exist adversaries A_{wi} against Z and B, B' against DLin such that*

$$\Pr[\mathrm{G}_4(\lambda)] - \Pr[\mathrm{G}_1(\lambda)] = \mathbf{Adv}^{\mathrm{wi}}_{\mathsf{Z},\mathsf{R}_z,\mathsf{A}_{\mathrm{wi}}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}'}(\lambda) \ . \tag{20}$$

By Eqs. (16) and (18) we have

$$\begin{aligned}
\mathbf{Adv}^{\text{s-zk}}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) &= \Pr[\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) \mid b = 0] + \Pr[\text{S-ZK}_{\Pi,\mathsf{R},\mathsf{X},\mathsf{S},\mathsf{A}}(\lambda) \mid b = 1] - 1 \\
&= \Pr[\mathrm{G}_5(\lambda)] - \Pr[\mathrm{G}_0(\lambda)] \\
&= \Pr[\mathrm{G}_5(\lambda)] - \Pr[\mathrm{G}_4(\lambda)] + \Pr[\mathrm{G}_4(\lambda)] - \Pr[\mathrm{G}_1(\lambda)] + \Pr[\mathrm{G}_1(\lambda)] - \Pr[\mathrm{G}_0(\lambda)] \\
&\leq 2 \cdot \mathbf{Adv}^{\mathrm{ke}}_{\mathsf{dGG},\mathsf{X}',\mathsf{E}_{\mathsf{X}'}}(\lambda) + \mathbf{Adv}^{\mathrm{wi}}_{\mathsf{Z},\mathsf{R}_z,\mathsf{A}_{\mathrm{wi}}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}'}(\lambda) \ ,
\end{aligned}$$

by Eqs. (17), (19) and (20). By Eq. (15) and assuming DLin (which also implies WI of Z), the right-hand side is negligible, as desired. ▌

5.2 Subversion SND and subversion WI

In this section we prove result **P2**: there exists an NI system that is simultaneously SND, WI, S-SND and S-SWI. We call Π an NI system with *trivial* CRS if $crs = \varepsilon$ and $\Pi.\mathsf{P}$ and $\Pi.\mathsf{V}$ ignore input crs. In Lemma 5.4 we observe that if such a Π is SND and WI then it is also S-SND and S-WI. (Intuitively, if the CRS is ignored then there's no harm in subverting it.) In Theorem 5.5 we then notice that an NI system with trivial CRS exists [38] which is SND and WI under the DLin assumption in bilinear groups (defined on p. 17). As in this instantiation the group is chosen by the prover (rather than fixed as for **P1**), it needs to be *verifiable* [38] (that is, one can efficiently check that it is a bilinear group).

Lemma 5.4 *Let R be an* **NP** *relation. Let Π be an NI system with trivial CRS for R. If Π is SND and WI then it is also S-SND and S-WI.*

Proof: Let A be an S-SND adversary. We define B against SND: on input $(1^\lambda, \varepsilon)$, run $(crs, x, \pi) \leftarrow\$ \mathsf{A}(1^\lambda)$ and return (x, π). Since $\Pi.\mathsf{V}(1^\lambda, \varepsilon, x, \pi) = \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$, we have $\Pr[\mathrm{SND}_{\Pi,\mathsf{R},\mathsf{B}}(\lambda)] = \Pr[\text{S-SND}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)]$. Thus, if Π is SND, it is S-SND.

Let A be a WI adversary. Define B against S-WI: on input $(1^\lambda, \varepsilon)$, run $(crs, st) \leftarrow\$ \mathsf{A}(1^\lambda)$; $b' \leftarrow\$ \mathsf{A}^{\mathrm{P{\scriptsize ROVE}}}(1^\lambda, crs, st)$ and return b'; forward A's queries to own oracle (this simulates A's oracle since

```
Π.Pg(1^λ)                      Π.P(1^λ, (σ, m_1), x, w)          Π.V(1^λ, (σ, m_1), x, π)
  $\sigma \leftarrow_{\$} \{0,1\}^{2\lambda}$            $m_2 \leftarrow_{\$} \mathsf{Z.P}(1^\lambda,$                Return
  $m_1 \leftarrow_{\$} \mathsf{Z.V}(1^\lambda)$               $(\sigma, x), (\perp, w), m_1)$      $\mathsf{Z.D}(1^\lambda, (\sigma, x), m_1, \pi)$
  Return $crs \leftarrow (\sigma, m_1)$       Return $\pi \leftarrow m_2$
```

Figure 12: NIZK scheme $\Pi[\mathsf{G}, \mathsf{Z}]$ satisfying SND, ZK and S-WI

$\Pi.\mathsf{P}(1^\lambda, \varepsilon, x, w_b) = \Pi.\mathsf{P}(1^\lambda, crs, x, w_b))$. We have $\Pr[\mathrm{WI}_{\Pi,\mathsf{R},\mathsf{B}}(\lambda)] = \Pr[\text{S-WI}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)]$. Thus, if Π is WI, it is S-WI. ▮

Theorem 5.5 *Let* R *be an* **NP** *relation. If the decision-linear assumption holds for a verifiable bilinear group then there exists an NI system* Π *for* R *that is S-SND and S-WI.*

Proof: Let Π be the NI system presented in [38]. Π is an NI system with trivial CRS satisfying SND and WI under the DLin assumption. By Lemma 5.4 it follows that Π is also S-SND and S-WI. ▮

5.3 Soundness, ZK and subversion WI

We prove result **P3** by presenting an NI system that is SND, ZK, and S-WI.

<u>Zaps.</u> A zap [27] for a relation R is a 2-move protocol (cf. Section 3.4), where the first move is *public-coin* and is generated *independently of the statement* to be proved. Zaps retain soundness and witness-indistinguishability even if the statements are chosen adaptively after the first-move m_1 is fixed. Consequently, the same m_1 can be reused for many proofs. We denote zaps by

$$m_1 \leftarrow_{\$} \mathsf{Z.V}(1^\lambda);\ m_2 \leftarrow_{\$} \mathsf{Z.P}(1^\lambda, x, w, m_1);\ b \leftarrow \mathsf{Z.D}(x, m_1, m_2)\ .$$

Dwork and Naor [27] show that zaps can be constructed from any NIZK in the shared random string model. Concretely, zaps can be based on any family of doubly-enhanced trapdoor permutations, when the underlying NIZK is instantiated with the system of [29].

<u>The scheme.</u> The CRS of our scheme consists of a random bit string σ of length 2λ and the first move m_1 of a zap. A proof consists of the second move of the zap for statement (x, σ), proving that either $x \in L$ or s is the pre-image of σ under a PRG G. The formal description of Π follows.

Let $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a pseudorandom generator and let Z be a zap for the following relation R_Z below. Then NI system $\Pi[\mathsf{G}, \mathsf{Z}]$ is given in Figure 12.

```
R_Z((σ, x), (s, w))
  If $\sigma = \mathsf{G}(s)$ then return 1
  If $\mathsf{R}(x, w) = 1$ then return 1; else return 0
```

Theorem 5.6 *Let* R *be an* **NP** *relation. Let* G *be a length-doubling function and* Z *be a zap for relation* R_Z*. If* G *is pseudorandom and* Z *is sound and witness-indistinguishable then* $\Pi[\mathsf{G}, \mathsf{Z}]$ *is SND, ZK and S-WI.*

Proof: Soundness of Π follows from the soundness of the zap and the fact that the probability that a randomly sampled string σ is in the range of the PRG G is negligible. ZK follows as in

```
GAME Z-WI_{Z,R_Z,B}(λ)                         B(1^λ)
  $b \leftarrow_\$ \{0,1\}$                       $((\sigma, m_1), st) \leftarrow_\$ A(1^\lambda)$
  $(m_1, st) \leftarrow_\$ B_1(1^\lambda)$        Return $(m_1, (\sigma, st))$
  $b' \leftarrow_\$ B_2^{\text{WIPROVE}}(1^\lambda, st)$
  Return $(b = b')$                               $B_2^{\text{WIPROVE}}(1^\lambda, (\sigma, st))$
                                                  Return $b' \leftarrow_\$ A^{\text{PROVE}}(1^\lambda, (\sigma, m_1), st)$
WIPROVE($\bar{x}, \bar{w}_0, \bar{w}_1$)
  If ($R_Z(\bar{x}, \bar{w}_0)$ = false or $R_Z(\bar{x}, \bar{w}_1)$ = false)     B_2's simulation of oracle PROVE($x, w_0, w_1$)
      then return $\bot$                          $m_2 \leftarrow \text{WIPROVE}((\sigma, x), (\bot, w_0), (\bot, w_1))$
  Return $m_2 \leftarrow Z.P(1^\lambda, \bar{x}, \bar{w}_b, m_1)$                  Return $\pi \leftarrow m_2$
```

Figure 13: Game defining WI for zaps (left) and adversary in proof of S-WI of Π

[29]: The ZK simulator picks $s \leftarrow_\$ \{0,1\}^\lambda$, sets the CRS to be $\sigma \leftarrow \mathsf{G}(s)$ and $m_1 \leftarrow_\$ \mathsf{Z.V}(1^\lambda)$. When the simulator is challenged to prove a theorem x, it has a witness for $(\sigma, x) \in \mathsf{R}_Z$ and can therefore compute $\pi \leftarrow_\$ \mathsf{Z.P}(1^\lambda, (\sigma, x), (s, \bot), m_1)$. Indistinguishability of the simulated CRS and proofs follows from the pseudorandomness of G and zap-WI (defined below).

To show S-WI, we prove that from an adversary A winning game S-WI$_{\Pi,\mathsf{R},\mathsf{X},\mathsf{A}}$ we can construct an adversary B winning the WI game of the underlying zap for relation R_Z. We denote this game by Z-WI$_{\mathsf{Z},\mathsf{R}_Z,\mathsf{B}}$ and define it in Figure 13. Note that it reflects the stronger notion of WI where the verifier can obtain several proofs, for theorems of her choice, computed using the same message m_1.

In its first stage B runs A to obtain a CRS consisting of σ and the first message m_1 and returns m_1. B then simulates oracle PROVE(x, w_0, w_1) for A by accessing its own oracle WIPROVE. Figure 13 specifies adversary B. Plugging its description into game Z-WI$_{\mathsf{Z},\mathsf{R}_Z,\mathsf{B}}$, we obtain

```
GAME Z-WI_{Z,R_Z,B}(λ)                               PROVE($x, w_0, w_1$)
  $b \leftarrow_\$ \{0,1\}$                             If $R_Z((\sigma, x), (\bot, w_0))$ = false or
  $((\sigma, m_1), st) \leftarrow_\$ A(1^\lambda)$          $R_Z((\sigma, x), (\bot, w_1))$ = false then return $\bot$
  $b' \leftarrow_\$ A^{\text{PROVE}}(1^\lambda, (\sigma, m_1), st)$     Return $m_2 \leftarrow Z.P(1^\lambda, (\sigma, x), (\bot, w_b), m_1)$
  Return $(b = b')$
```

As this is precisely the description of game S-WI$_{\Pi,\mathsf{R},\mathsf{A}}$, we have

$$\Pr[\text{Z-WI}_{\mathsf{Z},\mathsf{R}_Z,\mathsf{B}}(\lambda)] = \Pr[\text{S-WI}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)] \ . \tag{21}$$

Since Z is zap-WI, $2\Pr[\text{Z-WI}_{\mathsf{Z},\mathsf{R}_Z,\mathsf{B}}(\cdot)]-1$ is negligible and thus by Eq. (21) $\mathbf{Adv}^{\text{s-wi}}_{\Pi,\mathsf{R},\mathsf{A}}(\cdot)$ is negligible, which proves the theorem. ▌

References

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pp. 209–236. Springer, 2010. (Cited on page 3.)

[2] M. Abe, J. Groth, M. Ohkubo, and M. Tibouchi. Structure-preserving signatures from type II pairings. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pp. 390–407. Springer, 2014. (Cited on page 3.)

[3] G. Ateniese, B. Magri, and D. Venturi. Subversion-resilient signature schemes. In I. Ray, N. Li, and C. Kruegel:, editors, *ACM CCS 15*, pages 364–375. ACM, 2015. (Cited on page 7.)

[4] B. Barak, Y. Lindell, and S. P. Vadhan. Lower bounds for non-black-box zero knowledge. In *44th FOCS*, pp. 384–393. IEEE Computer Society Press, 2003. (Cited on page 7, 29.)

[5] M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pp. 194–211. Springer, 1990. (Cited on page 7.)

[6] M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pp. 273–289. Springer, 2004. (Cited on page 16.)

[7] M. Bellare, K. G. Paterson, and P. Rogaway. Security of symmetric encryption against mass surveillance. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 1–19. Springer, 2014. (Cited on page 7.)

[8] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pp. 409–426. Springer, 2006. (Cited on page 8.)

[9] E. Ben-Sasson, A. Chiesa, M. Green, E. Tromer, and M. Virza. Secure sampling of public parameters for succinct zero knowledge proofs. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 287–304. IEEE, 2015. (Cited on page 7.)

[10] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Scalable zero knowledge via cycles of elliptic curves. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pp. 276–294. Springer, 2014. (Cited on page 3, 7.)

[11] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a Von Neumann architecture. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 781–796, 2014. (Cited on page 3, 7.)

[12] D. J. Bernstein, T. Lange, and R. Niederhagen. Dual EC: A standardized back door. Cryptology ePrint Archive, Report 2015/767, 2015. (Cited on page 3.)

[13] N. Bitansky, R. Canetti, O. Paneth, and A. Rosen. On the existence of extractable one-way functions. In D.B. Shmoys, *46th ACM STOC*, pp. 505–514. ACM, 2014. (Cited on page 6, 7, 29.)

[14] M. Blum, A. De Santis, S. Micali, and G. Persiano. Noninteractive zero-knowledge. *SIAM Journal on Computing*, 20(6):1084–1118, 1991. (Cited on page 3, 9.)

[15] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pp. 103–112. ACM Press, 1988. (Cited on page 3.)

[16] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pp. 41–55. Springer, 2004. (Cited on page 15, 17.)

[17] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, ed., *CRYPTO 2001*, volume 2139 of *LNCS*, pp. 213–229. Springer, 2001. (Cited on page 17.)

[18] E. Brier, J.-S. Coron, T. Icart, D. Madore, H. Randriam, and M. Tibouchi. Efficient indifferentiable hashing into ordinary elliptic curves. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pp. 237–254. Springer, 2010. (Cited on page 17.)

[19] R. Canetti, R. Pass, and a. shelat. Cryptography from sunspots: How to use an imperfect reference string. In *48th FOCS*, pp. 249–259. IEEE, 2007. (Cited on page 7.)

[20] M. Chase and A. Lysyanskaya. On signatures of knowledge. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pp. 78–96. Springer, 2006. (Cited on page 7.)

[21] S. Checkoway, M. Fredrikson, R. Niederhagen, A. Everspaugh, M. Green, T. Lange, T. Ristenpart, D. J. Bernstein, J. Maskiewicz, and H. Shacham. On the practical exploitability of Dual EC in TLS implementations. In *USENIX Security*, 2014. (Cited on page 3.)

[22] I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pp. 445–456. Springer, 1992. (Cited on page 16.)

[23] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pp. 566–598. Springer, 2001. (Cited on page 7, 9.)

[24] Y. Dodis, C. Ganesh, A. Golovnev, A. Juels, and T. Ristenpart. A formal treatment of backdoored pseudorandom generators. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pp. 101–126. Springer, 2015. (Cited on page 7.)

[25] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pp. 613–631. Springer, 2010. (Cited on page 7.)

[26] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. (Cited on page 7.)

[27] C. Dwork and M. Naor. Zaps and their applications. In *41st FOCS*, pp. 283–293. IEEE Computer Society Press, 2000. (Cited on page 6, 24.)

[28] A. Escala and J. Groth. Fine-tuning Groth-Sahai proofs. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pp. 630–649. Springer, 2014. (Cited on page 3, 7.)

[29] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st FOCS*, pp. 308–317. IEEE Computer Society Press, 1990. (Cited on page 9, 24, 25.)

[30] O. Goldreich. A uniform-complexity treatment of encryption and zero-knowledge. *Journal of Cryptology*, 6(1):21–53, 1993. (Cited on page 9, 17.)

[31] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991. (Cited on page 9.)

[32] O. Goldreich and Y. Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, 1994. (Cited on page 6, 7, 11, 13, 29.)

[33] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. (Cited on page 9, 11.)

[34] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pp. 444–459. Springer, 2006. (Cited on page 7.)

[35] J. Groth. Short pairing-based non-interactive zero-knowledge arguments. In M. Abe, ed., *ASIACRYPT 2010*, volume 6477 of *LNCS*, pp. 321–340. Springer, 2010. (Cited on page 3, 6, 7, 16.)

[36] J. Groth. Efficient fully structure-preserving signatures for large messages. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pp. 239–259. Springer, 2015. (Cited on page 3.)

[37] J. Groth and R. Ostrovsky. Cryptography in the multi-string model. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pp. 323–341. Springer, 2007. (Cited on page 7.)

[38] J. Groth, R. Ostrovsky, and A. Sahai. Non-interactive zaps and new techniques for NIZK. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pp. 97–111. Springer, 2006. (Cited on page 6, 15, 18, 23, 24.)

[39] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pp. 339–358. Springer, 2006. (Cited on page 9.)

[40] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pp. 415–432. Springer, 2008. (Cited on page 3, 7, 15.)

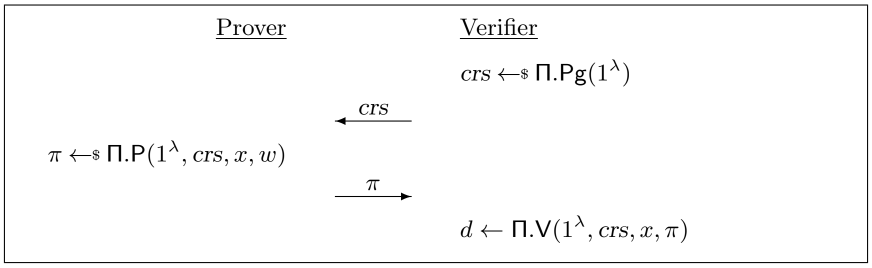
Prover
Verifier
$crs \leftarrow_{\$} \Pi.\mathsf{Pg}(1^\lambda)$
crs
$\pi \leftarrow_{\$} \Pi.\mathsf{P}(1^\lambda, crs, x, w)$
π
$d \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$

Figure 14: Two-move protocol associated to a NI system

[41] S. Hada and T. Tanaka. On the existence of 3-round zero-knowledge protocols. In H. Krawczyk, ed., *CRYPTO'98*, volume 1462 of *LNCS*, pp. 408–423. Springer, 1998. (Cited on page 16.)

[42] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pp. 427–437. ACM Press, 1990. (Cited on page 7.)

[43] J. Patarin and L. Goubin. Asymmetric cryptography with S-boxes. In Y. Han, T. Okamoto, and S. Qing, editors, *ICICS 97*, volume 1334 of *LNCS*, pp. 369–380. Springer, 1997. (Cited on page 7.)

[44] K. G. Paterson. Imprimitive permutation groups and trapdoors in iterated block ciphers. In L.R. Knudsen, editor, *FSE'99*, volume 1636 of *LNCS*, pp. 201–214. Springer, 1999. (Cited on page 7.)

[45] V. Rijmen and B. Preneel. A family of trapdoor ciphers. In E. Biham, editor, *FSE'97*, volume 1267 of *LNCS*, pages 139–148. Springer, 1997. (Cited on page 7.)

[46] A. Shallue and C. van de Woestijne. Construction of rational points on elliptic curves over finite fields. In F. Hess, S. Pauli, and M. E. Pohst, editors, *ANTS-VII*, volume 4076 of *LNCS*, pp. 510–524. Springer, 2006. (Cited on page 17.)

[47] A. Young and M. Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 89–103. Springer, 1996. (Cited on page 7.)

[48] A. Young and M. Yung. Kleptography: Using cryptography against cryptography. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 62–74. Springer, 1997. (Cited on page 7.)

A Relation of NI systems to 2-move protocols

We gave a syntax for two-move protocols in Section 3.4. Completeness, soundness, witness indistinguishability and zero knowledge for such protocols are defined as for any interactive protocols, with the important remark that for each notion there are actually many variant formalizations.

We can associate a 2-move protocol $\mathsf{2MV} = \mathbf{NIto2R}[\Pi]$ to a NI system Π in a natural way as shown in Figure 14. The picture shows the behavior of honest parties: the honest verifier runs $\Pi.\mathsf{Pg}$ to get a *crs* and sends it in the first move, and the honest prover, who has x and $w \in \mathsf{R}(x)$, runs $\Pi.\mathsf{P}$ to get a proof π and sends it back. The honest verifier can take a decision via $\Pi.\mathsf{V}$. It is easy from the picture to define the formal algorithms $\mathsf{2MV.P}, \mathsf{2MV.V}, \mathsf{2MV.D}$ that constitute $\mathsf{2MV}$ as per the syntax above.

The security of Π under subversion is closely connected, but not identical, to certain properties of the associated 2-move protocol. We can roughly think of the following correspondence of notions for Π and $\mathsf{2MV}$: S-WI $\leftrightarrow$ ZAP; ZK $\leftrightarrow$ honest-verifier ZK; S-ZK $\leftrightarrow$ Full (cheating verifier) ZK. In particular subversion of the CRS in the NI setting can be seen as replacing the honest verifier of $\mathsf{2MV}$ with a cheating one who generates the CRS itself. This connection provides guidance and intuition but one must be wary that it is not fully accurate in all cases. In particular, full ZK of $\mathsf{2MV}$

```
$\underline{\text{Adversary } \mathsf{A}_Z(1^\lambda)}$
$(p,\mathbb{G},\mathbb{G}_T,\mathbf{e},g) \leftarrow \mathsf{dGG}(1^\lambda)$
$S,h \leftarrow\$\ \mathbb{G}$
$b \leftarrow\$\ \{0,1\};\ s' \leftarrow\$\ \mathbb{Z}_p$
$S_b \leftarrow S;\ S_1 \leftarrow g^{s'};\ S_2 \leftarrow S^{s'}$
$(x,(\vec{C}_0,\vec{C}_1,\vec{D}_0,\vec{D}_1,\zeta)) \leftarrow\$\ \mathsf{A}(1^\lambda,(S_0,S_1,S_2))$
Return $((x,S_0,S_1,h,\vec{C}_0,\vec{C}_1),\zeta)$
```

```
$\underline{\text{Adversary } \mathsf{B}(1^\lambda,S,h)}$
$(p,\mathbb{G},\mathbb{G}_T,\mathbf{e},g) \leftarrow \mathsf{dGG}(1^\lambda)$
$b \leftarrow\$\ \{0,1\};\ s' \leftarrow\$\ \mathbb{Z}_p;\ r \leftarrow\$\ \{0,1\}^{\mathsf{A.rl}(\lambda)}$
$S_b \leftarrow S;\ S_1 \leftarrow g^{s'};\ S_2 \leftarrow S^{s'}$
$(x,(\vec{C}_0,\vec{C}_1,\vec{D}_0,\vec{D}_1,\zeta)) \leftarrow \mathsf{A}(1^\lambda,(S_0,S_1,S_2;r))$
For $i,j=0,1$: $v_{i,j} \leftarrow\$\ \mathsf{E}_{i,j}(1^\lambda,S,h,(b,s',r))$
If $(\exists\, j : C_{0,0} = g^{v_{0,j}})$ and $(\exists\, j : C_{0,1} = g^{v_{1,j}})$
    Return $\mathsf{Ln.D}((v_{0,j},v_{1,j}),(C_{0,2},C_{0,3},C_{0,4}))$
If $(\exists\, i : C_{1,0} = g^{v_{i,0}})$ and $(\exists\, i : C_{1,1} = g^{v_{i,1}})$
    Return $\mathsf{Ln.D}((v_{i,0},v_{i,1}),(C_{1,2},C_{1,3},C_{1,4}))$
Return $\perp$
```

Figure 15: Adversaries in the proof of soundness of $\mathsf{\Pi}[\mathsf{R},\mathsf{dGG}]$

does not appear to imply S-ZK of $\mathsf{\Pi}$ under any formalization of the former we know, so we cannot get a S-ZK $\mathsf{\Pi}$ directly from known results on 2-move protocols. In 2-move ZK, the simulator for a cheating verifier X gets input $x \in L(\mathsf{R})$ and must then produce a triple (r, crs, π) that is distributed like the view of the cheating verifier X. There are no constraints on how it produces this triple. But in S-ZK, we ask that the simulator produces *crs* without knowing x, so that it cannot depend on a particular instance. Also, once *crs* is produced, based on an associated trapdoor, the simulator must be able to produce a simulated proof π for a given instance x, and moreover be able to do this repeatedly, meaning for many x.

In this sense S-ZK is stronger. Then one may ask about the other direction, namely whether S-ZK of $\mathsf{\Pi}$ implies ZK of $\mathsf{2MV}$. This depends on the particular formalization of the latter used, which enters into understanding the implications of negative results about 2-move ZK. GO [32] showed that 2-move ZK is impossible (meaning only achievable for trivial languages) if the ZK is of the auxiliary-input form. This would rule out auxiliary-input S-ZK but it does not rule out S-ZK as we defined it, where we do not have auxiliary inputs. (One might ask why we do not and say this weakens the definition. The primary motivation for auxiliary inputs in the interactive case was to prove that zero knowledge is retained when proving multiple statements by sequential repetition [32]. But our definition directly requires security for multiple, even adaptively chosen statements. While lack of auxiliary inputs still reduces composability, it is not to the extent that it does in interactive ZK.) Other forms of 2-move ZK are ruled out in BLV [4] but none of these rule out S-ZK. S-ZK does imply some (non-auxiliary-input) forms of 2-move ZK such as used in [13] with the caveat that both definitions be expressed in the same style (either complexity-theoretic or cryptographic).

S-SND seems to have no natural analogue in classical notions for interactive protocols. It is in some sense asking that soundness is maintained in the 2-move protocol $\mathsf{2MV}$ even if the verifier cheats in its first move. SND+S-WI is implied by zaps.

B Additional details for the proof of Theorem 5.2

In the proof of SND, we constructed adversaries A_Z and B, which are formally defined in Figure 15.

In the proof of S-ZK we made the following claim, which we now prove.

Claim B.1 *There exist adversaries A_{wi} against WI of scheme Z and B,B' against DLin such that*

$$\Pr[\mathrm{G}_4(\lambda)] - \Pr[\mathrm{G}_1(\lambda)] = \mathbf{Adv}^{\mathrm{wi}}_{\mathsf{Z},\mathsf{R}_Z,\mathsf{A}_{\mathrm{wi}}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}}(\lambda) + \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}'}(\lambda)\ .$$

```
$\underline{\text{Adversary } \mathsf{A}_{\mathrm{wi}}^{\mathrm{WIPROVE}(\cdot,\cdot,\cdot)}(1^\lambda)}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r \leftarrow_\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$ ; $(\vec{S}, h) \leftarrow \mathsf{X}(1^\lambda; r)$
If $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
    $s \leftarrow_\$ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r)$
    If $g^s \neq S_0$ and $g^s \neq S_1$ return $\perp$
Else $s \leftarrow \perp$
$b' \leftarrow_\$ \mathsf{A}^{\mathrm{PROVE}}(1^\lambda, (\vec{S}, h), r)$; return $b'$
```

```
$\underline{\text{Simulation of A's oracle PROVE}(x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then return $\perp$
$\vec{u}_0, \vec{t}_0, \vec{u}_1, \vec{t}_1 \leftarrow_\$ \mathbb{Z}_p^2$
For $i, j = 0, 1$: $D_{i,j} \leftarrow g^{u_{0,i} u_{1,j}}$
$\vec{C}_0 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_0, \vec{t}_0))$
$\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
$\zeta \leftarrow_\$ \mathrm{WIPROVE}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (\perp, (s, \vec{u}_0, \vec{u}_1, \vec{t}_0, \vec{t}_1)), ((w, \perp)))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

```
$\underline{\mathsf{B}(1^\lambda, U_0, U_1, T_0, T_1, V)}$
$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{dGG}(1^\lambda)$
$r \leftarrow_\$ \{0,1\}^{\mathsf{X.rl}(\lambda)}$ ; $(\vec{S}, h) \leftarrow \mathsf{X}(1^\lambda; r)$
If $\mathbf{e}(S_0, S_1) = \mathbf{e}(g, S_2)$
    $s \leftarrow_\$ \mathsf{E}_{\mathsf{X}'}(1^\lambda, r)$
    If $g^s \neq S_0$ and $g^s \neq S_1$ return $0$
Else $s \leftarrow \perp$
$b' \leftarrow_\$ \mathsf{A}^{\mathrm{PROVE}}(1^\lambda, (\vec{S}, h), r)$; return $b'$
```

```
$\underline{\text{Simulation of A's oracle PROVE}(x, w)}$
If $\mathsf{R}(x, w) = \mathsf{false}$ then return $\perp$
If $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then return $\perp$
$(\vec{U}', \vec{T}', V') \leftarrow_\$ \mathsf{Rnd}(1^\lambda, \vec{U}, \vec{T}, V)$
$\vec{u}_1, \vec{t}_1 \leftarrow_\$ \mathbb{Z}_p^2$
For $i, j = 0, 1$: $D_{i,j} \leftarrow U_{0,i}^{u_{1,j}}$
$M_0 \leftarrow_\$ \mathbb{G}$; $C_{0,0} \leftarrow U_0$; $C_{0,1} \leftarrow U_1$;
$C_{0,2} \leftarrow T_0$; $C_{0,3} \leftarrow T_1$; $C_{0,4} \leftarrow V \cdot h^s$
$\vec{C}_1 \leftarrow \mathsf{Ln.C}(h^s; (\vec{u}_1, \vec{t}_1))$
$\zeta \leftarrow_\$ \mathsf{Z.P}((x, S_0, S_1, h, \vec{C}_0, \vec{C}_1), (w, \perp))$
Return $\pi \leftarrow (\vec{C}_0, \vec{C}_1, \vec{D}_0, \vec{D}_1, \zeta)$
```

Figure 16: Adversaries A_{wi} against WI of Z and B against DLin in the proof of S-ZK of $\Pi[\mathsf{R}, \mathsf{dGG}]$

Proof: Game G_2 differs from game G_1 only in which witness is used to compute ζ. Consider adversary A_{wi} against witness indistinguishability in Figure 16. Note that A_{wi} always calls its WIPROVE oracle with two valid witnesses when simulating PROVE(x, w): if $\mathbf{e}(S_0, S_1) \neq \mathbf{e}(g, S_2)$ then PROVE returns $\perp$ and otherwise $g^s = S_0$ or $g^s = S_1$ (if not, A_{wi} would have returned $\perp$ before running A); moreover, if $\mathsf{R}(x, w) = \mathsf{false}$ then PROVE returns $\perp$. When $b = 0$ in game $\mathrm{WI}_{\mathsf{Z}, \mathsf{R}_Z, \mathsf{A}_{\mathrm{wi}}}$ then A_{wi} simulates G_1 and A_{wi} wins when A outputs 0, that is, when A loses G_1. When $b = 1$ then A_{wi} simulates G_2 and wins when A outputs 1 (and thus wins G_2). When $g^s \neq S_0$ and $g^s \neq S_1$ then A_{wi} returns $\perp$ in which case games WI, G_1 and G_2 all return false. We have $\Pr[\mathrm{WI}_{\mathsf{Z}, \mathsf{R}_Z, \mathsf{A}_{\mathrm{wi}}} \,|\, b = 0] = 1 - \Pr[\mathrm{G}_1(\lambda)]$ and $\Pr[\mathrm{WI}_{\mathsf{Z}, \mathsf{R}_Z, \mathsf{A}_{\mathrm{wi}}} \,|\, b = 1] = \Pr[\mathrm{G}_2(\lambda)]$. Together this yields

$$\Pr[\mathrm{G}_2(\lambda)] - \Pr[\mathrm{G}_1(\lambda)] = \mathbf{Adv}^{\mathrm{wi}}_{\mathsf{Z}, \mathsf{R}_Z, \mathsf{A}_{\mathrm{wi}}}(\lambda) \ . \tag{22}$$

Game G_3 differs from G_2 in whether $\vec{C}_0$ is random or an encryption of h^s. Consider B for game EDLin in Figure 16, which makes use of the algorithm Rnd for self-randomizability. If B receives a linear tuple ($b = 0$ in EDLin) then it simulates G_2 and outputs 0 if $g^s \neq S_0$ and $g^s \neq S_1$, or if A outputs 0, which are the events in which G^2 returns false. We have thus $\Pr[\mathrm{DLin}_{\mathsf{dGG}, \mathsf{B}} \,|\, b = 0] = 1 - \Pr[\mathrm{G}_2(\lambda)]$. If B receives a linear tuple ($b = 1$ in EDLin), it simulates G_3 and outputs 1 if A outputs 1, which is the event in which G_3 returns true; thus $\Pr[\mathrm{DLin}_{\mathsf{dGG}, \mathsf{B}} \,|\, b = 1] = \Pr[\mathrm{G}_3(\lambda)]$. Together this yields

$$\Pr[\mathrm{G}_3(\lambda)] - \Pr[\mathrm{G}_2(\lambda)] = \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG}, \mathsf{B}}(\lambda) \ . \tag{23}$$

Since game G_4 differs from G_3 in how $\vec{C}_1$ is distributed, we could analogously construct an adversary

B' and show that

$$\Pr[\mathrm{G}_4(\lambda)] - \Pr[\mathrm{G}_3(\lambda)] = \mathbf{Adv}^{\mathrm{edlin}}_{\mathsf{dGG},\mathsf{B}'}(\lambda) \ . \tag{24}$$

Eqs. (22), (23) and (24) together now yield the claim. ▮

C Proof sketch for Theorem 5.1

In the "traditional" generic-group model group elements are represented by random strings and an adversary M only has access to operations on them (multiplication of elements in $\mathbb{G}$ and $\mathbb{G}_T$ and pairing of elements in $\mathbb{G}$) via oracles. In particular, M can only produce new group elements by multiplying received elements.

We also need to reflect the fact that by "hashing into the group", one can create a new group element *without knowing its discrete logarithm w.r.t. one of the received elements.* We extend the generic-group model and provide the adversary with an additional operation, namely to request a new group element "independently of the received ones". (And neither the adversary nor the extractor we construct knows its discrete logarithm.)

For DH-KEA the adversary M receives the group elements $(g, h_0 = g^{x_0}, h_1 = g^{x_1})$ and needs to output (S_0, S_1, S_2) where for some s_0, s_1: $S_0 = g^{s_0}$, $S_1 = g^{s_1}$ and $S_2 = g^{s_0 s_1}$. The adversary can produce these group elements by combining the received group elements with newly generated ("hashed") group elements that it has requested. We represent the latter as g^{x_i}, for $i = 2, \dots k$, for some k. The extractor keeps track of the group operations performed by M and thus knows

$$\alpha, \mu_0, \dots, \mu_k, \beta, \nu_0, \dots, \nu_k, \gamma, \xi_0, \dots, \xi_i \in \mathbb{Z}_p \tag{25}$$

such that M's output (S_0, S_1, S_2) is of the form

$$S_0 = g^{\alpha} \textstyle\prod_{i=0}^{k} (g^{x_i})^{\mu_i} \qquad\qquad S_1 = g^{\beta} \prod_{i=0}^{k} (g^{x_i})^{\nu_i} \qquad\qquad S_2 = g^{\gamma} \prod_{i=0}^{k} (g^{x_i})^{\xi_i}$$

(Note that the extractor does however not know $x_0, \dots, x_k$.)

If (I) for all $0 \le i \le k : \mu_i = 0$ then the extractor outputs α. If (II) for all $0 \le i \le k : \nu_i = 0$ then the extractor outputs β. Otherwise, it aborts. It is clear that when (I) or (II) happens then the extractor outputs the logarithm of either S_0 or S_1, as required.

To argue that with overwhelming probability the extractor does not abort, we show that the probability that

$$S_2 = g^{(\log_g S_0)\cdot(\log_g S_1)} \tag{26}$$

holds but neither (I) nor (II) holds is negligible. Taking the logarithms of Eq. (26), we get

$$\gamma + \textstyle\sum_{i=1}^{k} \xi_i\, x_i = \alpha\beta + \sum_{i=1}^{k} \alpha\nu_i\, x_i + \sum_{i=1}^{k} \mu_i\beta\, x_i + \sum_{i,j=1}^{k} \mu_i\nu_j\, x_i x_j \ \ ,$$

which we interpret as multivariate polynomials in $x_0, \dots, x_k$. If neither (I) nor (II) holds then for some i, j we have $\mu_i\nu_j \neq 0$ and thus the polynomial on the RHS is different from that on the LHS. Since the adversary has no information about $x_0, \dots, x_k$ (except for a negligible information leak by comparing group elements, which we ignore), the values in Eq. (25) are generated independently of $x_1, \dots, x_k$. By the Schwartz-Zippel lemma the probability that the two polynomials evaluate to the same for randomly chosen $x_1, \dots, x_k$ is negligible, and therefore so is the probability that Eq. (26) holds.

It follows thus that if Eq. (26) holds then with overwhelming probability the extractor succeeds, which proves the theorem. ▮

	S-SND/S-ZK/S-WI							
SND/ZK/WI	111	110	101	100	011	010	001	000
111	$\boxed{\text{N}}$	✗	$\boxed{\text{N}}$	$\boxed{\text{N}}$	$\boxed{\textbf{P1}}$	✗	$\boxed{\textbf{P3}}$	**P5**
110	✗	✗	✗	✗	✗	✗	✗	✗
101	✗	✗	$\boxed{\textbf{P2}}$	**P4**	✗	✗	**P8**	**P9**
100	✗	✗	✗	**P6**	✗	✗	✗	**P7**
011	✗	✗	✗	✗	**P10**	✗	**P12**	**P11**
010	✗	✗	✗	✗	✗	✗	✗	✗
001	✗	✗	✗	✗	✗	✗	**P13**	**P14**
000	✗	✗	✗	✗	✗	✗	✗	**P15**

Table 1: Achievability of combinations of notions (red marks impossibility, black marks achievability): "✗" marks a trivial impossibility, **N** marks impossibility due to Thm. 4.2. **Pxx** marks achievability: **P1**, **P2**, **P3** are from Theorems 5.2, 5.5 and 5.6; **P4**–**P15** are discussed here.

D Complete relations

We introduced three new security notions for NI systems: S-SND, S-ZK and S-WI and showed in Sections 4 and 5 that some combinations of them with standard notions are impossible while others are achievable. As there are 2^6 possible combinations of the notions SND, ZK, WI, S-SND, S-ZK, S-WI, we now investigate for each of them whether they can be achieved or not. We first note that since (S-)ZK implies (S-)WI and since the subversion-resistant notions imply their standard counterparts, quite a few of the combinations are impossible.

We list all possible combinations in Table 1. The rows correspond to the standard notions SND, ZK and WI and for example 110 means that the first two are satisfied while WI is not. The columns correspond to the subversion-resistant notions S-SND, S-ZK and S-WI. A mark "✗" indicates trivial impossibility, for example the notions in row 110 cannot be satisfied as ZK implies WI.[1]

N indicates the combinations that after trivial exclusions would still be possible, but which we showed impossible in Thm. 4.2, namely all combinations satisfing ZK and S-SND. **P1** corresponds to a scheme satisfying all notions except S-SND and was constructed in Thm. 5.2, **P2** satisfies all notions except ZK and S-ZK and was proved achievable in Thm. 5.5 and **P3** indicates a scheme achieving all notions except S-SND and S-ZK, which was constructed in Thm. 5.6.

We now show that the remaining combinations **P4**–**P15** are all achievable, which completes the picture.

P4. ("**P2** w/o S-WI") Assume the existence of IND-CPA-secure public-key encryption. Now consider the scheme from **P2** and add an encryption key to the CRS and add an encryption of the witness to the proof. As the subvertor can decrypt the witness, the scheme is not S-WI anymore. The scheme is still sound since verification did not change; it is still WI under WI of the original scheme and IND-CPA of the encryption scheme.

P5. ("**P3** w/o S-WI") Similarly, we can remove S-WI from the scheme in **P3**: add a key for public-

[1] In particular from ZK⇒WI we get that all combinations $(*10, ***)$ are impossible; S-ZK⇒S-WI makes $(***, *10)$ impossible; S-SND⇒SND makes $(0**, 1**)$ impossible; S-ZK⇒ZK makes $(*0*, *1*)$ impossible and S-WI⇒WI excludes all combinations $(**0, **1)$.

key encryption to the CRS and add an encryption of the witness to the proof. SND is preserved since verification is unchanged and ZK is preserved since the simulator can add an encryption of 0 to the ciphertext, which is still indistinguishable from real proofs by IND-CPA of the encryption scheme.

P6. A scheme only satisfying the soundness notions is trivial to construct. Define $crs \leftarrow \varepsilon$, a proof π to be the witness and verification to check $\mathsf{R}(x, w)$. The system is S-SND (which implies SND) and not WI (which implies not ZK, not S-WI and not S-ZK).

P7. ("**P6** w/o S-SND") Assume the existence of a length-doubling pseudorandom generator G. To make the scheme from **P6** not satisfy S-SND, set $crs \leftarrow_{\$} \{0,1\}^{2\lambda}$, a proof to be the witness and verification to accept if $\mathsf{R}(x, w) = 1$ or $crs = \mathsf{G}(w)$. A subvertor can choose $t \leftarrow_{\$} \{0,1\}^{\lambda}$, define $crs \leftarrow \mathsf{G}(t)$ and can then prove false theorems by sending t. Soundness still holds since an honestly generated CRS is in the range of G with negligible probability only. Since $\pi \leftarrow w$, the system is not WI.

P8. ("**P2** w/o S-SND") Similarly, we can make the scheme from **P2** not satisfy S-SND: add $s \leftarrow_{\$} \{0,1\}^{2\lambda}$ to the CRS and let verification also accept when it is given a preimage of s under G.

P9. ("**P8** w/o S-WI") To make the above scheme violate S-WI, use the trick from **P4**: add a public key to the CRS and an encryption of the witness to the proof.

P10. To guarantee S-ZK (and therefore S-WI, ZK and WI), the prover outputs $\pi \leftarrow \varepsilon$. To violate S-SND and SND, verification always accepts.

P11. ("**P10** w/o S-WI") To make the above scheme violate S-ZK and S-WI, use the trick from **P4**: add a public key to the CRS and an encryption of the witness to the proof. Note that this preserves the notions ZK and WI.

P12. ("**P3** w/o SND") Take the scheme from **P3** and change verification to always accept. The scheme is clearly not SND. As verification is irrelevant for the ZK and WI notions, we still have ZK, WI, S-WI but not S-ZK.

P13. ("**P2** w/o SND") Consider the scheme from **P2** which is not ZK, not S-ZK, but WI and S-WI. Define verification to always accept. The scheme is clearly not SND; as verification is irrelevant for the ZK and WI notions, these are preserved.

P14. ("**P13** w/o S-WI") Consider the scheme from **P13** and use the same trick as in **P4** to violate S-WI: add a public key to the CRS and an encryption of the witness to the proof.

P15. To violate all notions, set the proof π to be the witness and make verification always accept.