NIZKs with Maliciously Chosen CRS: Subversion Advice-ZK and Accountable Soundness

Prabhanjan Ananth* Gilad Asharov† Vipul Goyal‡ Hadar Kaner§
Pratik Soni¶ Brent Waters‖

Abstract

Trusted setup is commonly used for non-interactive proof and argument systems. However, there is no guarantee that the setup parameters in these systems are generated in a trustworthy manner. Building upon previous works, we conduct a systematic study of non-interactive zero-knowledge arguments in the common reference string model where the authority running the trusted setup might be corrupted.

We explore both zero-knowledge and soundness properties in this setting.

- We consider a new notion of NIZK called *subversion advice-ZK NIZK* that strengthens the notion of zero-knowledge with malicious authority security considered by Ananth, Asharov, Dahari and Goyal (EUROCRYPT'21), and present a construction of a subversion advice-ZK NIZK from the sub-exponential hardness of learning with errors.
- We introduce a new notion that strengthens the traditional definition of soundness, called *accountable soundness*, and present generic compilers that lift any NIZK for interesting languages in NP to additionally achieve accountable soundness.
- Finally, we combine our results for both subversion advice-ZK and accountable soundness to achieve a subversion advice-ZK NIZK that also satisfies accountable soundness. This results in the first NIZK construction that satisfies meaningful notions of both soundness and zero-knowledge even for maliciously chosen CRS.

*Department of Computer Science, UC Santa Barbara. prabhanjan@cs.ucsb.edu
†Department of Computer Science, Bar-Ilan University. gilad.asharov@biu.ac.il
‡School of Computer Science, Carnegie Mellon University and NTT Research. vipul@cmu.edu
§Department of Mathematics, Bar-Ilan University. kanerha@biu.ac.il
¶Kahlert School of Computing, University of Utah. psoni@cs.utah.edu
‖The University of Texas at Austin and NTT Research. bwaters@cs.utexas.edu

Contents

1 Introduction **2**
- 1.1 Our Results . . . 3

2 Technical Overview **7**
- 2.1 Subversion Advice-ZK NIZK . . . 7
- 2.2 Accountable Soundness . . . 10
- 2.3 Related Work . . . 14

3 Preliminaries **15**
- 3.1 One-way Functions . . . 16
- 3.2 Commitment Schemes . . . 16
- 3.3 Non-Interactive Zero Knowledge (NIZK) . . . 17
- 3.4 Witness Indistinguishable (WI) Arguments . . . 17

4 Two Round Advice ZK Arguments for NP **18**
- 4.1 Defining Advice ZK for Two Round Arguments . . . 18
- 4.2 Trapdoor Generation . . . 19
- 4.3 Construction of Two Round Advice ZK Argument . . . 20
- 4.4 Proof of Soundness . . . 21
- 4.5 Proof of Advice ZK . . . 23

5 Subversion Advice-ZK NIZKs **25**
- 5.1 Defining Subversion Advice-ZK NIZKs . . . 25
- 5.2 Construction of Subversion Advice-ZK NIZKs . . . 26
- 5.3 Proof of Theorem 5.3 . . . 27

6 Accountable Soundness **31**

7 Constructions of Non-interactive Arguments with Accountable Soundness **32**
- 7.1 Construction for Languages in NP ∩ co-NP . . . 32
  - 7.1.1 Generalization . . . 34
  - 7.1.2 Examples . . . 34
- 7.2 Construction for Sparse Languages . . . 37
  - 7.2.1 Examples of Sparse Languages . . . 40
- 7.3 Construction for Negligibly Sparse Languages . . . 40
- 7.4 General Feasibility for Accountable Soundness . . . 42

8 Combining Subversion Advice-ZK and Accountable Soundness **46**

A Instantiations of Trapdoor Generation **51**
- A.1 Trapdoor Generation from One-Way Functions with Efficient Recognizable Range . . . 51
- A.2 Trapdoor Generation from Collision-Resistant Hash Family . . . 51

B Subversion Advice-ZK NIZKs satisfy Subversion Witness Hiding, Subversion Function Hiding and More **52**
- B.1 Subversion Witness Hiding (WH) . . . 52
- B.2 Subversion Witness Indistinguishability (WI) . . . 53
- B.3 Subversion Function Hiding . . . 54

1 Introduction

Zero-knowledge (ZK) proofs have transformed modern cryptography. Informally, they allow to prove any NP statement without revealing anything but the statement's validity, and in particular, no information is revealed about the witness. A central question in the study of zero knowledge is the minimal round complexity or whether one can construct a completely non-interactive proof consisting of a single message from the prover to the verifier. Such non-interactive zero-knowledge (NIZK) protocols are possible, but they inherently require some setup [GO94].

The standard setup is the CRS model, where the prover and the verifier share a common reference string (CRS). Who generates the CRS? The most accepted practice is to sample the CRS using some trusted authority. This, however, has a prominent weakness: if one knows the coins used to sample the CRS, or if the CRS was not sampled from the right distribution, then one can either

- *break privacy*, in the sense of extracting secrets from proofs that were honestly generated with respect to the CRS; or,
- *break soundness*, in the sense of providing proofs of false statements, or providing proofs of true statements but without using the corresponding witness in generating the proof.

Whether one can break privacy or not depends on the scheme. As for soundness, for standard NIZK schemes for hard-to-decide languages, there exists an adversary that can generate a malicious CRS and break soundness. Such an adversary is exactly the simulator used for proving that the scheme satisfies the zero-knowledge property. The simulator generates a fake CRS and provides proofs to statements without knowing the witnesses. The fake CRS, together with the proof, are indistinguishable from the honest CRS and honestly generated proofs. Thereby, the simulator can also be used to prove statements that are incorrect but are indistinguishable from correct statements (e.g., proving statements of the form "com is a commitment of 0" while com is a commitment of 1.). This presents an interesting phenomenon, where as part of the proof of security of the scheme (for proving ZK), there is a description of an explicit attacker for that scheme (for breaking soundness).

Bellare, Fuchsbauer, and Scafuro [BFS16] introduced the study of subversion-resistant security for NIZKs, toward understanding the following intriguing question: what security properties are guaranteed if the CRS is maliciously sampled? They show that the ZK property can still be guaranteed (this notion is called "subversion-ZK NIZK", i.e., NIZK that provides ZK when the CRS is subverted). However, the construction uses knowledge-type assumptions, which are non-falsifiable. Constructing subversion-ZK NIZK based on standard cryptographic assumptions seems unlikely: Bellare et al. [BFS16] observe that subversion-ZK NIZKs imply certain forms of two-round ZK protocols, which, again, are only known from knowledge-type assumptions [BCPR14]. For soundness, they showed that no NIZK scheme could guarantee soundness when the CRS is maliciously sampled, as this clashes with the ZK property required when the CRS is honestly generated.

The current state of affairs is, therefore, unsatisfactory. On the one hand, NIZKs for all NP languages have been extensively studied but typically fail to provide security guarantees if the trusted authority is corrupted. On the other hand, the notions of NIZKs addressing corrupted authority are impossible to achieve or are based on non-standard cryptographic assumptions [BFS16]. This leads us to the following question:

> *Is it possible to achieve a non-interactive proof system based on standard (falsifiable) cryptographic assumptions which satisfy meaningful notions of privacy and soundness*

even if the CRS was maliciously generated?

1.1 Our Results

We study NIZKs with subverted CRS and show how relaxed security properties of privacy and soundness can be simultaneously achieved, even when the CRS is maliciously generated. Our constructions are based on standard cryptographic assumptions. Of course, when the CRS is honestly generated, our construction achieves the standard soundness and ZK properties. We tackle privacy and soundness separately and then show how to combine them generically.

Privacy. Our goal is to start with any NIZK which provides the zero-knowledge property when the CRS is honestly sampled and convert it into a NIZK that also achieves some privacy notion even when the CRS is maliciously sampled. As mentioned, achieving full ZK when the CRS is maliciously sampled based on standard cryptographic assumptions seems unlikely. Therefore, we consider relaxations of the ZK property for the case when the CRS is maliciously sampled.

We emphasize that we aim to achieve two notions of privacy simultaneously. (1) Primary – full ZK when the CRS is honestly generated; (2) Subverted – some relaxed notion of privacy when the CRS is maliciously generated (a "fallback"). Several relaxed notions of privacy were previously studied as the primary notion, such as witness indistinguishability or witness hiding. It seems natural to adopt those notions as the fallback guarantee. However, we propose a novel notion, which, as we will see, is strictly more robust than those notions:

Our New Privacy Notion: Subversion Advice-ZK . We compare the view of an adversary in the real world to its view in the ideal world, where:

> **Real:** The adversary generates some (possibly malicious) CRS^*, and then obtains honestly generated proofs computed with respect to CRS^* for statements of its choice (adaptively);
>
> **Ideal:** The adversary generates some (possibly malicious) CRS^*, and then we sample (not necessarily in polynomial time) one-time advice d from some distribution $\mathcal{D}(\mathsf{CRS}^*)$. An efficient simulator $\mathcal{S}$ then generates proofs for adversarially (adaptively) chosen statements using the one-time advice d.

In other words, this notion requires the existence of an *efficient* simulator that might receive some advice that depends on the code of the adversary sampling the CRS and the possibly malicious CRS, *but not the instances to be proven.*

Philosophically, the corrupted authority cannot learn any additional information about the witness from the NIZK proofs, beyond what is leaked by d, where d is generated from the advice distribution. In particular, if the language we are considering is hard against sub-exponential adversaries and moreover, the sampling from the distribution can be done in sub-exponential time then clearly, the NIZK proofs alone cannot help the adversary to recover the witness. Indeed, we show that our definition implies that seeing the NIZK proofs cannot provide any advantage to the adversary in computing *any deterministic predicate or function of the witness* as discussed later in more detail.

In many proofs for NIZK constructions, it is common to plant a trapdoor in the CRS to enable simulation. Could the advice d then be simply this CRS trapdoor of a typical NIZK construction? The answer is no, as standard security definitions for NIZKs provide no guarantee for maliciously chosen CRS.

Our main result for subversion advice-ZK NIZK is the following:

Theorem 1.1 (informal)**.** *Assuming two round delayed-input publicly-verifiable witness indistinguishable arguments for* NP*, NIZK arguments for* NP*, subexponentially-hard collision-resistant hash functions, and non-interactive commitments, then for every* NP *language* L *there exists a scheme* Π *such that:*

- *Primary:* Π *is a NIZK argument system for* L *(i.e., it achieves standard ZK and soundness when the CRS is sampled honestly as described by* Π*);*
- *Subverted:* Π *ensures ZK with advice when the CRS is maliciously sampled.*

Interestingly all the necessary tools for Theorem 1.1 along with the NIZKs can be instantiated from the subexponential hardness of the learning with errors problem. We also show that the privacy guarantee in our notion is *strictly stronger* than other relaxations of ZK, such as witness-indistinguishability and witness-hiding. We also introduce another notion, called "subversion function hiding", which informally means that whatever partial information the verifier, who also controls the CRS, can learn from seeing a proof can be efficiently simulated. We show that subversion advice-ZK also implies this notion.

Our notion is a special case of SPS (super polynomial simulation) ZK, introduced for *interactive* ZK proof systems by [Pas03], in which the simulator is allowed to be inefficient. However a key difference is that for SPS-ZK, the simulator is allowed to run in superpolynomial time for *every* statement. In contrast, we require our simulator to *efficiently* simulate proofs for multiple statements given a one-time advice that is generated via a superpolynomial time computation.

A drawback of SPS ZK is that it is only helpful for languages that are hard against algorithms running in super-polynomial time and, thus, not useful for "not-so-hard" languages. In contrast, we require *efficient* simulation (with an advice possibly generated inefficiently but prior to seeing the statement), which allows our notion to be meaningful also for "not-so-hard" languages. As an example, if the language is not-so-hard and has a unique witnesses, a perfectly acceptable construction for SPS-ZK is to just send the witness in clear (since the SPS simulator would just brute force the witness). This is not possible in our setting.

Accountable Soundness. It has been shown [BFS16] that no scheme can simultaneously achieve ZK in the case of honestly generated CRS, and soundness in the case of maliciously generated CRS. We, therefore, take a different route and use the notion of *accountability*. Accountability in the generation of the CRS was introduced by Ananth, Asharov, Dahari, and Goyal [AADG21]. It guarantees that if the authority misbehaves, one can generate a publicly verifiable proof certifying that the authority is corrupted. The work of Ananth et al. addresses only some specific settings where the authority breaks privacy, i.e., it shows how to hold the authority accountable only if the authority helps others to extract witnesses from honestly generated proofs. The case where the authority helps others to break soundness, that is, helps others to prove false statements, was never explored.

As we already mentioned, ensuring soundness even when the CRS is maliciously generated is important. Any NIZK proof system has an adversary (i.e., the simulator for the ZK property) that can provide proofs to statements without knowing the corresponding witness and therefore, also prove statements that are not in the language (but are indistinguishable from factual statements). To hold the misbehaving authority accountable, we require the following two properties:

1. **Accountability.** Suppose that the authority generated some malicious CRS^*, and then it helps others to prove statements (that are either valid or invalid). The accountability property

guarantees that we can hold such an authority accountable by producing a piece of evidence that can be presented in a court of law to penalize this authority. This is formalized by adding to the NIZK scheme another algorithm, called Judge, which determines if some given piece of evidence indicates that the CRS is corrupted.
This definition is modeled as a game between the authority $\mathcal{A}$ and an extractor $\mathcal{E}$. If the authority generating the CRS is engaged in the aforementioned activity, the goal of the extractor is to generate a string that can be used to implicate the authority. Specifically, upon receiving the CRS from the authority, the extractor can query the authority and ask for proofs of some instances of its choice. The extractor computes, from authority-provided proofs, a piece of evidence to implicate the authority.[1]

2. **Defamation-free.** Accountability cannot stand by itself. We complement the definition by defining another property called *defamation-free*. This definition states that if the CRS is honestly generated, then it is computationally hard to generate a piece of evidence that the Judge would accept.

If a NIZK scheme is a traditional NIZK scheme in the case of an honestly generated CRS, and in addition, satisfies the above two security requirements, then the scheme is a NIZK system with accountable soundness. We show:

Theorem 1.2 (informal)**.** *Let L be a $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ language. Then, every NIZK system for L can be transformed into a NIZK system with accountable soundness.*

Theorem 1.3 (informal)**.** *Let L be a sparse language.[2] Then, every NIZK system for L can be transformed into a NIZK system with accountable soundness.*

It is important to note that the above transformations preserve the efficiency of the proof system. As for Theorem 1.2, we also remark that the language L does not have to be in co-NP; we can also handle languages in which large enough NO instances have a witness of not being in L. We show that the above theorems capture languages and cryptosystems with practical interest. For instance:

- **ZCash:** Consider the proof system that exists in the ZCash system, such as the one for the language POUR – a user pours "old" coins into "new" coins. The statement consists of commitments to hidden values, and we show that such a proof system is captured by a generalization of Theorem 1.2. See Section 7.1.2 for an elaborated discussion.
- **GMW compiler:** A particular usage of NIZK systems is in transforming multi-party computation protocols from semi-honest to malicious security. Such a transformation was first proposed by the GMW compiler [GMW87]. We demonstrate that our results capture NIZK languages used in the GMW compiler by considering the particular case of applying the GMW compiler on Yao's semi-honest two-party protocol [Yao86].
- **Other cryptosystems:** Our theorems can also be applied to commonly used crypto-languages, such as proving that a given tuple is a DDH tuple (a language in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$), proving that a particular string is an output of a pseudorandom generation (a sparse language), or that a particular commitment is a bit commitment (a language in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$). We also show other

[1] We stress that this extractor interacts with the malicious authority online without being able to rewind it. This is because, in the real world, we cannot rewind such an authority.

[2] By "sparse" we mean that $L \subseteq \Sigma$ of some domain Σ, and $|L|/|\Sigma|$ is exponentially small in the security parameter.

examples – validity of ciphertexts, hash-time-lock contracts, and sequential composition of hash in Section 7.1.2 and Section 7.2.1.

General feasibility for accountable soundness. Our results above for accountable soundness are practical and do not slow down the proof system's process. Yet, they assume some structure in the language. We also show a general feasibility result for any NP language from subexponential hardness assumptions. We refer the reader to the technical overview for further details.

Putting it all together. As opposed to the negative result by [BFS16], our two notions can co-exist. Thus, we can have a NIZK scheme that provides subversion advice-ZK and accountable soundness for the case where the CRS is maliciously sampled. Our work is in the form of general compilers: we take an existing NIZK and uplift it to achieve this extra security. NIZKs can be instantiated from various assumptions, including factoring [FLS90], falsifiable assumptions on bilinear maps [CHK03, GOS06], and from the subexponential hardness of LWE [CCH+19, PS19]. This gives the first construction of NIZKs, providing reasonable notion of privacy in the subversion setting from post-quantum assumptions. Comparatively, prior works [BFS16, AADG21] on constructing NIZKs that offer security for maliciously chosen CRS are based on bilinear maps and are susceptible to quantum attacks.

When can our notions be applied? When using such notions it is imperative for a system designer to realize when they can and cannot be successfully applied. We examine accountable soundness and zero knowledge in turn. Our notion of soundness guarantees security against a corrupt authority that will willingly create false proofs for any statement as a service.

In practice to hold such an attacker accountable we need two things to occur. First, the attacker needs this service just enough that he will likely interact with someone willing to turn him in. If an authority works to help a small cabal or even a single user cheat, it might be difficult to expose the bad behavior. On the other hand, if the corrupted authority runs an open service or say is willing to help for a fee, he might be more likely to be caught. Arguably, this is somewhat similar to the problem of traitor tracing [CFN94] which shows how to trace the origin of a "pirate" decrypting box or algorithm. But this only works if the decoding algorithm is spread widely enough where it gets into the hands of a user that wishes to expose the corruption. Nonetheless, in our setting even if the authority helps a single user cheat, he risks leaving a proof of bad behavior with that user (which that user can exploit later on, e.g., by blackmailing the authority).

Second, one needs to ask what service does the authority need to provide in order to subvert the security of our system. Running a service that will create a proof for any submitted statement will almost certainly subvert security for most conceivable scenarios. However, it might be possible for a more circumspect authority to subvert security by being more judicious in what statements it will create proofs for. For example, the authority might only create a proof for a statement x under certain conditions. The goal of such an authority is then to provide a service that gives enough to help undermine the security of the larger system, but the service is limited in such a way that he will not be held accountable.

This leads us to the following viewpoint. It is not prudent to simply replace a NIZK with an accountable one in a system and presume that accountability will follow. Instead, one should define the desired security property of the larger system and then try to prove that if security is violated it will lead to the authority being held accountable. We believe that in some systems our notion will be sufficient and in others it might not. For cases where it falls short we expect it will lead to interesting open problems for strengthening accountability.

On the zero knowledge side it is instructive to compare our subversion notion of zero knowledge to the work of Barak and Pass [BP04] who show how to achieve one message zero knowledge against uniform attackers. At some level a one message zero knowledge system achieves zero knowledge against a corrupted authority simply by the virtue of having no authority. The main restriction on applying our subversion notion is that it is only applicable in a security game where the attacker/authority will publish the CRS at the beginning of the security game. This will not apply in a situation where the publication of the CRS can possibly be delayed or depend on other inputs in the game. (Since in this situation one cannot non-uniformly pre-compute the trapdoor advice for the CRS.) We get the usual (security against non-uniform attackers) notion of soundness when the authority is not corrupted and either no or accountable soundness (depending if the additional transformation is applied) if the authority is corrupted. In contrast the Barak-Pass system does not have such a restriction since there is no CRS. However, it only maintains soundness against uniform attackers and requires less standard assumption of the "keyless" flavor such as keyless collision resistant hash functions.

Open problems. We believe that a systematic study of the notions of accountability in the CRS model is an exciting line of research. Our work leaves open some interesting problems. In our definition of accountable soundness, we considered the most basic setting where the extractor has full control over the statement. That is, we capture only the setting where the authority runs some service in which it receives queries x and replies with proofs π without ever seeing the corresponding witnesses. Already addressing this basic setting is challenging and requires interesting technical work.

In reality, the authority might only answer limited types of statements, or with each query of some statement x, it might be willing to answer only on statement $f(x)$ for some function f. It is intriguing to understand under what functions f achieving accountable soundness is possible.

As for subverted advice ZK NIZK, our results are limited to NIZKs only, while exploring analogous definitions for two-party and multiparty computation is an interesting future direction.

2 Technical Overview

In this section, we give an overview of our techniques where Section 2.1 presents our new notion of subversion advice-ZK, and Section 2.2 discusses accountable soundness.

2.1 Subversion Advice-ZK NIZK

Let us recall the setting. We have an authority $\mathcal{A}$ that samples a malicious common reference string CRS^*. We want to ensure that some meaningful notion of privacy prevails for proofs of statements computed using CRS^*. We propose the following definition. For every authority $\mathcal{A}$, there exists a distribution $\mathcal{D}$ (not necessarily efficient to sample from) along with an efficient PPT simulator $\mathcal{S}$ such that the view of the adversary is computationally-indistinguishable in the following two processes:

- **Real**: The adversary generates some (possibly malicious) CRS^* and then obtains honestly generated proofs proven with respect to CRS^* for statements chosen by the adversary;
- **Ideal:** The adversary generates some (possibly malicious) CRS^*. We sample $d \leftarrow \mathcal{D}(\mathsf{CRS}^*)$. The adversary repeatedly gives some statements in which an efficient simulator $\mathcal{S}$ generates proofs with the help of the advice d and with respect to the flawed CRS^*.

We refer to a NIZK that additionally satisfies the above property as *subversion advice-ZK NIZK*. See Section 5.1 for a formal definition. We emphasize that the distribution $\mathcal{D}$, although inefficient to sample from, is independent of the statement/witness. In fact, in Definition 5.1, we explicitly require sampling from $\mathcal{D}$ to run in some fixed superpolynomial time $S(\cdot)$, where S is function of the security parameter λ but independent of the hardness of L.[3]

A discussion. How well does this notion protect the witness w? To get some sense of the privacy guarantee, we compare it to several other privacy notions. Those notions were studied as the primary security notion, i.e., the privacy guarantee when the CRS is honestly generated. In the following, we adopt those notions as the fallback guarantee while requiring NIZK as the primary notion:

- **Subversion Witness-Indistinguishability**: Witness-Indistinguishability (WI) [FS90] means that one cannot distinguish between proofs that were generated by different witnesses. In subversion WI [BFS16], we require standard ZK in case of honestly generated CRS, but when the CRS is maliciously generated, then two proofs generated with different witnesses are indistinguishable. This notion is meaningless for languages where each instance has a unique witness (in which case, a proof system that completely reveals the witness satisfies this notion). Moreover, subversion WI allows leaking bits of the witness (as long as those bits do not help to identify which one of the witnesses was used), whereas ours does not allow such a leakage. Our notion implies subversion witness indistinguishability, and is strictly stronger. We refer to Appendix B for a formalization of this argument.
- **Subversion Witness Hiding**: Witness hiding (WH) [FS90] ensures that one cannot learn the entire witness from the proof (unless such a witness can be efficiently computed from the statement alone). In subversion WH, we require that this holds even if the CRS is maliciously generated. However, the proof can reveal some bits of the witness as long as they do not allow for efficiently recovering the witness. Our notion implies subversion WH, and this notion is weaker than ours. We formalize this argument in Appendix B.
- **Subversion Super-Polynomial Simulation (SPS) NIZK**: Introduced for interactive protocols in the plain model, the notion of SPS-ZK [Pas03] allows the simulator to run in some fixed super-polynomial time *for every statement*. Therefore, in the subversion setting, one can also consider a NIZK that satisfies subversion SPS-ZK as the fallback guarantee. This is a natural security guarantee; and our notion implies it in a strong sense albeit in the non-black-box simulation setting since the advice distribution could depend on the code of the corrupt authority. In particular, our simulator can efficiently simulate proofs for multiple statements when given a one-time advice string generated via some super-polynomial time computation.

In Appendix B, we introduce a new notion called "function hiding" which is a strengthening of witness hiding. Roughly, function hiding requires that the NIZK should not give any advantage to the adversary in guessing *any* deterministic function or predicate of the witness. The probability of such a guess being correct should remain similar before and after seeing the NIZK. We show that subversion advice-ZK for NIZKs implies subversion function hiding.

On the strength of our definition. Philosophically, our notion enables that whatever the adversary learns from multiple proofs from the honest prover, it could have computed on its own by running in time $S(\lambda)+\mathsf{poly}(\lambda)$ on the CRS where S is some a-priori fixed superpolynomial function.

[3] In our construction, S can be set to be $2^{\omega(\log \lambda)}$ assuming 2^{λ^ϵ}-hardness of either a one-way permutation or a collision-resistant hash function.

Perhaps, the S-time computed advice given to the simulator (akin to preprocessing of the language) may reveal some information to the adversary. But, similarly to the case of SPS-ZK for interactive protocols [Pas03], the exact information revealed and its impact on security depends on several factors, including the hardness of the language, the adversary, and the considered application. For example, consider an adversary that embeds some "hard" instance x^* in the CRS, then the S-time computed advice could reveal a witness w^* for x^*. What meaningful guarantees does our notion provide for the adversary's chosen statement x? We elaborate this below:

1. When the statement x is chosen independently of the malicious CRS and the language L is hard-on-the-average for polynomial-time algorithms, then our notion essentially says that any information the verifier learnt by talking to the prover can be simulated by first performing a super-polynomial time instance-independent processing (i.e., this phase does not depend on x) and then running a PPT algorithm on x.
2. The statement x being dependent on the CRS requires more care. The extreme case is where $x = x^*$, i.e., the statement is the statement embedded in the CRS. In that case, our real-world proofs may completely reveal the witness $w = w^*$, if the language L has unique witnesses. In that case and for that particular instance, our notion provides no meaningful guarantee. Such a similar weakness also exists in SPS-ZK for interactive protocols in the plain model, where the super-polynomial time simulator might learn a witness of a hard instance embedded by the verifier in its messages.
   On the other hand, even in the extreme case of $x = x^*$, but when instances in L have multiple witnesses, there is still some privacy guarantee. This is essentially captured by the fact that our notion implies subversion witness-indistinguishability.

Subversion Advice-ZK NIZK construction. We now show how to construct subversion advice-ZK NIZK. We reduce this goal to a slightly weaker building block, which is a two-round argument system $(\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ that we introduce:

- $\mathsf{V}_0(1^\lambda)$: The verifier sends the first message zk_1. We require that the system be *delayed input*; that is, zk_1 is independent of the statement x.
- $\mathsf{P}(x, \mathsf{zk}_1, w)$: The prover sends the second message zk_2, that depends on zk_1 and x.
- $\mathsf{V}_1(x, \mathsf{zk}_1, \mathsf{zk}_2)$: the verifier decides whether to accept or reject. We also require that this scheme is *publicly verifiable*, namely, that the verifier does not keep a secret state after its first message, and so everyone can verify the proof given the transcript $(\mathsf{zk}_1, \mathsf{zk}_2)$.

The hiding requirement of this proof system is similar to subversion advice-ZK :

- advice-ZK: The hiding property is that for every corrupted verifier V^*, there exists a (not necessarily efficient) distribution $\mathcal{D}$, and an efficient simulator $\mathcal{S}$, such that the following holds. Given a sample $d \leftarrow \mathcal{D}$, the simulator can generate transcript $\mathsf{zk}_1, \mathsf{zk}_2$ for a statement x that is indistinguishable from an execution of the protocol with an honest prover and with the corrupted V^*.

Given such a primitive, it seems immediate to convert it to a NIZK proof system: Simply run $\mathsf{V}_0(1^\lambda)$ to generate zk_1 and treat it as the common reference string; To prove that a statement x is in the language we run the honest prover P on (x, zk_1) and obtain $\pi = \mathsf{zk}_2$. To verify the proof, run the verifier V_1 on $(x, \mathsf{zk}_1, \mathsf{zk}_2)$. Moreover, the subversion advice-ZK property for the case of a maliciously generated CRS follows directly from the advice-ZK property of the two-round scheme.

However, the aforementioned proof system is not a NIZK. Specifically, while it provides the "fallback" guarantee when the CRS is maliciously generated, its primary privacy notion, i.e., the privacy guarantee when the CRS is honestly generated, is not full ZK. In that above construction, there is no guarantee that the simulator would be efficient without the advice since the underlying two-round scheme satisfies just ZK with advice. The simulation in ZK with advice includes the non-efficient sampling of the advice d. To solve this problem, we wrap the construction with an inner NIZK argument of knowledge scheme $(\mathsf{GenCRS}_{\mathsf{Inner}}, \mathsf{P}_{\mathsf{Inner}}, \mathsf{V}_{\mathsf{Inner}})$ as follows:

- To generate the CRS, we generate $\mathsf{CRS}_{\mathsf{Inner}}$ according to the inner NIZK scheme, and zk_1 according to the two-round accountable ZK scheme. The CRS is therefore $(\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$.
- Given (x, w), the prover computes the second message zk_2 of the two-round scheme using (x, w) and zk_1. Then, it generates using the inner NIZK scheme a proof π for the statement "I know zk_2 for which the $\mathsf{V}_1(\mathsf{zk}_1, \mathsf{zk}_2)$ accepts". It outputs the proof π.
- To verify a proof, we simply run the verifier of the inner NIZK scheme.

It is easy to see that this scheme satisfies completeness and soundness. Moreover, ZK with advice in the case of malicious CRS, follows easily from the ZK with advice property of the two-round scheme. However, now we also have an efficient simulator for the case of an honestly generated CRS: We can use the simulator of the inner NIZK scheme to generate proofs for the statement that the prover knows zk_2 such that V_1 accepts $(\mathsf{zk}_1, \mathsf{zk}_2)$, even without knowing zk_2. Thus, the simulator does not need to know the witness w to generate zk_2.

Therefore, to obtain subversion advice-ZK NIZKs, all that is left to show is how to construct a two-round ZK with advice scheme.

Construction of Two Round ZK Argument. At a high level, the construction follows the template of Pass's two-round SPS-ZK protocol [Pas03]. We quickly recall their construction: the verifier message consists of an image $y = f(s)$ of a one-way permutation f. The prover on input a statement witness pair (x, w) computes a non-interactive commitment $\mathbf{c}$ to the all-zero string and computes a non-interactive witness-indistinguishable proof for the statement "either $x \in L$ or $\mathbf{c}$ commits to the pre-image of f". The super-polynomial-time simulator for this construction first receives the verifier message y and brute-force inverts y to get the corresponding pre-image s and then uses s to finish the simulation.

Here, we observe that their super-polynomial-time simulator can be decomposed into an inefficient distribution $\mathcal{D}$ and an efficient simulator $\mathcal{S}$ where the inefficient distribution $\mathcal{D}$ is as follows: it runs the verifier on a uniform random tape r^* to get the first message y^* and then runs in super-polynomial time to brute-force invert y^* to compute the pre-image s^*. Then, it outputs (r^*, s^*). Now, the simulator $\mathcal{S}$ on input (r^*, s^*) and statement x can compute a commitment $\mathbf{c}$ to s^* along with the witness-indistinguishable proof using witness s^*.

While the above two-round construction is sufficient to get ZK with advice, it requires one-way permutations and non-interactive witness-indistinguishable proofs (NIWIs). In Section 4, we present a generalization of this protocol that allows us for more general instantiations, including a post-quantum instantiation from the subexponential hardness of learning with errors problem.

2.2 Accountable Soundness

We now turn our attention to soundness. When the CRS is generated by a corrupted authority, it could exploit the randomness used in creating the CRS to generate proofs for false statements

(or even true statements but without actually knowing what the witness is). We recall that one cannot hope to provide subversion-soundness for a NIZK, as this is impossible by the negative result by [BFS16]. Therefore, it is always possible to generate proofs for false statements when the CRS is maliciously sampled. While we cannot prevent this behavior, we can at least hope to implicate authorities who engage in this activity, if they ever actively use this knowledge to, for instance, sell proofs of false statements.

Definition. Recall that our definition (which is inspired by the accountable NIZK of Ananth et al. [AADG21]) changes the syntax of the NIZK proof system, by adding a new algorithm, called Judge, in addition to the standard algorithms of Gen, Prove, Vrfy. The Judge algorithm receives as an input (possibly malicious) CRS^* together with some transcript τ and has to decide whether the CRS^* is corrupted or not. In addition,

- **Accountability:** accountability is modeled as a game between the authority $\mathcal{A}$ and an extractor $\mathcal{E}$. If the CRS generating authority is engaged in the aforementioned activity in the "real world", then there exists an extractor in an "ideal world," where the goal of the extractor is to generate a transcript τ that implicates the authority. The extractor, upon receiving the CRS from the authority, can query the authority and ask for proofs of instances of its choice. The extractor uses the information produced by the authority-provided proofs to compute evidence τ for which Judge outputs $\mathsf{corrupted}$.
- **Defamation-free:** As mentioned, the accountability property alone does not suffice, and so we augment this property with defamation-freeness. This roughly states that if CRS is honestly generated according to GenCRS, then no adversary $\mathcal{A}(\mathsf{CRS})$ can output τ such that $\mathsf{Judge}(\mathsf{CRS}, \tau)$ outputs $\mathsf{corrupted}$. I.e., no adversary can produce evidence that implicates an honest authority.

Construction for languages in NP ∩ co-NP. We start with a simple example. Consider for instance the language of DDH tuples, namely, consider some cyclic group $\mathbb{G}$ of order q with generator g where the DDH problem is believed to be hard. Then, consider the following language over $\mathbb{G} \times \mathbb{G} \times \mathbb{G}$:

$$L = \{(h_1, h_2, h_3) \mid \exists x, y \in \mathbb{Z}_q \quad \text{s.t.} \quad h_1 = g^x, \quad h_2 = g^y, \quad h_3 = g^{xy}\} \ .$$

Since for every $h \in \mathbb{G}$ there exists a unique $x \in \mathbb{Z}_q$ such that $h = g^x$, we can conclude that the complement language $\overline{L}$, which consists of tuples of the form (g^x, g^y, g^z) where $z \neq xy$ is also in NP. Now, consider a proof system for this problem, and assume that one generated the CRS maliciously such that it can, given an instance $(h_1, h_2, h_3) \in L$, generate proof π which is accepted by a verifier (even though this algorithm does not receive the witness x and y).

Assuming DDH, this authority cannot distinguish whether a given (h_1, h_2, h_3) is in L or in $\overline{L}$. To hold the authority accountable, it is enough to sample some triplet $(h_1, h_2, h_3) \notin L$ (say, by sampling x, y, z and then set $h_1 = g^x$, $h_2 = g^y$, and $h_3 = g^z$), use the authority to obtain a valid proof π that "shows" that the instance is in L, and then publicize the proof π together with the witness (x, y, z) that shows that the instance is not in L. From the soundness property of the NIZK proof system, the only way to obtain the accepting proof π for an instance $(h_1, h_2, h_3) \notin L$ is by using a malicious CRS. This implies that the authority generated the CRS is corrupted.

To be slightly more formal, in the above scheme, we do not modify the way the CRS is being generated, nor the code of the prover or the verifier. The Judge algorithm receives a (possibly maliciously generated) CRS and a transcript τ and should decide whether the CRS is corrupted.

In our case, the transcript τ consists of an instance (h_1, h_2, h_3), a proof π that is accepted by the scheme, and a witness $\overline{w}$ (x, y, z) showing that $(h_1, h_2, h_3) \notin L$. The judge algorithm outputs corrupted if indeed π is accepted but $\overline{w}$ validates that $(h_1, h_2, h_3) \notin L$.

Since the construction does not modify the generation of the CRS, the defamation-free property follows directly from the soundness property of the scheme. No adversary can generate an acceptable proof for an instance that is not in the language, and therefore it is impossible to frame an innocent authority that generated the CRS honestly. Importantly, the resulting scheme is practical, we do not modify the construction nor the CRS generation, and one can even use our paradigm on a previously generated CRS that is currently in use.

Distribution over the inputs. Before proceeding to other results, we mention some property of our definition. To have a meaningful security notion, we have to model the fact that the authority does not know the witness of a given instance. In particular, if the authority has some auxiliary information about the statement, then it can possibly produce the proof π honestly (i.e., using the witness) and without forging it. To avoid this issue, we specify a distribution $\mathcal{D}$ such that the inputs in the security experiment are sampled from this distribution. We stress that this requirement is only for the accountability security experiment, and the NIZK construction satisfies the usual definition of NIZK and is well-defined for any input.

Note that the above requires the extractor also to have some distribution $\mathcal{D}'$ for which it generates the instance $(h_1, h_2, h_3) \notin L$ used as part of the evidence. Moreover, in our example, the distribution $\mathcal{D}$ might be picking x and y at random and setting $(h_1, h_2, h_3) = (g^x, g^y, g^{xy})$, whereas the distribution that the extractor uses would be picking x, y, z at random and setting $(h_1, h_2, h_3) = (g^x, g^y, g^z)$. Note that according to the DDH assumption, the malicious authority cannot distinguish between samples of the two distributions, and if it ever produces proofs without knowing the witness, even if it intends to do so only for statements that are in the language, then it can also be used to generate proofs for statements that are not in the language, thereby holding it accountable.

Additional results for accountable soundness. Motivated by the simple example of DDH, we essentially show that accountable soundness can be achieved for any hard language in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$. By a hard language, we mean the following: it should be computationally hard to distinguish yes instances from no instances. Essentially, the extractor generates a NO instance, receives a valid proof for that instance, and uses the instance, the witness showing that this is indeed a NO instance, and the proof generated by the authority – to frame the authority.

We then generalize this condition and show that it also applies to languages that are not in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$, but for which there exists an efficiently checkable witness for some (as opposed to all) NO instances. We show that this already gives a powerful framework and captures several interesting languages that are used in cryptographic systems, such as non-interactive commitments, the GMW compiler [GMW87], validity of ciphertexts, hash-time-lock contracts [ZKC], the ZCash's POUR transaction [BSCG+14]. See Section 7.1.2.

We then turn our attention to languages for which NO instances may not have a short validating proof of non-membership. For example, consider the language L_G, which contains strings in the range of some length doubling pseudorandom generator G, that is, $L_G = \{y \in \{0,1\}^\ell : \exists\ s \in \{0,1\}^{\ell/2} \text{ s.t. } y = G(s)\}$. Then, for any ℓ-bit string $\overline{y}$ there does not (seem to) exist an efficiently checkable witness for $\overline{y} \notin L_G$. As a first step towards achieving accountable soundness for such languages, we consider "sparse languages". We also discuss a generalization to any NP language under strong assumptions on the distribution $\mathcal{D}$.

Sparse Languages. We show that if the language is "sparse", namely, the number of elements in $|L|$ is only a negligible fraction, then it is also possible to hold the authority accountable. Here, however, we have to modify the way the CRS is generated. In particular, if L is over ℓ-bit strings then we add an ℓ-bit random string x^* to the CRS. If, for instance, L had only $\delta \cdot 2^\ell$ instances, then for every $x \in L$ it is only with δ probability that $x \oplus x^*$ belongs to the language (over the choice of x^*). By a union bound over all $x \in L$, the probability over the choice of a random x^* that there exists some $x \in L$ such that $x^* \oplus x \in L$ is at most $\delta^2 \cdot 2^\ell$, which is negligible for $\delta = 2^{-\ell/2-\omega(\log \lambda)}$. As such, we define the Judge to require an acceptable proof for $x \oplus x^*$ for some $x \in L$ to deem the CRS that contains x^* as being corrupted.

Defamation-freeness holds since given an honestly generated CRS that contains a uniformly random x^*, any adversary $\mathcal{A}$ that convinces the above Judge algorithm must find an accepting proof for $x \oplus x^*$ for some $x \in L$. But we just argued that except with negligible probability $x \oplus x^*$ is not in L. Therefore, we can use $\mathcal{A}$ to break the adaptive soundness of the underlying NIZK. Additionally, we can show accountability for any distribution $\mathcal{D}$ on yes instances, which is computationally indistinguishable from the uniform distribution over ℓ bit strings. Section 7.2 discusses the case of such "subexponentially sparse" languages, and we defer the case of negligibly sparse languages to Section 7.3.

Languages captured by this approach include – outputs of pseudorandom generators or sequential composition of hash, which is a prominent benchmark for designing time- and space-efficient arguments for RAM computations [BHR+20, BHR+21], and for proving knowledge of a T-sized blockchain. See section 7.2.1.

General Feasibility Result. The main idea of going beyond sparse languages is to push the sparsity requirements onto the distribution $\mathcal{D}$ from which the statements are sampled from. That is, even though the language itself is not sparse, we artificially consider a subset of the language which is sparse. The main idea is that if $\mathcal{D}$ needs $r(\lambda)$-bits of randomness for generating instances where r is some polynomial, we use a PRG $G : \{0,1\}^{t(\lambda)} \to \{0,1\}^{r(\lambda)}$, and sample an instance using $\mathcal{D}(G(s))$ for a random $s \in \{0,1\}^{t(\lambda)}$. In fact, assuming subexponentially-secure PRGs, we can instantiate G such that it expands a $t = \log^2(\lambda)$-bit seed into an r length string. Therefore, the set $\{\mathcal{D}(G(s))\}_{s \in \{0,1\}^t}$ is sparse, and now we can borrow ideas from our result for sparse languages.

Coming back to the NIZK construction, we define the Judge algorithm identically as before, except that it now requires an accepting proof for $x \oplus x^* \in L$ for an x for which there exists s such that $x = \mathcal{D}(G(s))$. Recall that x^* is a random string that is part of the CRS.

Showing defamation-freeness is now more challenging. While the high-level idea is still to contradict the adaptive soundness of the underlying NIZK, this requires some care. Specifically, let $\mathcal{A}$ with some PPT adversary that given an honestly generated CRS containing a uniform string x^* convinces the Judge with some non-negligible probability ϵ. In particular, $\mathcal{A}$ finds some random string s and an accepting proof for $x \oplus x^*$ where $x = \mathcal{D}(G(s))$. Then consider the following sequence of hybrids.

1. Hybrid Hyb_1: it is identical to the defamation-free experiment except we change the winning condition for $\mathcal{A}$. Specifically, Hyb_1 while generating the CRS that contains x^*, also samples a t-bit string s^* as a guess for $\mathcal{A}$'s string s. Then, we let $\mathcal{A}$ win Hyb_1 only if it finds an accepting proof for $x \oplus x^*$ for $x = \mathcal{D}(G(s))$ and it is the case that $s = s^*$. Then, it is then clear that $\mathcal{A}$ wins Hyb_1 with probability $\epsilon/2^t$.

2. Hybrid Hyb_2: it is identical to the Hyb_1, except we generate the string x^* embedded inside

the CRS differently. In particular, after sampling the guess s^*, Hyb_1 programs x^* such that the string $\mathcal{D}(G(s^*)) \oplus x^*$ is a NO instance sampled from some distribution $\mathcal{D}_{\mathsf{No}}$.

The only difference between Hyb_2 and Hyb_1 is the distribution of the string $x' = \mathcal{D}(G(s^*)) \oplus x^*$. In particular, in Hyb_1, the string x' is actually distributed according to the uniform distribution, whereas in Hyb_2 it is distributed according to $\mathcal{D}_{\mathsf{No}}$. Then if the two distributions are δ-indistinguishable then $\mathcal{A}$ wins Hyb_2 with probability at least $\epsilon/2^t - \delta$. Finally, note that if $\mathcal{A}$ wins Hyb_2, then it wins by finding an accepting proof for a false statement $x \oplus x^*$ where $x = \mathcal{D}(G(s^*))$. This is because $\mathcal{D}(G(s^*)) \oplus x^*$ was a NO instance, and $\mathcal{A}$ wins only if $s = s^*$.

Now, $\mathcal{A}$ can be used to build a cheating prover that breaks the adaptive soundness of the underlying soundness with probability $\epsilon/2^t - \delta$. Then, if $\delta = 2^{-\lambda^\epsilon}$ and $t = \log(\lambda)$, we arrive at a contradiction as long as the adaptive soundness of the underlying NIZK is such that no PPT adversary can break soundness with probability better than $2^{-\lambda^\epsilon}$. This concludes the discussion on defamation-freeness. Identical to the case of sparse languages, accountability can be shown for $\mathcal{D}$ which are indistinguishable from the uniform distribution over ℓ bit strings.

There are a few caveats associated with the above construction. First, the construction (i.e., Judge) depends on the distribution $\mathcal{D}$, which means that for different distributions, we need a different construction. Moreover, it requires somewhat strong assumptions (sub-exponential - indistinguishability), whereas our previous results were based on standard security. We bring this result as evidence that the general problem of (practical) accountable soundness for every NP language is intriguing and requires further investigation.

2.3 Related Work

To better understand the role of trust in CRS generation, a number of works have studied relaxed settings: Groth and Ostrovsky study the multi-string model [GO07] where multiple authorities individually publish common reference strings with only a majority of them are guaranteed to be honest. Garg et al.[GGJS11] study replacing a single CRS generating authority in UC with multiple, untrusted authorities. Bellare et al. [BFS16] introduce and study the feasibility of security properties retained by a NIZK under a maliciously chosen CRS, and Ananth et al. [AADG21] study accountability in the CRS generation.

In addition, numerous works have considered relaxed security notions for privacy to get non-interactive constructions in the plain model (one that does not require any common reference string). These include witness-indistinguishability [DN07], super polynomial simulation security [Pas03, BP04], and witness-hiding [KZ20].

Two works that come closest in spirit to ours are that of [BFS16] and [AADG21], which we give a detailed comparison next.

Comparison with [BFS16]. Bellare, Fuchsbauer, and Scafuro [BFS16] introduced the study of subversion-resistant security for NIZKs. Specifically, this asks what security properties are guaranteed if the CRS is maliciously sampled. They show a construction of a NIZK system in which the witness is protected (i.e., satisfies ZK) even if the CRS is maliciously sampled. However, the construction relies on knowledge-type assumptions, which are non-falsifiable (specifically, a knowledge-of-exponent assumption in a group equipped with a bilinear map). Constructing subversion-ZK NIZK based on standard cryptographic assumptions seems unlikely. This is because subversion-ZK NIZKs immediately imply two-round ZK protocols, which again are only known from knowledge-type assumptions. Our construction achieves a weaker notion of security than full ZK (in case of a

corrupted CRS) but is based on standard assumptions.

Comparison with [AADG21]. Recently, Ananth et al. [AADG21] propose a notion of accountability towards addressing trust assumptions in the CRS generation procedure. More specifically, they consider a CRS generation authority that extracts witnesses from NIZK proofs generated using the maliciously sampled CRS, and then sells these witnesses for monetary benefit on the black market.

Ananth et al. build a NIZK system that achieves both accountability and defamation-free properties from polynomial-time standard assumptions on bilinear maps. Our work expand their result in two different dimensions:

1. Our subversion advice ZK NIZK subsumes accountable NIZK (see below);
2. Ananth et al. studied accountable NIZK while addressing only privacy; and we study the orthogonal question of soundness.

Subversion advice ZK NIZKs. Our notion expands their results in three important aspects. First, subversion advice ZK NIZK immediately satisfies the notion of accountability as formulated by [AADG21]. Specifically, no PPT adversary, including the authority who generates the malicious CRS^* and perhaps knows some backdoors, can extract witnesses from proofs that were proven with respect to CRS^*. Thus, there is no need to hold the authority accountable, as no such authority exists. Second, the construction of [AADG21] works only for a large class of NP languages (but not all). This, of course, limits the applicability of that result. Our construction also holds for all NP languages. Third, the work of [AADG21] leaves open the question of how to handle authorities that, given a proof reveals only hard-to-compute partial information about the witness instead of the witness in its entirely. Our notion of subverted advice ZK NIZKs hides not only entire witnesses but also any partial information about the witness.

Additional related works. The notion of accountable soundness is inspired by broadcast encryption with traitor tracing [CFN94, BSW06, NWZ15, GKW18, GKWW21], the accountable authority of identity-based encryption [Goy07], watermarking or copy protection [GKM+19]. In all of those works, capturing flavor of security is challenging, and there can be many perils.

Organization. In Section 3 we provide preliminaries and definitions. In Section 4, we formally define advice ZK for two-round arguments and give our construction for NP. In Section 5, we formally define the notion of an subversion advice-ZK NIZK, and describe our subversion advice-ZK NIZK construction. Section 6 is devoted to defining accountable soundness and Section 7 describes our constructions for $\mathsf{NP} \cap \mathsf{coNP}$ and sparse languages, as well as our general feasibility result for NP. In Section 8 we build NIZKs with both subversion advice-ZK and accountable soundness. In Appendix B, we formally show connections of our subversion advice-ZK with other notions for subversion-security.

3 Preliminaries

Notation and Conventions. We let $\lambda \in \mathbb{N}$ denote the security parameter. We use PPT as a shorthand for probabilistic polynomial time. We denote by $x \leftarrow \mathcal{D}$ a sampling of an instance x according to the distribution $\mathcal{D}$.

- A function μ is negligible if for every positive polynomial $p(\cdot)$ there exists $\lambda_0 \in \mathbb{N}$ such that for all $\lambda > \lambda_0$ it holds that $\mu(\lambda) < 1/p(\lambda)$.

- A probability ensemble $X = \{X(a,\lambda)\}_{a\in\{0,1\}^*;\lambda\in\mathbb{N}}$ is an infinite sequence of random variables indexed by $a \in \{0,1\}^*$ and $\lambda \in \mathbb{N}$. In the context of zero knowledge, the value a will represent the parties' inputs and λ will represent the security parameter. All parties are assumed to run in time that is polynomial in the security parameter.

- Two probability ensembles $X = \{X(a,\lambda)\}_{a\in\{0,1\}^*;\lambda\in\mathbb{N}}$, $Y = \{Y(a,\lambda)\}_{a\in\{0,1\}^*;\lambda\in\mathbb{N}}$ are said to be computationally indistinguishable, denoted by $X \approx_c Y$, if for every non-uniform PPT distinguisher D there exists a negligible function μ such that for every $a \in \{0,1\}^*$ and every $\lambda \in \mathbb{N}$,
$$|\Pr[D(X(a,\lambda)) = 1] - \Pr[D(Y(a,\lambda)) = 1]| \leq \mu(\lambda) \ .$$

3.1 One-way Functions

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if:

1. There exists a deterministic polynomial time algorithm that on input s computes $f(s)$.
2. For every non-uniform PPT adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ such that:
$$\Pr\left[f(x') = y \ : \ x \leftarrow \{0,1\}^\lambda, \ y = f(x), \ x' \leftarrow \mathcal{A}(y)\right] \leq \mu(\lambda) \ .$$
Further, we say f is T-one-way for function $T : \mathbb{N} \to \mathbb{N}$, if the above holds for adversaries $\mathcal{A}$ that run in time $T(\lambda) \cdot p(\lambda)$ for some polynomial $p(\cdot)$.

3.2 Commitment Schemes

We require commitment scheme that is perfectly binding and computationally hiding. The non-interactive commitment scheme Com has the following syntax and properties:

- $\mathbf{c} \leftarrow \mathsf{Com}(m; r)$: The algorithm gets $m \in \{0,1\}^{\ell_{\mathsf{m}}(\lambda)}$ and randomness $r \in \{0,1\}^{\ell_{\mathsf{r}}(\lambda)}$ and outputs a commitment $\mathbf{c} \in \{0,1\}^{\ell_{\mathsf{c}}(\lambda)}$. The opening of the commitment is simply the randomness r.

We require the following properties from the commitment scheme:

- **Perfectly Binding:** For all $(m_0, m_1) \in \mathcal{M}$ such that $m_0 \neq m_1$ it holds that
$$\{\mathsf{Com}(m_0; r_0)\}_{r_0\in\{0,1\}^{\ell_{\mathsf{r}}(\lambda)}} \bigcap \{\mathsf{Com}(m_1; r_1)\}_{r_1\in\{0,1\}^{\ell_{\mathsf{r}}(\lambda)}} = \emptyset \ .$$
- **Computationally Hiding:** For every polynomially bounded function $\alpha(\cdot)$ and every polynomial-time non-uniform adversary $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that every auxiliary input $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$, the probability that $\mathcal{A}$ wins the following game is at most $1/2 + \mu(\lambda)$: For security parameter λ, $\mathcal{A}(1^\lambda, z)$ outputs a pair of values $m_0, m_1 \in \{0,1\}^{\alpha(\lambda)}$. The challenger on input m_b, for a randomly chosen bit $b \in \{0,1\}$, outputs a commitment to m_b. $\mathcal{A}$ then outputs a bit b' and wins iff $b' = b$.
- ***T*-Extraction:** There exists a deterministic algorithm that runs in time $T(\lambda) \cdot p(\lambda)$ for some polynomial $p(\cdot)$ such that on input any string $\mathbf{c} \in \{0,1\}^{\ell_c}$, outputs $\mathsf{val}(\mathbf{c})$ where
$$\mathsf{val}(\mathbf{c}) = m \iff \exists r \in \{0,1\}^{\ell_{\mathsf{r}}(\lambda)} \text{ s.t. } \mathbf{c} = \mathsf{Com}(m; r) \ .$$

3.3 Non-Interactive Zero Knowledge (NIZK)

Let L be an NP language and let R_L be its associated relation. For $(x, y) \in R_L$ we sometimes denote x the statement and w its associated witness.

Definition 3.1. *Let $L \in \mathsf{NP}$ and let R_L be the corresponding NP relation. A triple of algorithms $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ is called* non-interactive zero knowledge (NIZK) argument *for L if it satisfies:*

- **Perfect Completeness:** *For all security parameters $\lambda \in \mathbb{N}$ and for all $(x, w) \in R_L$,*

$$\Pr\left[\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1 : \mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda);\ \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}, x, w)\right] = 1 \ .$$

- **Computational Adaptive Soundness:** *For every PPT prover P^*, there exists a negligible function $\mu(\cdot)$ such that for all $\lambda \in \mathbb{N}$:*

$$\Pr\left[\mathsf{Soundness}_{\Pi, P^*}(\lambda) = 1\right] \leq \mu(\lambda) \ ,$$

  where the random variable $\mathsf{Soundness}_{\Pi, P^*}(\lambda)$ is defined as follows:

  1. $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$,
  2. $(x, \pi) \leftarrow P^*(\lambda, \mathsf{CRS})$,
  3. *The output of the experiment is* 1 *if* $x \notin L \wedge \mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$.

  When this probability is 0, we say that Π is perfectly sound.

- **Computational Zero-Knowledge:** *There exists a PPT simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ where $S_1(1^\lambda)$ outputs $(\mathsf{CRS}_\mathcal{S}, \tau)$ and $\mathcal{S}_2(\mathsf{CRS}_\mathcal{S}, \tau, x)$ outputs $\pi_\mathcal{S}$ such that for all non-uniform PPT adversaries $\mathcal{A}$:*

$$\left\{\mathcal{A}^{\mathcal{O}_1(\mathsf{CRS}, \cdot, \cdot)}(\mathsf{CRS}) : \mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)\right\}$$
$$\approx_c \left\{\mathcal{A}^{\mathcal{O}_2(\mathsf{CRS}_\mathcal{S}, \tau, \cdot, \cdot)}(\mathsf{CRS}_\mathcal{S}) : (\mathsf{CRS}_\mathcal{S}, \tau) \leftarrow \mathcal{S}_1(1^\lambda)\right\}$$

  where $\mathcal{O}_1, \mathcal{O}_2$ on input (x, w) first check that $(x, w) \in R_L$, else output $\perp$. Otherwise $\mathcal{O}_1$ outputs $\mathsf{Prove}(\mathsf{CRS}, x, w)$ and $\mathcal{O}_2$ outputs $\mathcal{S}_2(\mathsf{CRS}_\mathcal{S}, \tau, x)$.

3.4 Witness Indistinguishable (WI) Arguments

Definition 3.2 (Witness Indistinguishable Argument)**.** *Let L be an NP language, and let R_L be its associated relation. We say that a delayed-input two-message argument system $\mathsf{WI} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ is* witness distinguishable *for $L \in \mathsf{NP}$, if the following properties hold:*

- **Perfect Completeness:** *For every $(x, w) \in R_L$ and every $\mathsf{wi}_1 \in [\mathsf{V}_0(1^\lambda)]$*

$$\Pr\left[\mathsf{V}_1(x, \mathsf{wi}_1, \mathsf{P}(x, w, \mathsf{wi}_1)) = 1\right] = 1 \ .$$

- **(Two Rounds) Soundness:** *We say that a delayed-input two-message argument system* $\Pi = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ *achieves* (two-rounds) soundness *if for every* PPT *algorithm (corrupted prover)* P^* *there exists a negligible function* $\mu(\cdot)$ *such that:*

$$\Pr\left[\mathsf{2RndSND}_{\Pi,P^*}(\lambda) = 1\right] \leq \mu(\lambda)$$

where the random variable $\mathsf{2RndSND}_{\Pi,P^*}(\lambda)$ *is defined as follows:*

1. $m_1 \leftarrow \mathsf{V}_0(1^\lambda)$,
2. $(x^*, m_2) \leftarrow P^*(m_1)$,
3. *The outputs of the experiment is* 1 *if* $\mathsf{V}_1(x^*, m_1, m_2) = 1$ *and* $x^* \notin L$.

We say that Π *satisfies* T-soundness *for some* $T : \mathbb{N} \to \mathbb{N}$ *if the above also holds for* P^* *that run in time* $T(\lambda) \cdot p(\lambda)$ *for some polynomial* $p(\cdot)$.

- **Witness Indistinguishability:** *For every polynomially bounded function* s *and every polynomial-time non-uniform adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\cdot)$ *such that every auxiliary input* $z \in \{0,1\}^*$, *the probability that* $\mathcal{A}$ *wins the following game is at most* $1/2 + \mu(\lambda)$*:*

  For security parameter λ, $\mathcal{A}(1^\lambda, z)$ *outputs an instance* $x \in L \cap \{0,1\}^{s(\lambda)}$, *witnesses* w_0, w_1 *such that* (x, w_0) *and* (x, w_1) *belong to* R_L *and the first message* wi_1. *The challenger on input* w_b, *for a randomly chosen bit* $b \in \{0,1\}$, *sends* wi_2 *to* $\mathcal{A}$ *where* $\mathsf{wi}_2 \leftarrow \mathsf{P}(x, w_b, \mathsf{wi}_1)$. $\mathcal{A}$ *then outputs a bit* b' *and wins iff* $b' = b$.

4 Two Round Advice ZK Arguments for NP

In this section, we build a two round argument system for NP that satisfies advice zero-knowledge. Our construction follows the Fiat-Lapidot-Shamir (FLS) paradigm [FLS90] where: (a) the verifier's message sets up a secret trapdoor, and (b) the prover's message contains a WI proof showing either that the given statement is true or that it knows some trapdoor. Section 4.1 defines advice zero-knowledge, Section 4.2 gives an abstraction of the verifier's trapdoor-generation protocol, and Section 4.3 contains our two-round argument with formal security proofs in Section 4.4 (soundness) and Section 4.5 (advice-ZK).

4.1 Defining Advice ZK for Two Round Arguments

Definition 4.1. *Let* L *be an* NP *language, let* R_L *be its associated relation and let* S *be some super-polynomial function. We say that a two-message argument system* $\Pi = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ *satisfies* S-advice ZK *if for every non-uniform* PPT *algorithm* V^*, *there exists a* PPT *algorithm* $\mathcal{S}$ *and an* $S(\lambda)$*-time computable advice distribution* $\mathcal{D}$ *such that the distributions* $\mathsf{REAL}_{\mathsf{V},\Pi}$ *and* $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}$ *are computationally indistinguishable:*
$\underline{\mathsf{REAL}_{\mathsf{V}^*,\Pi}(\lambda)}$*:*

1. *On input* 1^λ, V^* *sends the first message* zk_1^*.
2. V^* *chooses statements adaptively until halting, and the experiment outputs its view upon halting:*
   (a) V^* *outputs* x. *If* $x \notin L$, *reply with* $\perp$, *else let* w *be a witness for* x.

(b) Compute the second message $\mathsf{zk}_2 \leftarrow \mathsf{Prove}(\mathsf{zk}_1^*, x, w)$ *and give* V^*.

$\underline{\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)}$*:*

1. *On input* 1^λ, V^* *sends the first message* zk_1^*.
2. *Sample an advice string* d *from the distribution* $\mathcal{D}(1^\lambda, \mathsf{zk}_1^*)$ *in* $S(\lambda)$ *time.*
3. V^* *chooses statements adaptively until halting, and the experiment outputs its view upon halting:*
   (a) V^* *outputs* x*. If* $x \notin L$*, reply with* $\perp$*, else let* w *be a witness for* x*.*
   (b) Run $\mathcal{S}(1^\lambda, \mathsf{zk}_1^*, x, d)$ *and give it to* V^**. Note that* w *is not used in this computation.*

We emphasize that sampling from the advice distribution $\mathcal{D}$ is not efficient as it requires S time, a-priori fixed super-polynomial function. However, upon given a sample from $\mathcal{D}$, the simulator must run in polynomial time. Philosophically, our notion ensures that whatever the adversary could have learnt from multiple interactions, it could have *efficiently* computed on its own after a *one-time* S time preprocessing of the language. Comparatively, the notion of superpolynomial simulation (SPS)-ZK allows the simulator to perform superpolynomial time computations for every statement. In this sense, our notion implies SPS-ZK with an S-time simulator with the advice distribution $\mathcal{D}$ and the advice ZK simulator $\mathcal{S}$ acting as a single SPS simulator. In some cases though it may be qualitatively better. E.g., consider languages where (a) statements have unique witnesses, and (b) witnesses can be brute-forced in some superpolynomial time S. Here, for SPS-ZK with an S time simulator, a perfectly acceptable solution is to send the witness in the clear as the proof; the S time simulator can brute-force for it during simulation. Whereas this is not possible for S-advice ZK as the S time computation (i.e., sampling from the advice distribution) happens even before the instance is known.

Remark 4.2. *Note that the adversary* V^* *can only choose statements* x*. The corresponding witness* w *given to the honest prover in the real experiment is computed via brute force. Our definition requires indistinguishability of the two experiments irrespective of how and which witness is computed.*

4.2 Trapdoor Generation

Towards constructing two round advice ZK argument, we consider a trapdoor generation scheme which is a tuple of four algorithms $(\mathsf{TDGen}, \mathsf{TDCheck}, \mathsf{TDValid}, \mathsf{TDSol})$ with the following syntax:

1. $\mathsf{TDGen}(1^\lambda)$ is a randomized PPT algorithm that on input the security parameter λ outputs an element $y \in \mathcal{Y}$.
2. $\mathsf{TDCheck}(x, y)$ is a deterministic PPT algorithm that on input elements $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, outputs a bit $b \in \{0, 1\}$.
3. $\mathsf{TDValid}(y)$ is a deterministic PPT algorithm that on input element $y \in \mathcal{Y}$, outputs a bit $b \in \{0, 1\}$.
4. $\mathsf{TDSol}(y)$ is a deterministic algorithm (possibly inefficient) that on input element $y \in \mathcal{Y}$, outputs an element $x' \in \mathcal{X}$.

We require the following properties:

1. For every $\lambda \in \mathbb{N}$ we have that $\mathsf{TDValid}\left(\mathsf{TDGen}(1^\lambda)\right) = 1$.

2. For every $\lambda \in \mathbb{N}$ and every $y \in \mathcal{Y}$, $\mathsf{TDValid}(y) = 1$ iff there exists an $x \in \mathcal{X}$ such that $\mathsf{TDCheck}(x, y) = 1$.
3. T-hardness: For every T-sized family of circuits $A = \{A_\lambda\}_\lambda$ there exists a negligible function μ such that for all $\lambda \in \mathbb{N}$

$$\Pr[\mathsf{TDCheck}(x', y) = 1 : y \leftarrow \mathsf{TDGen}(1^\lambda), x' \leftarrow A_\lambda(y)] \leq \mu(\lambda) \ .$$

4. S-solvability: TDSol is an $S(\lambda)$-time algorithm such that for every $\lambda \in \mathbb{N}$,

$$\Pr[\mathsf{TDCheck}(x', y) = 1 : y \leftarrow \mathsf{TDGen}(1^\lambda), x' = \mathsf{TDSol}(y)] = 1 \ .$$

In Appendix A, we provide instantiations from one-way functions with efficiently recognizable range and collision-resistant hash functions.

4.3 Construction of Two Round Advice ZK Argument

Tools. Let L be some NP language and R_L be its associated relation. Let $T, S : \mathbb{N} \to \mathbb{N}$ be some super-polynomial functions. The construction of two-round advice ZK argument for L is based on the following building blocks:

- A T-hard, S-solvable trapdoor generation protocol $\mathsf{TD} = (\mathsf{TDGen}, \mathsf{TDCheck}, \mathsf{TDValid})$.
- A perfectly binding and T-extractable non-interactive commitment Com (Section 3.2).
- A delayed-input, two-message, T-sound and publicly-verifiable, witness indistinguishable argument (Section 3.4) $\mathsf{WI} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ for the language:

$$\begin{aligned} L_{\mathsf{WI}} \quad = \quad & \{(x, \mathsf{s_{out}}, \mathbf{c}) \mid \exists (w, r, \mathsf{s_{in}}) \text{ s.t. } (x, w) \in R_L \\ & \vee (\mathsf{TDCheck}(\mathsf{s_{in}}, \mathsf{s_{out}}) = 1 \wedge \mathbf{c} = \mathsf{Com}(\mathsf{s_{in}}; r))\} \ . \end{aligned}$$

Construction. In our construction, the verifier runs TDGen to compute $\mathsf{s_{out}}$, and sends it along with the first message of the WI argument. Then, upon input a statement-witness pair (x, w), the prover computes a commitment $\mathbf{c}$ to 0 and proves, using the WI argument, that either $(x, w) \in L$ or $\mathbf{c}$ is a commitment to a valid pre-image of $\mathsf{s_{out}}$.

Construction 4.3: Two-round advice ZK argument

$\mathbf{V_0^{zk}}(\mathbf{1^\lambda})$:

1. Sample $\mathsf{s_{out}} \leftarrow \mathsf{TDGen}(1^\lambda)$
2. Sample the first message $\mathsf{wi}_1 \leftarrow \mathsf{WI}.\mathsf{V}_0(1^\lambda)$ of the WI protocol
3. Output $\mathsf{zk}_1 = (\mathsf{s_{out}}, \mathsf{wi}_1)$.

$\mathbf{P^{zk}}(\mathbf{1^\lambda}, \boldsymbol{x}, \boldsymbol{w}, \mathbf{zk_1})$:

1. Parse zk_1 as $(\mathsf{s_{out}}, \mathsf{wi}_1)$
2. If $\mathsf{TDValid}(\mathsf{s_{out}}) \neq 1$ then abort and output $\perp$
3. Compute a commitment $\mathbf{c} = \mathsf{Com}(0; r)$ using randomness $r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}$.
4. Compute $\mathsf{wi}_2 \leftarrow \mathsf{WI}.\mathsf{P}\left((x, \mathsf{s_{out}}, \mathbf{c}), (w, \perp, \perp), \mathsf{wi}_1\right)$.
5. Output $\mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2)$.

$\mathbf{V}_1^{\mathsf{zk}}(1^\lambda, x, (\mathbf{zk_1}, \mathbf{zk_2}))$:

1. Parse zk_1 as $(\mathsf{s}_{\mathsf{out}}, \mathsf{wi}_1)$, and $\mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2)$.
2. Accept iff $\mathsf{WI}.\mathsf{V}_1((x, \mathsf{s}_{\mathsf{out}}, \mathbf{c}), (\mathsf{wi}_1, \mathsf{wi}_2)) = 1$.

Before proceeding to the formal theorem, we highlight that:

1. The construction is two-message (verifier to prover; prover to verifier).
2. *Delayed-input*: the first message of the verifier is independent of the instance.
3. *Publicly verifiable*: The verifier does not store a secret state after its first message. The decision to accept/reject depends only on the transcript.

Theorem 4.4. *Let L be an NP language, and $T(\cdot), S(\cdot)$ be super-polynomial functions. Assume $(\mathsf{TDGen}, \mathsf{TDCheck}, \mathsf{TDValid})$ is T-hard and S-solvable, Com is a T-extractable non-interactive commitment scheme, and $\mathsf{WI} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ is a two-message delayed-input, publicly-verifiable, witness-indistinguishable T-adaptively sound argument for L_{WI}. Then, Construction 4.3 is a two-round delayed-input, publicly-verifiable argument for L with T-adaptive soundness and S-advice ZK.*

Before proving Theorem 4.4, we make some remarks. Recently, Kuykendall and Zhandry [KZ20] built a two round witness-hiding argument from perfectly sound non-interactive witness-indistinguishable proofs (NIWIs) and subexponentially-secure injective one-way functions with efficiently recognizable codomain. In comparison, instantiating trapdoor generation protocol with one-way functions with efficiently recognizable range, we get a two-round advice ZK construction from two-round WI adaptively-sound arguments and subexponentially-secure non-interactive commitments and one-way functions with efficiently recognizable codomain. This is an improvement over [KZ20] as (a) advice ZK implies the witness-hiding notion considered in [KZ20], (b) we require two-round WI *arguments* as opposed to NIWI *proofs*. Also, our framework allows for more instantiations including from only post-quantum assumptions: trapdoor generation protocol from T-secure collision-resistant hash functions based on subexponential SIS [Ajt96], and T-extractable non-interactive commitments [GHKW17] and two-round delayed-input publicly-verifiable WI arguments from subexponential LWE [LVW19, BFJ+20].

Proof of Theorem 4.4. First, note that completeness is straightforward. Secondly, the delayed-input and public-verifiability properties are inherited directly from the underlying WI protocol. We give formal proofs for the soundness and advice ZK properties in Section 4.4 and Section 4.5 respectively, which together will conclude the proof of Theorem 4.4.

4.4 Proof of Soundness

Lemma 4.5. *The construction 4.3 is T-sound.*

Proof. We will show that every non-uniform adversary P^* running in time $T(\lambda) \cdot \mathsf{poly}(\lambda)$ there exists a negligible function $\mu(\cdot)$ such that:

$$\Pr[\mathsf{2RndSND}_{\mathsf{ZK},P^*}(\lambda) = 1] \leq \mu(\lambda) \ .$$

To show this we consider the following two hybrids:.

- Hyb_0: This is the soundness security game played by P^*. In particular,

1. The challenger in the soundness game first honestly samples the first verifier message $\mathsf{zk}_1 = (\mathsf{s}_{\mathsf{out}}, \mathsf{wi}_1)$ as specified in 4.3.
2. Run $P^*(1^\lambda, \mathsf{zk}_1)$ to get (x, zk_2).
3. Parse $\mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2)$.
4. The output of the hybrid is 1 iff $x \notin L$ and $(\mathsf{wi}_1, \mathsf{wi}_2)$ is accepted by the WI verifier on the statement $(x, \mathsf{s}_{\mathsf{out}}, \mathbf{c})$.

- Hyb_1: This hybrid is identical to Hyb_0, except that we change the winning condition for P^*: The output of the hybrid is 1 iff $\mathsf{Hyb}_0 = 1$ *and* the commitment $\mathbf{c}$ is a commitment to s_{in} for which $\mathsf{TDCheck}(\mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 0$.

For a fixed corrupted T-time prover P^* denote by p_b the probability that P^* wins Hyb_b, for $b \in \{0, 1\}$. We have the following claims:

Claim 4.6. *Assuming that* TD *is a trapdoor generation protocol which satisfies T-hardness, and* Com *is a perfectly binding T-extractable commitment scheme. For every corrupted T-time prover P^* there exists a negligible $\mu(\cdot)$ such that $p_0 - p_1 \leq \mu(\lambda)$.*

Proof. Let $\mathsf{s}_{\mathsf{in}}, r$ be the opening of the commitment $\mathbf{c}$ sent by P^*. P^* is invoked on zk_1 and outputs (x, zk_2). We have that:

$$\begin{aligned}
p_0 &= \Pr\left[(x \notin L \ \wedge\ \mathsf{V}_1^{\mathsf{zk}}(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1) \ \wedge\ (\mathsf{TDCheck}(\mathsf{s}_{\mathsf{out}}, \mathsf{s}_{\mathsf{in}}) = 0 \ \vee\ \mathsf{TDCheck}(\mathsf{s}_{\mathsf{out}}, \mathsf{s}_{\mathsf{in}}) = 1)\right] \\
&= \Pr\left[(x \notin L \ \wedge\ \mathsf{V}_1^{\mathsf{zk}}(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1) \ \wedge\ \mathsf{TDCheck}(\mathsf{s}_{\mathsf{out}}, \mathsf{s}_{\mathsf{in}}) = 0\right] \\
&\quad + \Pr\left[(x \notin L \ \wedge\ \mathsf{V}_1^{\mathsf{zk}}(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1) \ \wedge\ \mathsf{TDCheck}(\mathsf{s}_{\mathsf{out}}, \mathsf{s}_{\mathsf{in}}) = 1\right] \\
&= p_1 + \Pr\left[(x \notin L \ \wedge\ \mathsf{V}_1^{\mathsf{zk}}(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1) \ \wedge\ \mathsf{TDCheck}(\mathsf{s}_{\mathsf{out}}, \mathsf{s}_{\mathsf{in}}) = 1\right] .
\end{aligned}$$

Assume towards a contradiction that $p_0 - p_1 \geq \epsilon(\lambda)$ for some non-negligible $\epsilon(\cdot)$. We show that P^* can be used to construct a T-time adversary $\mathcal{A}$ for the T-hardness game of the trapdoor generation protocol TD: Upon a given input $y \leftarrow \mathsf{TDGen}(1^\lambda)$, $\mathcal{A}$ computes $\mathsf{wi}_1 \leftarrow \mathsf{WI.V}_0(1^\lambda)$ and parses $\mathsf{zk}_1 = (y, \mathsf{wi}_1)$. It runs $P^*(\mathsf{zk}_1)$ and obtains its output (x, zk_2). It parses $\mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2)$ and extracts $(\mathsf{s}_{\mathsf{in}}, r)$ from $\mathbf{c}$ with brute-force in time $T(\lambda) \cdot p(\lambda)$ for some polynomial p. $\mathcal{A}$ outputs s_{in}.

$\mathcal{A}$ runs in time $T(\lambda) \cdot q(\lambda)$ for some polynomial $q(\cdot)$. From the binding property of Com, s_{in} is indeed the unique value of $\mathbf{c}$. Moreover, $\mathcal{A}$ perfectly simulates $\mathsf{2RndSND}_{\mathsf{ZK},P^*}(\lambda)$ for P^* and therefore with probability at least $\epsilon(\lambda)$, $\mathcal{A}$ breaks the T-hardness of TD. ■

Claim 4.7. *By the T-adaptive computational soundness of the WI argument, there exists a negligible function μ such that $p_1 \leq \mu(\lambda)$.*

Proof. Assume there exists a corrupted T-time prover P^* which wins in Hyb_1 with non-negligible probability. We show that P^* can be used to construct a corrupted T-time prover P_{WI} that breaks the soundness property of WI: Upon an honestly generated input wi_1, P_{WI} computes $\mathsf{s}_{\mathsf{out}} \leftarrow \mathsf{TDGen}(1^\lambda)$ and parses $\mathsf{zk}_1 = (\mathsf{s}_{\mathsf{out}}, \mathsf{wi}_1)$. It runs $P^*(\mathsf{zk}_1)$ and obtains its output $(x, \mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2))$. P_{WI} outputs $((x, \mathsf{s}_{\mathsf{out}}, \mathbf{c}), \mathsf{wi}_2)$.

P_{WI} perfectly simulates P^* in Hyb_1. Therefore, if P^* wins Hyb_1, we have with probability at least $\epsilon(\lambda)$ that $x \notin L$, $\mathsf{WI.V}_1\left((x, \mathsf{s}_{\mathsf{out}}, \mathbf{c}), \mathsf{zk}_1, \mathsf{zk}_2\right) = 1$ and $\mathsf{TDCheck}(\mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 0$ where $\mathsf{s}_{\mathsf{out}}$ is the unique de-commitment value of $\mathbf{c}$ (this is by the definition of Hyb_1 and the binding property of Com). This

implies that with non-negligible probability $\epsilon(\lambda)$, $(x, \mathsf{s_{out}}, \mathbf{c}) \notin L_{\mathsf{wi}}$ and $\mathsf{WI.V}_1\left((x, \mathsf{s_{out}}, \mathbf{c}), \mathsf{wi}_1, \mathsf{wi}_2\right) = 1$, which breaks the soundness property of WI. ■

Combining Claim 4.6 and Claim 4.7 we have that there exists a negligible function $\mu(\cdot)$ such that $p_0 \leq \mu$. This concludes the proof of Lemma 4.5. □

4.5 Proof of Advice ZK

Lemma 4.8. *The construction 4.3 is S-advice ZK as per Definition 4.1.*

Proof. We will show that for every non-uniform PPT adversary V^* there exists an S-time computable advice distribution $\mathcal{D}$ and a PPT simulator $\mathcal{S}$ such that for every $(x, w) \in R_L$ the distributions $\mathsf{REAL}_{\mathsf{V}^*,\Pi}(\lambda)$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)$ defined in Definition 4.1 are computationally indistinguishable.

We first define the advice distribution $\mathcal{D}$ and the simulator $\mathcal{S}$:

Recall that in the $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)$, V^* first outputs zk_1, and then the advice distribution outputs some string d that would help the simulator. **Advice Distribution $\mathcal{D}(1^\lambda, \mathsf{zk}_1)$:**

1. Parse $\mathsf{zk}_1 = (\mathsf{s_{out}}, \mathsf{wi}_1)$.
2. If $\mathsf{TDValid}(\mathsf{s_{out}}) \neq 1$: set $\mathsf{s_{in}} = \bot$.
3. Else: set $\mathsf{s_{in}} = \mathsf{TDSol}(\mathsf{s_{out}})$.
4. Output $\mathsf{s_{in}}$.

Simulator $\mathcal{S}(1^\lambda, \mathsf{zk}_1, x, d)$:

1. Parse $d = \mathsf{s_{in}}$.
2. If $\mathsf{s_{in}} = \bot$, then send the prover message $\bot$ to V^* and output its view.
3. Otherwise, parse $\mathsf{zk}_1 = (\mathsf{s_{out}}, \mathsf{wi}_1)$.
4. Compute a non-interactive commitment $\mathbf{c} = \mathsf{Com}(\mathsf{s_{in}}; r)$ to $\mathsf{s_{in}}$ using randomness $r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}$.
5. Compute wi_2 using $(\mathsf{s_{in}}, r)$ as the witness. That is, $\mathsf{wi}_2 \leftarrow \mathsf{WI.Prove}((x, \mathsf{s_{out}}, \mathbf{c}), (\bot, \mathsf{s_{in}}, r), \mathsf{wi}_1)$.
6. Send $(x, \mathsf{zk}_2 = (\mathbf{c}, \mathsf{wi}_2))$ to V^* and output its view.

First, note that sampling from $\mathcal{D}$ can be computed by an S-time algorithm as TDSol is an S-time algorithm. Next, to show that the view of the adversary is computationally-indistinguishable in both executions $\mathsf{REAL}_{\mathsf{V}^*,\Pi}$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}$, consider the adversary V^*, and let zk_1 be its first message. Run $d \leftarrow \mathcal{D}(1^\lambda, \mathsf{zk}_1)$. We show that its view in each iteration is computationally-indistinguishable. Specifically, let (x, w) be the instance that the adversary outputs in the current iteration such that $R_L(x, w) = 1$. Consider the following sequence of distributions G_0, G_1 and G_2:

- **Distribution $G_0(\lambda, \mathsf{zk}_1, x, w, d)$:** This distribution is identical to the view of the adversary in the current iteration in the real. In particular,
  1. Parse $\mathsf{zk}_1 = (\mathsf{s_{out}}, \mathsf{wi}_1)$.
  2. Compute the prover message $\mathsf{P}^{\mathsf{zk}}(x, w, \mathsf{zk}_1)$ honestly by first computing a commitment $\mathbf{c}$ to 0 and then computing wi_2 for the statement $(x, \mathsf{s_{out}}, \mathbf{c})$ using the witness w.
  3. Send $(x, \mathbf{c}, \mathsf{wi}_2)$ to V^* .
- **Distribution $G_1(\lambda, \mathsf{zk}_1, x, w, d)$:** Parse $d = \mathsf{s_{in}}$. This game is identical to distribution G_0, except that the commitment $\mathbf{c}$ inside $\mathsf{P}^{\mathsf{zk}}(x, w, \mathsf{zk}_1)$ is computed for $\mathsf{s_{in}}$ instead of being a commitment to 0 as in G_0. That is, $\mathbf{c} = \mathsf{Com}(\mathsf{s_{in}}; r)$ for a random $r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}$.

- **Distributions** $G_2(\lambda, \mathsf{zk}_1, x, w, d)$**:** We further modify $\mathsf{P}^{\mathsf{zk}}(x, w, \mathsf{zk}_1)$. Now, wi_2 is generated using the witness $(\mathsf{s}_{\mathsf{in}}, r)$ where r is the random coins used to compute the commitment $\mathbf{c}$. Note that the proof that the verifier receives is the output of $\mathcal{S}(1^\lambda, \mathsf{zk}_1, x, d)$.

We show that the distributions G_0 and G_2 are computationally indistinguishable. First, we show that G_0 and G_1 are computationally indistinguishable due to the non-uniform hiding of the commitment scheme and G_1 and G_2 are indistinguishable due to the non-uniform witness indistinguishability of the WI scheme. To conclude the proof, we remark that once the view of the adversary in some particular distribution is computationally-indistinguishable then its output (i.e., the instance (x, w)) is also computationally-indistinguishable, as follows from a simple hybrid argument.

Claim 4.9. *By the non-uniform hiding of the commitment scheme, the distributions G_1 and G_2 are computationally indistinguishable.*

Proof. Note that the only difference between G_2 and G_1 is how the commitment $\mathbf{c}$ is generated. Specifically, notice that in G_1, $\mathbf{c}$ is a commitment to 0 whereas in G_2, $\mathbf{c}$ is a commitment s_{in}. Let us assume for contradiction that there exists some PPT distinguisher D, a polynomial $p(\cdot)$ such that for infinitely many $\lambda \in \mathbb{N}$, D distinguishes the two distributions with advantage $1/p(n)$.

By a standard averaging argument, there exists a $1/2p(n)$ fraction of the outputs d of the advice distribution $\mathcal{D}$, such that, conditioned on such a d occurring in both G_1 and G_2, D's advantage in distinguishing the two distributions is at least $1/2p(n)$. Fix one such $d = (r_{\mathcal{A}}, \mathsf{s}_{\mathsf{in}})$. Then, using d we build a non-uniform adversary B that breaks the hiding of the commitment scheme with advantage $1/2p(n)$.

Specifically, B is given d, x, w as non-uniform advice. Recall that $d = (r_{\mathcal{A}}, \mathsf{s}_{\mathsf{in}})$ where $r_{\mathcal{A}}$ is random coins of $\mathcal{A}$ and $\mathsf{TDCheck}(\mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 1$ where $\mathsf{s}_{\mathsf{out}}$ is defined by $r_{\mathcal{A}}$. Let $(\mathsf{s}_{\mathsf{out}}, \mathsf{wi}_1)$ be the output of $\mathcal{A}$ run with coins $r_{\mathcal{A}}$. B then gives the commitment challenger the message $m_0 = 0$ and $m_1 = \mathsf{s}_{\mathsf{in}}$ and receives a commitment $\mathbf{c}$ to m_b for a randomly chosen b. It then computes wi_2 for the statement $(x, \mathbf{c})$ by using the witness w. It runs $\mathcal{A}$ on inputs $(x, \mathbf{c}, \mathsf{wi}_2)$ to generate the view of $\mathcal{A}$. As its guess for the challenge bit b, it outputs whatever D outputs on the view of $\mathcal{A}$.

First note that B runs in polynomial time. Second, note that when the commitment challenge bit b is 0, then B perfectly emulates game G_1 for $\mathcal{A}$ conditioned on d being the output of the advice distribution. When $b = 1$, B perfectly emulates game G_2 for $\mathcal{A}$ conditioned on d being the output of the advice distribution. Since, conditioned on d being the output of $\mathcal{D}$, the distinguisher D distinguishes the two distributions with advantage $1/2p(n)$, we have that B breaks the non-uniform hiding of the commitment scheme with advantage $1/2p(n)$.

This contradicts the non-uniform hiding of the commitment scheme. ■

Claim 4.10. *By the non-uniform witness-indistinguishability of the WI argument, the distributions G_2 and G_3 are computationally indistinguishable.*

Proof. Note that the only difference between G_2 and G_3 is the witness used in computing the WI proof. Specifically, notice that in G_2, the WI proof is computed using the witness w for the statement x whereas in G_3 the WI proof is computed using the witness $(\mathsf{s}_{\mathsf{in}}, r)$. Let us assume for contradiction that there exists some PPT distinguisher D, a polynomial p such that for infinitely many $\lambda \in \mathbb{N}$, D distinguishes the two distributions with advantage $1/p(n)$.

By a standard averaging argument, there exists a $1/2p(n)$ fraction of the outputs d of the advice distribution $\mathcal{D}$, such that, conditioned on such a d occuring in both G_1 and G_2, D's advantage in

distinguishing the two distributions is at least $1/2p(n)$. Fix one such $d = (r_{\mathcal{A}}, \mathsf{s_{in}})$. Then, using d we build a non-uniform adversary B that breaks the hiding of the commitment scheme with advantage $1/2p(n)$.

Specifically, B is given d, x, w as non-uniform advice. Recall that $d = (r_{\mathcal{A}}, \mathsf{s_{in}})$ where $r_{\mathcal{A}}$ is random coins of $\mathcal{A}$ and $\mathsf{TDCheck}(\mathsf{s_{in}}, \mathsf{s_{out}}) = 1$ where $\mathsf{s_{out}}$ is defined by $r_{\mathcal{A}}$. Let $(\mathsf{s_{out}}, \mathsf{wi}_1)$ be the output of $\mathcal{A}$ run with coins $r_{\mathcal{A}}$. B then computes a non-interactive commitment $\mathbf{c}$ to $\mathsf{s_{in}}$ using randomness r. B then forwards to the witness-indistinguishability challenger the statement $(x, \mathsf{s_{out}}, \mathbf{c})$ as well as the two witnesses $w_0 = w$ and $w_1 = (\mathsf{s_{in}}, r)$. B then receives wi_2 computed using the witness w_b for a randomly chosen bit b. It runs $\mathcal{A}$ on inputs $(x, \mathbf{c}, \mathsf{wi}_2)$ to generate the view of $\mathcal{A}$. As its guess for the challenge bit b, it outputs whatever D outputs on the view of $\mathcal{A}$.

First note that B runs in polynomial time. Second, note that when the commitment challenge bit b is 0, then B perfectly emulates game G_2 for $\mathcal{A}$ conditioned on d being the output of the advice distribution. When $b = 1$, B perfectly emulates game G_3 for $\mathcal{A}$ conditioned on d being the output of the advice distribution. Since, conditioned on d being the output of $\mathcal{D}$, the distinguisher D distinguishes the two distributions with advantage $1/2p(n)$, we have that B breaks the non-uniform witness-indistinguishability of the WI argument with advantage $1/2p(n)$. ■

Combining Claim 4.9 and Claim 4.10, concludes the proof of Lemma 4.8. □

5 Subversion Advice-ZK NIZKs

In this section, we first give the formal definition of an subversion advice-ZK NIZK in Section 5.1. Then, in Section 5.2 we proceed to give our construction which converts any NIZK for NP into a subversion advice-ZK NIZK for NP while relying on two-round advice zero-knowledge arguments from Section 4.3.

5.1 Defining Subversion Advice-ZK NIZKs

Definition 5.1. *Let L be an NP language, and let R_L be its associated relation. Let S be some super-polynomial function. A NIZK argument $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ for L is an S-subversion advice-ZK NIZK if for every non-uniform PPT adversary $\mathcal{A}$, there exists a PPT $\mathcal{S}$ and an $S(\cdot)$-time computable advice distribution $\mathcal{D}$ such that the output of the following two distributions are computationally-indistinguishable:*

$\mathsf{REAL}_{\mathcal{A},\Pi}(\lambda)$*:*

1. *Obtain $\mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)$.*
2. *$\mathcal{A}$ chooses statements adaptively until halting, and the experiment outputs its view upon halting:*
   (a) *$\mathcal{A}$ queries on x. If $x \notin L$, $\mathcal{A}$ receives $\perp$. Else, it receives $\pi = \mathsf{Prove}(\mathsf{CRS}^*, x, w)$ where w is a witness for x.*

$\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)$*:*

1. *Obtain $\mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)$.*
2. *Sample an advice string d from the distribution $\mathcal{D}(1^\lambda, \mathsf{CRS}^*)$ in $S(1^\lambda)$ time.*
3. *$\mathcal{A}$ chooses statements adaptively until halting, and the experiment outputs its view upon halting:*

(a) $\mathcal{A}$ queries on x. If $x \notin L$, $\mathcal{A}$ receives $\perp$. Else, it receives $\pi = \mathcal{S}(1^\lambda, \mathsf{CRS}^*, x, d)$. Note that $\mathcal{S}$ does not use w.

5.2 Construction of Subversion Advice-ZK NIZKs

Tools. Let L be an NP language and R_L be its associated relation. For $T, S : \mathbb{N} \to \mathbb{N}$ be some super-polynomial functions, we use the following building blocks:

- A delayed-input two-message publicly-verifiable S-advice ZK argument $\mathsf{ZK} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ (Definition 4.1) with perfect completeness and T-adaptive soundness for L.
- A non-interactive perfectly binding and T-extractable commitment Com (Section 3.2) .
- An adaptively-sound NIZK argument $\mathsf{NIZK} = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ (Definition 3.1) for the associated relation of the language

$$L_{\mathsf{Inner}} = \{(x, \mathsf{zk}_1, \mathbf{c}) \mid \exists\, (\mathsf{zk}_2, r) \text{ s.t. } \mathsf{ZK.V}_1(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1 \wedge \mathbf{c} = \mathsf{Com}(\mathsf{zk}_2; r)\} \ .$$

We describe below the construction and the theorem (proven in Section 5.3).

Construction 5.2: Subversion Advice-ZK NIZK Π

$\mathsf{GenCRS}(1^\lambda)$:

1. Generate $\mathsf{CRS}_{\mathsf{Inner}} \leftarrow \mathsf{NIZK.GenCRS}(1^\lambda)$.
2. Compute $\mathsf{zk}_1 \leftarrow \mathsf{ZK.V}_0(1^\lambda)$.
3. $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$.

$\mathsf{Prove}(\mathsf{CRS}, x, w)$, where $x \in \{0,1\}^\lambda$:

1. Parse CRS as $(\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$.
2. $\mathsf{zk}_2 \leftarrow \mathsf{ZK.P}(x, \mathsf{zk}_1, w)$.
3. $\mathbf{c} = \mathsf{Com}(\mathsf{zk}_2; r)$ for a random $r \in \{0,1\}^{\mathsf{poly}(\lambda)}$.
4. Compute $\pi_{\mathsf{Inner}} = \mathsf{NIZK.Prove}\,(\mathsf{CRS}_{\mathsf{Inner}}, (x, \mathsf{zk}_1, \mathbf{c}), (\mathsf{zk}_2, r))$.
5. Output $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$.

$\mathsf{Verify}\,(\mathsf{CRS}, x, \pi)$:

1. Parse CRS as $(\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ and π as $(\pi_{\mathsf{Inner}}, \mathbf{c})$.
2. Output the decision of $\mathsf{NIZK.Verify}\,(\mathsf{CRS}_{\mathsf{Inner}}, (x, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}})$.

Theorem 5.3. *Let L be any NP language, let $T, S : \mathbb{N} \to \mathbb{N}$ be some superpolynomial functions. Then, Π is an adaptively-sound S-subversion advice-ZK NIZK argument for L assuming*

1. *ZK is a delayed-input, publicly-verifiable two-message S-advice ZK argument with perfect completeness and T-adaptive soundness;*
2. *NIZK is an adaptively-sound NIZK argument system for the language L_{Inner};*
3. *Com be a non-interactive, perfectly binding, and T-extractable commitment.*

5.3 Proof of Theorem 5.3

We show completeness, soundness and zero-knowledge, where the CRS is honestly generated. This would show that Π is a NIZK. Moreover, we will show the subversion advice-ZK property, for the case of a maliciously generated CRS.

Perfect Completeness. Let $(x, w) \in R_L$ and let $\pi = \mathsf{Prove}(\mathsf{CRS}, x, w)$. That is, $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$, where $\pi_{\mathsf{Inner}} = \mathsf{NIZK.Prove}\left(\mathsf{CRS}_{\mathsf{Inner}}, (x, \mathsf{zk}_1, \mathbf{c}), (\mathsf{zk}_2, r)\right)$ and $\mathsf{zk}_2 \leftarrow \mathsf{ZK.P}(x, \mathsf{zk}_1, w)$ and r is the randomness such that $\mathbf{c} = \mathsf{Com}(\mathsf{zk}_2; r)$. From the completeness of the two-round advice ZK argument it holds that $\mathsf{ZK.V}_1(x, \mathsf{zk}_1, \mathsf{zk}_2) = 1$. Thus, it holds that (zk_2, r) is a valid witness for $(x, \mathsf{zk}_1, \mathbf{c})$ in the inner language L_{Inner}. Then, from perfect completeness of the inner NIZK argument, we have that

$$\Pr\left[\mathsf{NIZK.Verify}(\mathsf{CRS}_{\mathsf{Inner}}, (x, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1\right] = 1 \ .$$

Adaptive Soundness. We show that for every PPT corrupted prover P^* there exists a negligible function $\mu(\cdot)$ such that for every $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathsf{Soundness}_{\Pi, P^*}(\lambda) = 1\right] \leq \mu(\lambda) \ .$$

Towards that end, we consider the following hybrids:

- Hyb_0: This is the real soundness game for Π played by P^*. Recall that in this hybrid, an honestly generated CRS is chosen according to $\mathsf{GenCRS}(1^\lambda)$. Then the corrupted prover P^* is invoked on CRS, and outputs a statement-proof pair (x^*, π). The output of this hybrid is 1 if $x^* \notin L$ and $\mathsf{Verify}(\mathsf{CRS}, x^*, \pi) = 1$. Recall from the construction that CRS is to be parsed as the tuple $(\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ and π is to be parsed as the tuple $= (\pi_{\mathsf{Inner}}, \mathbf{c})$. Then, the winning condition can be stated more precisely as follows:

$$\mathsf{Hyb}_0 \text{ outputs 1 if } x^* \notin L \ \wedge \ \mathsf{NIZK.Verify}(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1 \ .$$

- Hyb_1: In this hybrid, we require a stronger winning condition. In particular, the winning condition is identical to Hyb_0 except that we additionally require that $(x^*, \mathsf{zk}_1, \mathbf{c})$ be in the language L_{Inner}. In particular, this means that the value $\tilde{\mathsf{zk}}_2 = \mathsf{val}(\mathbf{c})$ committed inside $\mathbf{c}$ is an accepting second message for the statement x^* w.r.t. the two-round argument verifier $\mathsf{ZK.V}_1$. The winning condition can be more precisely stated as follows: Hyb_1 outputs 1 if (1) $x^* \notin L$; (2) $\mathsf{NIZK.Verify}(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1$; (3) $\mathsf{NIZK.Verify}(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1 \ \wedge \ \mathsf{ZK.V}_1(x^*, \mathsf{zk}_1, \mathsf{val}(c)) = 1$.

Let P^* be a non-uniform PPT corrupted prover P^*, and denote by p_i the success probability of P^* in Hyb_i for $i \in \{0, 1\}$.

Claim 5.4. *By the adaptive soundness of* NIZK*, for every non-uniform* PPT *corrupted prover* P^* *there exists a negligible function* $\mu_0(\cdot)$ *such that* $p_0 - p_1 \leq \mu_0(\lambda)$.

Proof. Assume there exists a corrupted prover P^* and a non-negligible function $\epsilon(\cdot)$ such that $p_0 - p_1 \geq \epsilon(\lambda)$. We show that P^* can be used to construct a cheating prover P_{NIZK} that breaks the adaptive-soundness of the NIZK for the language L_{Inner}: On input an honestly generated $\mathsf{CRS}_{\mathsf{Inner}}$, P_{NIZK} samples $\mathsf{zk}_1 \leftarrow \mathsf{ZK.}V_0(1^\lambda)$ and sets $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$. It then invokes P^* on input CRS to receive $(x^*, \pi = (\pi_{\mathsf{Inner}}, \mathbf{c}))$. It then outputs the statement $(x^*, \mathsf{zk}_1, \mathbf{c})$ along with π_{Inner} as the proof.

To analyse the success probability of P_{NIZK}, first recall that by our assumption that $p_0 - p_1 \geq \epsilon(\lambda)$, this implies that the output of P_{NIZK} is such that

$$\Pr\left[x^* \notin L \wedge \mathsf{NIZK.Verify}(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1 \right. \\ \left. \wedge \mathsf{ZK.V}_1(x^*, \mathsf{zk}_1, \mathsf{val}(c)) \neq 1\right] \geq \epsilon(\lambda) \ .$$

If $\mathsf{ZK.V}_1(x^*, \mathsf{zk}_1, \mathsf{val}(c)) \neq 1$ then this implies that $(x^*, \mathsf{zk}_1, \mathbf{c}) \notin L_{\mathsf{Inner}}$. Therefore, we have that

$$\Pr\left[(x^*, \mathsf{zk}_1, \mathbf{c}) \notin L_{\mathsf{Inner}} \wedge \mathsf{NIZK.Verify}^*(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1\right] \geq \epsilon(\lambda) \ .$$

This contradicts the adaptive-soundness of the NIZK. ■

Claim 5.5. *By the T-adaptive soundness property of ZK and T-extractability of Com, for every non-uniform* PPT *corrupted prover P^* there exists a negligible function $\mu_1(\cdot)$ such that $p_1 \leq \mu_1(\lambda)$.*

Proof. Assume there exists a corrupted prover P^* and a non-negligible function $\epsilon(\cdot)$ such that $p_1 \geq \epsilon(\lambda)$. We show that P^* can be used to construct a T-time cheating prover P_{ZK} that breaks the T-adaptive-soundness of the two round ZK argument for the language L: On input the first message zk_1 of the two-round argument ZK, P_{ZK} samples $\mathsf{CRS}_{\mathsf{Inner}}$ via $\mathsf{NIZK.GenCRS}(1^\lambda)$ and sets $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$. It then invokes P^* on CRS to receive $(x^*, \pi = (\pi_{\mathsf{Inner}}, \mathbf{c}))$ as the output. It then runs the T-time extractor for the commitment scheme Com to extract $\tilde{\mathsf{zk}}_2$ from the commitment $\mathbf{c}$, that is, $\tilde{\mathsf{zk}}_2 = \mathsf{val}(\mathbf{c})$. It then outputs the statement x^* along with the second message $\tilde{\mathsf{zk}}_2$.

To analyse the success probability of P_{ZK}, first recall that by our assumption that $p_1 \geq \epsilon(\lambda)$ we have that,

$$\Pr\left[x^* \notin L \wedge \mathsf{NIZK.Verify}^*(\mathsf{CRS}_{\mathsf{Inner}}, (x^*, \mathsf{zk}_1, \mathbf{c}), \pi_{\mathsf{Inner}}) = 1 \right. \\ \left. \wedge \mathsf{ZK.V}_1(x^*, \mathsf{zk}_1, \mathsf{val}(c)) = 1\right] \geq \epsilon(\lambda) \ .$$

Then, by the perfect T-extraction of Com we have that $\tilde{\mathsf{zk}}_2$ equals $\mathsf{val}(c)$. Therefore, the output $(x^*, \tilde{\mathsf{zk}}_2)$ of P_{ZK} is such that

$$\Pr\left[x^* \notin L \wedge \mathsf{ZK.V}_1(x^*, \mathsf{zk}_1, \mathsf{val}(c)) = 1\right] \geq \epsilon(\lambda) \ .$$

This contradicts the T-adaptive soundness of the two round argument ZK. ■

Combining the above two claims, we conclude that for every non-uniform PPT corrupted prover P^* there exists a negligible function $\mu := \mu_0 + \mu_1$ such that

$$\Pr\left[\mathsf{Soundness}_{\Pi, P^*}(\lambda) = 1\right] \leq \mu(\lambda) \ .$$

Zero Knowledge. We first give the zero-knowledge simulator $\mathcal{S}$ for Π. $\mathcal{S}$ samples a simulated CRS $(\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ where $\mathsf{CRS}_{\mathsf{Inner}}$ is generated using the zero-knowledge simulator $\mathcal{S}_{\mathsf{Inner}}$ of NIZK and zk_1 is honestly generated using $\mathsf{ZK.V}_0$. On each query x, $\mathcal{S}$ computes a commitment $\mathbf{c}$ to all zero strings, and runs $\mathcal{S}_{\mathsf{Inner}}$ to generate a simulated proof π_{Inner} for the statement $(x, \mathsf{zk}_1, \mathbf{c})$, and outputs $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$.

We show zero-knowledge by considering the following three hybrids.

- Hyb_0: This is the real world for the ZK property of Π. The adversary receives an honestly generated CRS, and on each query (x, w), the experiment checks that $(x, w) \in R_L$. If so, it replies with $\mathsf{Prove}(\mathsf{CRS}, x, w)$. Otherwise, it replies with $\perp$. The output of the experiment is the view of the adversary.

- Hyb_1: This hybrid is identical to Hyb_0 except we use the ZK simulator $\mathcal{S}_{\mathsf{Inner}}$ to compute simulated proofs. More specifically, the simulator $\mathcal{S}$ generates $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$, where $\mathsf{CRS}_{\mathsf{Inner}}$ is generated by $\mathcal{S}_{\mathsf{Inner}}$, and zk_1 is honestly generated according to $\mathsf{ZK.V}_0(1^\lambda)$. On each query (x, w), the experiment checks that $(x, w) \in R_L$. If so, it first computes an accepting zk_2 by running $\mathsf{ZK.Prove}(x, \mathsf{zk}_1, w)$ and then computes a commitment $\mathbf{c}$ to zk_2. It then runs $\mathcal{S}_{\mathsf{Inner}}$ to generate a simulated proof π_{Inner} for the statement $(x, \mathsf{zk}_1, \mathbf{c})$, and returns $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$.
- Hyb_2: This hybrid is identical to Hyb_1 except that $\mathcal{S}$, for every query, generates the commitment $\mathbf{c}$ as a commitment to the value 0. More specifically, the simulator generates $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$, where $\mathsf{CRS}_{\mathsf{Inner}}$ is generated by $\mathcal{S}_{\mathsf{Inner}}$, and zk_1 is honestly generated according to $\mathsf{ZK.V}_0(1^\lambda)$. On each query (x, w), the experiment checks that $(x, w) \in R_L$. If so, it first computes an accepting zk_2 by running $\mathsf{ZK.Prove}(x, \mathsf{zk}_1, w)$ and then computes a commitment $\mathbf{c}$ to zk_2. It then runs $\mathcal{S}_{\mathsf{Inner}}$ to generate a simulated proof π_{Inner} for the statement $(x, \mathsf{zk}_1, \mathbf{c})$, and returns $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$.

Claim 5.6. *By the zero-knowledge property of* NIZK *and the perfect completeness of the two round argument* ZK*, the view of the adversary $\mathcal{A}$ in Hyb_0 and Hyb_1 are indistinguishable.*

Proof. Assume there exists such adversary $\mathcal{A}$ for which there exists a non-uniform PPT distinguisher D that can distinguish with non-negligible probability between $\mathcal{A}$'s output in Hyb_0 and Hyb_1. We show that $\mathcal{A}$ can be used to construct a non-uniform PPT adversary $\mathcal{A}_{\mathsf{Inner}}$ that breaks the zero-knowledge of NIZK: $\mathcal{A}_{\mathsf{Inner}}$ receives $\mathsf{CRS}_{\mathsf{Inner}}$ (generated by either $\mathsf{NIZK.GenCRS}(1^\lambda)$ or $\mathcal{S}_{\mathsf{Inner}}(1^\lambda)$). It computes $\mathsf{zk}_1 \leftarrow \mathsf{ZK.V}_0(1^\lambda)$ and sends $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ to $\mathcal{A}$. On each query (x, w) that $\mathcal{A}$ makes to its oracle, $\mathcal{A}_{\mathsf{Inner}}$ computes zk_2 as in Prove and then computes a commitment $\mathbf{c}$ to zk_2 using randomness r. It then queries its oracle on $((x, \mathsf{zk}_1, \mathbf{c}), (\mathsf{zk}_2, r))$. It receives the oracle's output π_{Inner} and sends $(\pi_{\mathsf{Inner}}, \mathbf{c})$ to $\mathcal{A}$. It outputs $\mathcal{A}$'s output.

First, by the perfect completeness of the two-round argument, we have that for any query $(x, w) \in L$ by $\mathcal{A}$, the query $((x, \mathsf{zk}_1, \mathbf{c}), (\mathsf{zk}_2, r))$ generated by $\mathcal{A}_{\mathsf{Inner}}$ is such that $(x, \mathsf{zk}_1, \mathbf{c}) \in L_{\mathsf{Inner}}$ where (zk_2, r) is the witness. Now, D can be used as a distinguisher for NIZK: If $\mathcal{A}_{\mathsf{Inner}}$ is interacting with the real experiment, then it perfectly simulates $\mathcal{A}$ in Hyb_0. If $\mathcal{A}_{\mathsf{Inner}}$ is interacting with the $\mathcal{S}_{\mathsf{Inner}}$, then it perfectly simulates $\mathcal{A}$ in Hyb_1. Since D can distinguish between $\mathcal{A}$'s output in both executions, then it distinguishes between $\mathcal{A}_{\mathsf{Inner}}$'s outputs between the real experiment and the simulated experiment with non-negligible probability as well, violating the zero-knowledge property of NIZK. ■

Claim 5.7. *By the hiding of the commitment scheme* Com*, the view of the adversary $\mathcal{A}$ in Hyb_1 and Hyb_2 are indistinguishable.*

Proof. Assume there exists such adversary $\mathcal{A}$ for which there exists a non-uniform PPT distinguisher D that can distinguish with non-negligible probability between $\mathcal{A}$'s output in Hyb_1 and Hyb_2. We show that $\mathcal{A}$ can be used to construct a non-uniform PPT adversary B that breaks the hiding of Com:

B first computes $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ as in Hyb_1. That is, $\mathsf{CRS}_{\mathsf{Inner}}$ is computed by running $\mathcal{S}_{\mathsf{Inner}}$ and zk_1 is computed honestly. On each query (x, w) that $\mathcal{A}$ makes to its oracle, B computes zk_2 as in Prove, and then forwards $(\mathsf{zk}_2, 0^\lambda)$ as the two challenge messages for the hiding game. Upon receiving the challenge commitment $\mathbf{c}^*$ (which is either a commitment to zk_2 or to 0^λ), B runs $\mathcal{S}_{\mathsf{Inner}}$ on the statement $(x, \mathsf{zk}_1, \mathbf{c}^*)$ to generate a simulated proof π_{Inner}. It sends $(\pi_{\mathsf{Inner}}, \mathbf{c}^*)$ to

$\mathcal{A}$. It outputs $\mathcal{A}$'s output. It then runs the distinguisher D on $\mathcal{A}$'s view and outputs whatever it outputs.

If B is interacting with the challenger of the hiding game of Com with challenge bit $b = 0$, then it perfectly simulates $\mathcal{A}$ in Hyb_1. Otherwise, when $b = 1$, then B perfectly simulates $\mathcal{A}$ in Hyb_2. Since D can distinguish between $\mathcal{A}$'s output in both executions, then it distinguisher between the case of $b = 0$ and $b = 1$. ■

Subversion Advice-ZK NIZK. We show that for every PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$ and an advice distribution $\mathcal{D}$ such that the distributions $\mathsf{REAL}_{\mathcal{A},\Pi}(\lambda)$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)$ defined in Definition 5.1 are computationally indistinguishable.

We first give the description of the $\mathcal{D}$ and $\mathcal{S}$. Towards this, it will be helpful to consider the following PPT adversary $\mathcal{A}_{\mathsf{ZK}}$ for the advice -ZK property of $\mathsf{ZK} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$. In particular:

1. $\mathcal{A}_{\mathsf{ZK}}$ runs $\mathcal{A}(1^\lambda)$ to generate $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$.
2. It forwards zk_1 as its first message.
3. For each iteration until $\mathcal{A}$ halts:
   (a) When $\mathcal{A}$ queries on (x, w), it outputs (x, w).
   (b) Then, upon receiving zk_2 from the challenger, it internally computes π_{Inner} identically to the honest prover algorithm Prove. In particular, it computes a commitment $\mathbf{c}$ to zk_2 using randomness r and then computes a NIZK proof for the statement $(x, \mathsf{zk}_1, \mathbf{c})$ using witness (zk_2, r).
   (c) It then sends $\pi = (\pi_{\mathsf{Inner}}, \mathbf{c})$ to $\mathcal{A}$.
4. When $\mathcal{A}$ halts, $\mathcal{A}_{\mathsf{ZK}}$ simply outputs $\mathcal{A}$'s view.

Now, since $\mathsf{ZK} = (\mathsf{V}_0, \mathsf{P}, \mathsf{V}_1)$ satisfies advice ZK, we know that for $\mathcal{A}_{\mathsf{ZK}}$ as constructed above there exists an advice distribution $\mathcal{D}_{\mathsf{ZK}}$ and $\mathcal{S}_{\mathsf{ZK}}$ such that $\mathsf{REAL}_{\mathcal{A}_{\mathsf{ZK}},\mathsf{ZK}}(\lambda)$, and $\mathsf{IDEAL}_{\mathcal{S}_{\mathsf{ZK}},\mathsf{ZK},\mathcal{D}_{\mathsf{ZK}}}(\lambda)$ (as defined in Definition 4.1) are computationally indistinguishable. We will use $\mathcal{D}_{\mathsf{ZK}}$ and $\mathcal{S}_{\mathsf{ZK}}$ to define $\mathcal{D}$ and $\mathcal{S}$ respectively in a straightforward way.

Advice Distribution $\mathcal{D}(1^\lambda, \mathsf{CRS})$:

1. Sample an advice d_{ZK} from $\mathcal{D}_{\mathsf{ZK}}(1^\lambda, \mathsf{CRS}_{\mathsf{Inner}})$.
2. Output $d = d_{\mathsf{ZK}}$.

Simulator $\mathcal{S}(1^\lambda, \mathsf{CRS}, x, d)$:

1. Parse $\mathsf{CRS} = (\mathsf{CRS}_{\mathsf{Inner}}, \mathsf{zk}_1)$ and $d = d_{\mathsf{ZK}}$.
2. Run $\mathcal{S}_{\mathsf{ZK}}(1^\lambda, \mathsf{CRS}_{\mathsf{Inner}}, x, d_{\mathsf{ZK}})$ to compute zk_2.
3. Abort if $\mathsf{ZK.Verify}(x, \mathsf{zk}_1, \mathsf{zk}_2) \neq 1$.
4. Compute π_{Inner} like the honest prover algorithm Prove. That is, computes a commitment $\mathbf{c}$ to zk_2 using randomness r and then computes a NIZK proof for the statement $(x, \mathsf{zk}_1, \mathbf{c})$ using witness (zk_2, r).
5. Send $(x, \pi = (\pi_{\mathsf{Inner}}, \mathbf{c}))$ to $\mathcal{A}$ and output its view.

First note that since $\mathcal{D}_{\mathsf{ZK}}$ can be computed by an S-time algorithm, we have that $\mathcal{D}$ also can be computed by an S-time algorithm. To conclude the proof we observe that $\mathcal{A}$'s view in $\mathsf{REAL}_{\mathcal{A},\Pi}(\lambda)$ is identical to $\mathcal{A}_{\mathsf{ZK}}$'s view in $\mathsf{REAL}_{\mathcal{A}_{\mathsf{ZK}},\mathsf{ZK}}(\lambda)$. Similarly, $\mathcal{A}$'s view in $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda)$ is identical to $\mathcal{A}_{\mathsf{ZK}}$'s view in $\mathsf{IDEAL}_{\mathcal{S}_{\mathsf{ZK}},\mathsf{ZK},\mathcal{D}_{\mathsf{ZK}}}(\lambda)$. Then, the proof follows by the computationally indistinguishability of $\mathsf{REAL}_{\mathcal{A}_{\mathsf{ZK}},\mathsf{ZK}}(\lambda)$ and $\mathsf{IDEAL}_{\mathcal{S}_{\mathsf{ZK}},\mathsf{ZK},\mathcal{D}_{\mathsf{ZK}}}(\lambda)$.

6 Accountable Soundness

In this section, we define accountable soundness for *any* non-interactive argument system in the CRS model which captures both NIZKs and SNARGs.

There are two aspects to our definition, depending on whether the CRS is honestly or maliciously generated. When the CRS is honestly generated, we require that the scheme satisfies the same traditional security requirements (e.g., either be a NIZK or a SNARG). In the latter case, we want to prevent the authority from running a service in which it provides accepting proofs for (either valid or invalid) statements without receiving the corresponding witness. If it does run such a service, we should be able to implicate the malicious authority for its wrongdoing. We define an extractor that interacts with the malicious authority and comes up with a piece of evidence τ that can be presented to a Judge, defined by a Judge algorithm, who verifies whether the presented piece of evidence is valid. At the same time, we require that no efficient adversary can compute a piece of evidence that can falsely accuse an honest authority.

Consider a non-interactive argument system consisting of a triplet of algorithms $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$. In addition, we define a PPT algorithm Judge, which is necessary to define accountable soundness for maliciously

- $b \leftarrow \mathsf{Judge}(\mathsf{CRS}, \tau)$ where $b \in \{\mathsf{honest}, \mathsf{corrupted}\}$: The algorithm receives as input the (possibly corrupted) CRS and some transcript τ, and outputs a bit b, indicating whether the CRS CRS is corrupted of not.

Definition 6.1 (Argument System with Accountable Soundness)**.** *Let $L \in \mathsf{NP}$. We say that a non-interactive argument system $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ for L achieves* accountable soundness *with respect to distribution $\mathcal{D}$ if all the following properties hold:*

1. **(Accountability:)** *There exists a PPT extractor $\mathcal{E}$, such that for every (possibly stateful) PPT adversary $\mathcal{A}$ there is a negligible function $\mu(\cdot)$ such that for all λ:*

$$|\Pr[\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda) = 1]| \leq \mu(\lambda)$$

where random variables $\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A}}(\lambda)$, $\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda)$ are:

$\underline{\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A}}(\lambda)}$*:*

- $\mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)$*;*
- $x \leftarrow \mathcal{D}(1^\lambda)$*;*
- *Query $\mathcal{A}$ on x and receive back a proof π.*
- *The output is 1 iff* $\mathsf{Verify}(\mathsf{CRS}^*, x, \pi) = 1$.

$\underline{\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda)}$*:*

- $\mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)$*;*
- $\mathcal{E}(\mathsf{CRS}^*)$ *outputs some query x.*
- *The adversary $\mathcal{A}$ receives x and outputs a proof π.*
- *Give π to $\mathcal{E}$ which replies with τ.*
- *The output is 1 iff* $\mathsf{Judge}(\mathsf{CRS}^*, \tau) = \mathsf{corrupted}$.

2. **(Defamation-free:)** *For every PPT adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$, such that for all $\lambda \in \mathbb{N}$:*

$$\Pr[\mathsf{Judge}(\mathsf{CRS}, \tau) = \mathsf{corrupted} : \ \mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda);\ \tau \leftarrow \mathcal{A}(\mathsf{CRS})] \leq \mu(\lambda)\ .$$

We next define NIZKs (resp., SNARGs) with accountable soundness.

Definition 6.2. *A non-interactive argument system* $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ *for any* NP *language* L *is a NIZK (resp., SNARG) with accountable soundness w.r.t. distribution* $\mathcal{D}$ *if* $(\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ *is a NIZK (resp., SNARG) for* L *and it satisfies accountable soundness w.r.t.* $\mathcal{D}$*.*

Notation. For a distribution D over $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$ (i.e., pairs of instances and their associated witnesses), we denote by $M(\mathcal{DR}_L)$ the marginal distribution over the instances only where the distributions are parameterized by the security parameter.

Comparison with Ananth et al. [AADG21]. Our definition is inspired by the definition of accountability of Ananth et al. [AADG21]. Their definition is with respect to an authority who helps others to open witnesses of proofs proven with respect to the maliciously generated CRS^*. We highlight few differences from the definition of [AADG21]:

- **Universal extractor:** The definition of [AADG21] requires that for every malicious authority $\mathcal{A}$ there exists an extractor, whereas we require a universal extractor that works for all possible malicious authorities. Ours is a stronger definition, and we believe it is more natural. Specifically, if one identifies that the authority misbehaves, our definition guarantees explicit instructions on how to hold the authority accountable, as opposed to the definition in [AADG21] which only guarantees the existence of such instructions which may additionally depend on the specific code of the authority.
- **Probability of errors:** The definition in [AADG21] allows a gap between the probability that the adversary succeeds in the real, vs. the probability that the extractor succeeds to hold the authority accountable in the ideal. Specifically, it just requires that if the authority succeeds with some non-negligible probability in the real, then the extractor succeeds with some non-negligible probability in the ideal. The gap between the two might be large. We require that the gap between the two is only negligible.

7 Constructions of Non-interactive Arguments with Accountable Soundness

In this section, we provide several constructions of non-interactive arguments with accountable soundness. Section 7.1 gives a construction for languages in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ and its generalization. But, this construction crucially relies on the ability to efficiently sample NO instances along with an efficiently checkable witness of non-membership. Towards capturing general languages, we discuss sparse languages in Section 7.2 and Section 7.3, and also give a general feasibility result for NP in Section 7.4. A particularly interesting aspect of our constructions is that we only add a (short) random string to the CRS of an non-interactive argument system, and do not change the prover and the verifier algorithms. Thus, our constructions preserve the privacy (e.g., ZK) and the efficiency (e.g., succinctness and prover/verifier time) properties of the underlying non-interactive argument.

7.1 Construction for Languages in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$

Let L be a $\mathsf{NP} \cap \mathsf{coNP}$ language with relations R_L and $R_{\overline{L}}$ over L and its complement $\overline{L}$ respectively. Let $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be ***any*** non-interactive argument system for L. We present the construction Π below followed by the security theorem and its proof.

Construction 7.1: Non-interactive argument with accountable sndness

GenCRS(1^λ): Output $\mathsf{CRS} \leftarrow \mathsf{GenCRS}'(1^\lambda)$.
Prove(CRS, x, w): Output $\pi \leftarrow \mathsf{Prove}'(\mathsf{CRS}, x, w)$.
Verify (CRS, x, π): Output $b \leftarrow \mathsf{Verify}'(\mathsf{CRS}, x, \pi)$.
Judge (CRS*, $\tau = (x^*, \pi^*, \overline{w}^*)$):

- Output $b = \mathsf{corrupted}$ iff $\mathsf{Verify}(\mathsf{CRS}^*, x^*, \pi^*) = 1 \ \wedge \ R_{\overline{L}}(x^*, \overline{w}^*) = 1$.

Theorem 7.2. *Let L be **any** $\mathsf{NP} \cap \mathsf{coNP}$ language, and R_L and $R_{\overline{L}}$ be the relations for L and its complement $\overline{L}$. Let $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be a non-interactive adaptively sound argument for the language L. Then, the above construction Π is a non-interactive argument system for L with accountable soundness w.r.t. any distribution $\mathcal{D}$ over $\{0,1\}^*$ for which there exists a distribution $\mathcal{DR}_{\mathsf{No}}$ over $R_{\overline{L}}$ such that the marginal distribution $M(\mathcal{DR}_{\mathsf{No}})$ and $\mathcal{D}$ are computationally indistinguishable. Further, if Π' is a NIZK (resp., SNARG) then Π is a NIZK (resp., SNARG) with accountable soundness for $\mathcal{D}$.*

Proof. The completeness and soundness properties hold by the completeness and soundness of the non-interactive argument Π'. Moreover, GenCRS, Prove and Verify are identical to GenCRS', Prove' and Verify' respectively. Therefore, if Π' is a NIZK (resp., SNARG), then so is Π. Next, we show that the accountability and defamation free properties hold:

Accountability. We describe the extractor $\mathcal{E}$:

1. It receives (possibly maliciously generated) CRS^* generated by the authority.
2. Sample $(x, \overline{w})$ from $\mathcal{DR}_{\mathsf{No}}(1^\lambda)$.
3. Query $\mathcal{A}$ on x and receive back π.
4. Output $\tau = (x, \overline{w}, \pi)$.

We claim that the view of the malicious authority is indistinguishable between $\mathsf{AccSnd.REAL}$ and $\mathsf{AccSnd.IDEAL}$. In $\mathsf{AccSnd.REAL}$, the authority receives x sampled according to $\mathcal{D}$. In the ideal, the extractor samples (x, w) according to $\mathcal{DR}_{\mathsf{No}}$, and then gives x to the authority. This is equivalent to sampling from $M(\mathcal{DR}_{\mathsf{No}})$, and according to our assumption, $M(\mathcal{DR}_{\mathsf{No}}) \approx_c \mathcal{D}$.

We conclude that the authority provides a valid proof to the extractor query with probability negligibly close to the probability in the real experiment. Whenever the authority provides a valid proof in the ideal, the extractor provides τ that makes Judge accept, and the output of $\mathsf{AccSnd.IDEAL}$ is 1.

Defamation Free. We show that for an honestly generated CRS $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$, no PPT adversary $\mathcal{A}$ can output $\tau = (x^*, \pi^*, \overline{w}^*)$ for which Judge accepts with non-negligible probability.

Suppose that there exists a PPT adversary $\mathcal{A}$ and a non-negligible function $\epsilon(\cdot)$ such that $\Pr[\mathsf{Judge}(\mathsf{CRS}, \mathcal{A}(\mathsf{CRS})) = \mathsf{corrupted}] \geq \epsilon(\lambda)$ for an honestly generated $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$. We show that $\mathcal{A}$ can be used to construct a PPT corrupted prover P that breaks the adaptive soundness property of the inner NIZK system $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ with non-negligible probability: Given an honestly generated $\mathsf{CRS}' \leftarrow \mathsf{GenCRS}'(1^\lambda)$ as an input, P sets $\mathsf{CRS} = \mathsf{CRS}'$ and runs $\mathcal{A}(\mathsf{CRS})$ to obtain it's output τ. It parses $\tau = (x^*, \pi^*, \overline{w}^*)$ and outputs (x^*, π^*).

First, since $\mathcal{A}$ is a PPT algorithm, then so is P. Second, P perfectly simulates $\mathcal{A}$ in the defamation-free experiment. Therefore, $\mathcal{A}$ outputs $\tau = (x^*, \pi^*, \overline{w}^*)$ such that $\Pr[\mathsf{Judge}(\mathsf{CRS}, \tau) =$

$\mathsf{corrupted}] \geq \epsilon(\lambda)$. That is, $\mathsf{Verify}(\mathsf{CRS}, x^*, \pi^*) = 1$ *and* $(x^*, \overline{w}^*) \in R_{\overline{L}}$. Thus, we have $\mathsf{Verify}(\mathsf{CRS}, x^*, \pi^*) = \mathsf{Verify}'(\mathsf{CRS}, x^*, \pi^*) = 1$ and $x^* \notin L$. Overall, the following violates the adaptive soundness property of Π':

$$\begin{aligned}\Pr[\mathsf{Soundness}_{\Pi',P}(\lambda) = 1] &= \Pr[\mathsf{Verify}'(\mathsf{CRS}, x^*, \pi^*) = 1 \ \wedge \ x^* \notin L] \\ &= \Pr[\mathsf{Verify}(\mathsf{CRS}, x^*, \pi^*) = 1 \ \wedge \ x^* \notin L] \\ &= \Pr[\mathsf{Verify}(\mathsf{CRS}, x^*, \pi^*) = 1 \ \wedge \ (\exists \overline{w}^* : (x^*, \overline{w}^*) \in R_{\overline{L}})] \geq \epsilon(\lambda) \ .\end{aligned}$$

□

7.1.1 Generalization

The class $\mathsf{NP} \cap \mathsf{coNP}$ is both theoretically and practically relevant as it contains many interesting cryptographic languages. However, for a language L to be in $\mathsf{NP} \cap \mathsf{coNP}$, *all* NO instances must have an efficiently checkable witness for non-membership in L. But Theorem 7.2's proof readily extends to languages L where only sufficiently large (but still only negligible fraction) of the NO instances have such a witness. More formally, Theorem 7.2's proof only requires the existence of a subset L_{No} of its complement $\overline{L}$ along with an efficiently checkable relation R_{No} over L_{No}. In particular, we require that for any x: $x \in L_{\mathsf{No}}$ iff $\exists \overline{w}$ s.t. $R_{\mathsf{No}}(x, \overline{w}) = 1$. This is a strict generalization of $\mathsf{NP} \cap \mathsf{coNP}$ and it allows us to capture more languages.

Then, consider a minor modification of the construction Π where the Judge algorithm uses the relation R_{No} instead of $R_{\overline{L}}$. Then, this modified construction achieves accountable soundness for distributions $\mathcal{D}$ as long as there exist a computationally indistinguishable distribution $\mathcal{DR}_{\mathsf{No}}$ over R_{No}. We formalize this in the subsequent theorem whose proof is identical to that of Theorem 7.2.

Theorem 7.3. *Let L be **any** NP language, R_L be the its relation. For $L_{\mathsf{No}} \subseteq \bar{L}$, let R_{No} be the its relation. Let Π' be a non-interactive adaptively sound argument for L. Then, there exists a non-interactive argument system with accountable soundness w.r.t. any distribution $\mathcal{D}$ over $\{0,1\}^*$ assuming the existence of a distribution $\mathcal{DR}_{\mathsf{No}}$ over R_{No} such that the marginal distribution $M(\mathcal{DR}_{\mathsf{No}})$ and $\mathcal{D}$ are computationally indistinguishable. Further, if Π' is a NIZK (resp., SNARG) then Π is a NIZK (resp., SNARG) with accountable soundness for $\mathcal{D}$.*

7.1.2 Examples

We present examples of several languages beyond $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ for which accountable soundness is achievable. A key feature of these languages is that the instances contain commitments/encryptions where the plaintext satisfies an efficiently verifiable relation (e.g., preimage of a one-way function). For such languages, we can exhibit L_{No} (and R_{No}) and $\mathcal{DR}_{\mathsf{No}}$ relying on semantic-security of the commitments/encryptions.

GMW Compiler on Yao's 2PC. The GMW compiler [GMW87] is a central compilation technique in cryptography that allows to upgrade *any* multi-party computation protocol with semi-honest security to malicious security. For simplicity, we discuss applying the GMW compiler to Yao's semi-honest secure 2PC protocol [Yao86]. Recall, Yao's protocol relies on a two-round oblivious transfer protocol $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ where OT_1 (resp., OT_2) computes receiver's (resp., sender's) message, and the receiver uses OT_3 to compute its output, and a garbled circuit $\mathsf{Garble} = (\mathsf{Gen}, \mathsf{Eval})$. In the protocol, the receiver first generates and sends ot_1 on its input $b \in \{0,1\}$, followed by the sender garbling the circuit $C(x, \cdot)$ that has its input x hardwired and generating appropriate ot_2

w.r.t. the keys obtained by the garbling. To get malicious security, the sender also sends a NIZK proof for the following language that shows that the message is well-formed:

$$L_{\mathsf{GMW}} = \left\{ (\mathsf{ot}_1, \mathsf{ot}_2, \hat{C}) : \exists (x, r_{\mathsf{OT}}, r_{\mathsf{GC}}) \text{ s.t. } \begin{smallmatrix} (\hat{C}, k_0, k_1) = \mathsf{Gen}(C(x,\cdot); r_{\mathsf{GC}}) \\ \mathsf{ot}_2 = \mathsf{OT}_2(\mathsf{ot}_1, k_0, k_1; r_{\mathsf{OT}}) \end{smallmatrix} \right\} .$$

This language may not be in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ as $\overline{L_{\mathsf{GMW}}}$ consists of strings $(\mathsf{ot}_1, \mathsf{ot}_2, \hat{C})$ where $\hat{C}$ may be outside the range of the Gen algorithm for which no efficiently checkable witness may exist. However, consider the subset $L_{\mathsf{No}} \subseteq \overline{L_{\mathsf{GMW}}}$ containing tuples $(\mathsf{ot}_1, \mathsf{ot}_2, \hat{C})$ where $\hat{C}$ garbles a constant function:

$$L_{\mathsf{No}} = \left\{ (\mathsf{ot}_1, \mathsf{ot}_2, \hat{C}) : \exists (b, r, x, r_{\mathsf{OT}}, r_{\mathsf{GC}}) \text{ s.t. } \begin{smallmatrix} \mathsf{ot}_1 = \mathsf{OT}_1(b; r); z = C(x, b); (\hat{C}, k_0, k_1) = \mathsf{Gen}(z; r_{\mathsf{GC}}) \\ \mathsf{ot}_2 = \mathsf{OT}_2(\mathsf{ot}_1, k_0, k_1; r_{\mathsf{OT}}) \end{smallmatrix} \right\} .$$

The associated relation R_{No} takes an instance $(\mathsf{ot}_1, \mathsf{ot}_2, \hat{C})$ and a witness $(b, r, x, r_{\mathsf{OT}}, r_{\mathsf{GC}})$, and verifies that $\hat{C}$ is indeed a garbling of the constant function that outputs $z = C(x, b)$. Given any NIZK for L_{GMW}, we can obtain another NIZK that achieves accountable soundness w.r.t. the uniform distribution $\mathcal{D}$ over L_{GMW}: the required distribution $\mathcal{DR}_{\mathsf{No}}$ is the uniform distribution over $R_{L_{\mathsf{No}}}$, and computational indistinguishability follows from the security of garbling.

ZCash's POUR transaction. The ZCash cryptocurrency [BSCG+14] uses ZK-SNARKs for privacy. ZK-SNARKs are NIZKs with short proofs and a polylogarithmic time verifier. In ZCash, such NIZKs are used for its POUR transaction that allows any user to pour the value of two of its coins $(c_1^{\mathsf{old}}, c_2^{\mathsf{old}})$ into two new coins while preserving the monetary value. Here, a user needs to generate its new coins $c_1^{\mathsf{new}}, c_2^{\mathsf{new}}$ and post hiding commitments to the description of these coins along with a NIZK proof that proves (among other things) (a) ownership of old coins $c_1^{\mathsf{old}}, c_2^{\mathsf{old}}$, (b) well-formedness of new coins $c_1^{\mathsf{new}}, c_2^{\mathsf{new}}$, and (c) the total value in old coins and new coins are equal, that is, $v_1^{\mathsf{new}} + v_2^{\mathsf{new}} = v_1^{\mathsf{old}} + v_2^{\mathsf{old}}$. At a high level, L_{No} contains instances which are identical except that the condition (c) is not satisfied. The hiding of the commitment scheme will ensure that random instances from L_{POUR} and L_{No} are indistinguishable.

Please refer to [BSCG+14] for a formal description of L_{POUR}; we only discuss a simpler abstraction below. The instances in L_{POUR} are tuples $(\mathsf{rt}, \mathsf{sn}_1^{\mathsf{old}}, \mathsf{sn}_2^{\mathsf{old}}, \mathbf{c}_1^{\mathsf{new}}, \mathbf{c}_2^{\mathsf{new}})$ with the witness $\{(\mathsf{path}_i, \mathbf{c}_i^{\mathsf{old}}, r_i^{\mathsf{old}}, v_i^{\mathsf{old}}, r_i^{\mathsf{new}}, v_i^{\mathsf{new}}, \mathsf{sn}_i^{\mathsf{new}})\}_{i \in [1,2]}$ where:

1. for both $i \in \{1, 2\}$:
   (a) path_i is the Merkle membership proof for $\mathbf{c}_i^{\mathsf{old}}$ w.r.t. the Merkle root rt.
   (b) $\mathbf{c}_i^{\mathsf{old}} = \mathsf{Com}(\mathsf{sn}_i^{\mathsf{old}} || v_i^{\mathsf{old}}; r_i^{\mathsf{old}})$; $\mathbf{c}_i^{\mathsf{new}} = \mathsf{Com}(\mathsf{sn}_i^{\mathsf{new}} || v_i^{\mathsf{new}}; r_i^{\mathsf{new}})$.
2. $v_1^{\mathsf{old}} + v_2^{\mathsf{old}} = v_1^{\mathsf{new}} + v_2^{\mathsf{new}}$.

Note that L_{POUR} may not be in $\mathsf{NP} \cap \mathsf{coNP}$ as $\overline{L_{\mathsf{POUR}}}$ contains instances where strings $\mathbf{c}^{\mathsf{new}}$ may be outside the range of Com, and there may not be any efficiently checkable witness for this. However, let L_{No} be the subset of $\overline{L_{\mathsf{POUR}}}$ that samples instances identically to L_{POUR} except that condition (2) doesn't hold for the new coins. One can verify that L_{No} has an efficient-to-compute relation R_{No}.

For accountable soundness, consider the distribution $\mathcal{D}$ that samples instances from L_{POUR} while using uniform randomness to compute $\mathbf{c}$ where $\mathcal{D}$ may choose values (and other attributes of the coins) arbitrarily. Then, we can transform any NIZK for L_{POUR} to additionally achieve accountable soundness for $\mathcal{D}$: the required distribution $\mathcal{DR}_{\mathsf{No}}$ over $R_{L_{\mathsf{No}}}$ is the uniform distribution,

and the required computational indistinguishability follows from the hiding of Com. Recall that our compiler for achieving accountable soundness only adds the Judge algorithm without changing the underlying NIZK's CRS as well as $\mathsf{Prove}/\mathsf{Verify}$ algorithms. Therefore, we can already upgrade ZK-SNARKs, currently implemented in ZCash, to satisfy accountable soundness.

Non-interactive Commitments. Let Com be any perfectly binding non-interactive commitment scheme. Then, consider the language $L_{\mathbf{c},1} = \{c : \exists r \text{ s.t. } c = \mathsf{Com}(1; r)\}$. This language may not be in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ as $\overline{L_{\mathbf{c},1}}$ also consists of strings outside the range of Com for which no efficiently checkable witness may exist. However, for $L_{\mathsf{No}} = \{c : \exists r \text{ s.t. } c = \mathsf{Com}(0; r)\}$ there exists an efficiently checkable relation R_{No} such that for every $c \in L_{\mathsf{No}}$ there exists some witness r for which $R_{\mathsf{No}}(c, r) = 1$ (and vice versa). For accountable soundness, a particularly interesting distribution $\mathcal{D}$ is the uniform distribution over $L_{\mathbf{c},1}$ for which the uniform distribution over $R_{L_{\mathsf{No}}}$ satisfies the required computational indistinguishability due to the hiding of Com.

Validity of Ciphertexts. Next, we consider the NP language that consists of well-formed ciphertexts where the plaintext satisfies a publicly-verifiable relation. NIZKs for such languages have appeared in several different applications including providing range proofs [BBB+18] and building verifiable timed signatures [TBM+20]. For simplicitly, we choose to describe the language w.r.t. encryption schemes but our ideas would also extend to languages where that use primitives like commitments or time-lock puzzles in place of encryption schemes.

As mentioned above the language is parametrized by some PKE encryption scheme $(\mathsf{KgGen}, \mathsf{Enc}, \mathsf{Dec})$ and a signature scheme $(\mathsf{KgGen}, \mathsf{Sign}, \mathsf{Ver})$. The language is as follows:

$$L = \{(\mathsf{pk}, \mathsf{vk}, \mathsf{ct}) : \exists (m, \sigma, r) : \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, m||\sigma; r) \wedge \mathsf{Ver}(vk, m, \sigma) = 1\} \ .$$

This language may not be in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ as $\overline{L}$ consists of strings $(\mathsf{pk}, \mathsf{vk}, \mathsf{ct})$ where ct may be outside the range of the Enc algorithm for which no efficienctly checkable witness may exist. However, consider the following subset $L_{\mathsf{No}} \subseteq \bar{L}$ which contains tuples $(\mathsf{pk}, \mathsf{vk}, \mathsf{ct})$ where ct encrypts 0^λ:

$$L_{\mathsf{No}} = \left\{(\mathsf{pk}, \mathsf{vk}, \mathsf{ct}) : \exists r : \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, 0^\lambda; r)\right\} \ .$$

For the above language, there does exists an associated relation R_{No} that on inputs a statement $(\mathsf{pk}, \mathsf{vk}, \mathsf{ct})$ and a witness r outputs 1 iff the ct is an encryption of 0^λ using randomness r. For accountable soundness, consider the distribution $\mathcal{D}$ that samples instances from L while using randomness used to sample pk, vk and compute the ciphertext ct uniformly at random. In particular, $\mathcal{D}$ may choose the plaintext to be encrypted arbitarily. Then, we can transform any NIZK for L to additionally achieve accountable soundness for $\mathcal{D}$: the required distribution $\mathcal{D}\mathcal{R}_{\mathsf{No}}$ over $R_{L_{\mathsf{No}}}$ is the uniform distribution and the required computational indistinguishability would follow from the semantic security of the PKE scheme.

Hash-time-lock Contracts. Hash-time-lock contracts (HTLC) are a type of smart contracts supported by Bitcoin and used in several blockchain applications. An HTLC allows a party to redeem a transaction containing some hash value h if they can produce a pre-image (under SHA256) of h. Such contracts along with NIZKs have recently found applications in zero-knowledge contingency payments to execute fair sale of NP secrets [ZKC]. Specifically, Alice wanting to sell a witness w for some NP statement x generates a secret-key k for some encryption scheme and computes (h, ct) where $h = \mathsf{SHA256}(k)$ and ct is an encryption of w under k. It additionally computes a NIZK proof π to show the well-formedness of (h, ct) and sends the tuple (h, ct, π) to Bob. Then, Bob generates a hash-time-lock contract w.r.t. the hash h included by Alice. To redeem, Alice needs to post the

pre-image of h on the blockchain which would also allow Bob to learn the witness w via decrypting ct using the posted pre-image.

More formally, Alice generates a NIZK proof for the following language:

$$L = \{(x, h, \mathsf{ct}) : \exists (k, w) \text{ s.t. } h = \mathsf{SHA256}(k) \wedge w = \mathsf{Dec}(k, \mathsf{ct}) \wedge R(x, w) = 1\} \ .$$

This language may not be in $\mathsf{NP} \cap \mathsf{co\text{-}NP}$ as $\overline{L}$ consists of strings (x, h, ct) where ct may be outside the range of the Enc algorithm and no efficienctly checkable witness may exist. However, consider the following subset $L_{\mathsf{No}} \subseteq \overline{L}$ which contains tuples (x, h, ct) where ct encrypts a non-witness (e.g., a special symbol $\perp$):

$$L_{\mathsf{No}} = \{(x, h, \mathsf{ct}) : \exists (k, r) \text{ s.t. } h = \mathsf{SHA256}(k) \wedge \mathsf{ct} = \mathsf{Enc}(k, \perp; r)\} \ .$$

For the above language, there does exist an associated relation R_{No} that on inputs a statement (x, h, ct) and a witness k, r outputs 1 iff ct is an encryption of $\perp$ under key k and randomness r.

For accountable soundness, consider the distribution $\mathcal{D}$ that samples instances from L while using randomness used to sample k and compute the ciphertext ct uniformly at random. In particular, $\mathcal{D}$ may choose the plaintext to be encrypted arbitarily. Then, we can transform any NIZK for L to additionally achieve accountable soundness for $\mathcal{D}$: the required distribution $\mathcal{DR}_{\mathsf{No}}$ over $R_{L_{\mathsf{No}}}$ is the uniform distribution and the required computational indistinguishability would follow from the semantic security of the encryption scheme. Note here that we require semantic security even in the presence of the leakage h which is the hash of the corresponding secret-key. If h is sufficiently compressing then we can show that the secret-key has enough min-entropy and relyin on the standard notion of semantic security. Else, modelling $\mathsf{SHA256}$ as a one-way function, we would need a symmetric-key encryption scheme which is leakage-resilient against computational leakages of secret-key.

7.2 Construction for Sparse Languages

In this section, we focus on languages where it may not be possible to sample NO instances along with their witness. In particular, the idea is to "force" a successful defamation-freeness adversary to find accepting proofs for NO instances. We rely on the "sparsity" of the underlying language L to achieve this. We next formally define the notion of δ-*sparsity* and then state our theorem.

Sparsity. For security parameter λ, let $L \subseteq \{0,1\}^{\ell(\lambda)}$ be a language with instances of length $\ell(\lambda)$. Then, for a function $\delta = \delta(\lambda)$, L is δ-sparse if $|L| \leq 2^{\ell} \cdot \delta$.

Theorem 7.4. *Let L be a δ-sparse language over ℓ bit strings and Π' be a non-interactive adaptively sound argument for L. Then, there exists a non-interactive argument system Π with accountable soundness w.r.t. any distribution $\mathcal{D} = \mathcal{DR}_{\mathsf{yes}}$ over R_L as long as $M(\mathcal{DR}_{\mathsf{yes}})$ is computationally indistinguishable from $\mathcal{U}_\ell$ and $\delta^2 \cdot 2^{\ell}$ is negligible in λ. Further, if Π' is a NIZK (resp., SNARG) then Π is a NIZK (resp., SNARG) with accountable soundness for $\mathcal{D}$.*

Proof. Towards proving Theorem 7.4, we first outline the construction and discuss the security proof.

Construction We now give our construction for δ-sparse languages L, for, e.g., $\delta = 2^{-3\ell/4}$ and $\ell \in \mathsf{poly}(\lambda)$. Let L be ***any*** such language in NP, and let $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be a non-interactive argument system for L. We present the following construction $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ of an non-interactive argument with accountable soundness:

Construction 7.5: Accountable soundness for sparse languages

$\mathsf{GenCRS}(1^\lambda)$:

1. Run $\mathsf{CRS}_\Pi = \mathsf{GenCRS}'(1^\lambda)$.
2. Sample $x^* \leftarrow \mathcal{U}_\ell$.
3. Output $\mathsf{CRS} = (\mathsf{CRS}_\Pi, x^*)$.

$\mathsf{Prove}(\mathsf{CRS}, x, w)$ where $\mathsf{CRS} = (\mathsf{CRS}_\Pi, x^*)$:

1. Output $\pi = \mathsf{Prove}'(\mathsf{CRS}_\Pi, x, w)$.

$\mathsf{Verify}\,(\mathsf{CRS}, x, \pi)$ where $\mathsf{CRS} = (\mathsf{CRS}_\Pi, x^*)$:

1. Output $b = \mathsf{Verify}'(\mathsf{CRS}_\Pi, x, \pi)$.

$\mathsf{Judge}\,(\mathsf{CRS} = (\mathsf{CRS}_\Pi, x^*), \tau = (x, w, \pi))$:

1. If CRS is not well-formed (i.e., $\mathsf{CRS} \neq (\mathsf{CRS}_\Pi, x^*)$) then output $\mathsf{corrupted}$.
2. Otherwise, output $\mathsf{corrupted}$ iff $R_L(x, w) = 1$, and $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus x, \pi) = 1$. (Note that π is a proof for the statement is $x^* \oplus x$ and not x.)

Security Proof. The completeness and soundness properties hold by the completeness and soundness of the non-interactive argument Π'. Moreover, GenCRS is identical to GenCRS' except that we add random string x^* of length ℓ in the CRS, and Prove and Verify algorithms ignore x^* and behave identically to Prove' and Verify'. Therefore, if Π' is a NIZK (resp., SNARG), we can conclude that so is Π.

Next, we show that the accountability and defamation free properties hold:

Accountability. We present the extractor Ext. Let $\mathcal{A}$ be some PPT adversary. The extractor has one black-box query to the adversary $\mathcal{A}$, after $\mathcal{A}$ gives out CRS^*.

1. The extractor is invoked on an input CRS^*. If CRS^* is not of the form of (CRS_Π, x^*) then just use $x^* = 0$.
2. It samples $(x, w) \leftarrow \mathcal{DR}_{\mathsf{yes}}$ and queries $\mathcal{A}$ on $x^* \oplus x$, to receive π.
3. It outputs $\tau = (x, w, \pi)$.

Recall that in real world, $\mathcal{A}$ is queried on some random x sampled from $\mathcal{DR}_{\mathsf{yes}}$ whereas in the ideal world the extractor samples some $x \in L$, but then queries $\mathcal{A}$ on the input $x \oplus x^*$. We proceed to show that $\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A}}(\lambda)$, $\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda)$ are negligibly close via the following sequence of hybrids.

- Hybrid Hyb_0: This is identical to $\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda)$. More specifically,
  1. $\mathsf{CRS} = (\mathsf{CRS}^*, x^*) \leftarrow \mathcal{A}(\lambda)$.
  2. Sample $(x, w) \leftarrow \mathcal{DR}_{\mathsf{Yes}}$.
  3. Query $\mathcal{A}$ on x to get back π.
  4. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$.

- Hybrid Hyb_1: This is identical to Hyb_0 except that $x \leftarrow \mathcal{U}_\ell$.
- Hybrid Hyb_2: This is identical to Hyb_1 except that $\mathcal{A}$ is queried on the statement $x \oplus x^*$ instead of querying it on x (as in Hyb_1) where $x \leftarrow \mathcal{U}_\ell$. Furthermore, the winning condition of this hybrid is now changed: Hyb_2 outputs 1 only if $\mathsf{Verify}(\mathsf{CRS}, x \oplus x^*, \pi) = 1$.
- Hybrid Hyb_3: This is identical to Hyb_2 except that we switch to sampling $(x, w) \leftarrow \mathcal{DR}_{\mathsf{Yes}}$. Recall that the output of hybrid is 1 if $\mathsf{Verify}(\mathsf{CRS}, x \oplus x^*, \pi)$.
- Hybrid Hyb_4: This hybrid is identical to Hyb_3 except that the output condition of this hybrid is now changed: Hyb_4 output 1 if $\mathsf{Verify}(\mathsf{CRS}, x \oplus x^*, \pi) = 1 \wedge R_L(x, w) = 1$. This is the identical to $\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}$.

For each $i \in \{0, \ldots, 4\}$, let p_i be the probability that the hybrid Hyb_i outputs 1. To conclude the proof we need to argue that there exists some negligible function μ such that $|p_0 - p_4| \leq \mu(\lambda)$. First, by the indistinguishability of $M(\mathcal{DR}_{\mathsf{Yes}})$ and $\mathcal{U}_\ell$, we can conclude that there exist negligible functions μ_{01}, μ_{23} such that $|p_0 - p_1| \leq \mu_{01}$ as well as $|p_2 - p_3| \leq \mu_{23}$. Furthermore, $p_1 = p_2$ as the two hybrids Hyb_1 and Hyb_2 are identical. Finally, we argue that $p_3 = p_4$: the only difference between the hybrids Hyb_3 and Hyb_4 are their respective winning conditions: in particular, Hyb_3 outputs 1 only if $\mathsf{Verify}(\mathsf{CRS}, x \oplus x^*, \pi) = 1$ whereas Hyb_4 additionally also need $R_L(x, w) = 1$. However, note that in both hybrids $R_L(x, w) = 1$ and so $p_3 = p_4$. Therefore, $|p_0 - p_4|$ is upperbounded by the negligible function $\mu = \mu_{01} + \mu_{23}$.

Defamation free. We show that it is infeasible to frame an honestly generated CRS. First, we claim that

$$\Pr_{x^* \leftarrow \mathcal{U}_\ell}[\exists x \in L \text{ s.t. } x^* \oplus x \in L] \leq \sum_{x \in L} \Pr_{x^* \leftarrow \mathcal{U}_\ell}[x^* \oplus x \in L] \leq |L| \frac{|L|}{2^\ell} = \delta^2 \cdot 2^\ell \ ,$$

which is negligible according to our assumption in the theorem statement.

Now, fix some PPT adversary $\mathcal{A}$. Denote by $\mathsf{win}_{\mathcal{A}}$ the event in which $\mathsf{CRS} = (\mathsf{CRS}_\Pi, x^*)$ is chosen honestly according to $\mathsf{GenCRS}(1^\lambda)$, and then $\mathcal{A}$, on input CRS outputs (x, w, π) such that $\mathsf{Judge}((\mathsf{CRS}_\Pi, x^*), (x, w, \pi)) = \mathsf{corrupted}$.

$$\begin{aligned} \Pr[\mathsf{win}_{\mathcal{A}}] &= \Pr[\mathsf{win}_{\mathcal{A}} \ \wedge \ x \oplus x^* \notin L] + \Pr[\mathsf{win}_{\mathcal{A}} \ \wedge \ x \oplus x^* \in L] \\ &\leq \Pr[\mathsf{win}_{\mathcal{A}} \ \wedge \ x \oplus x^* \notin L] + \delta^2 \cdot 2^\ell \end{aligned}$$

where the latter holds since the probability is taken also over the choice of the x^* (as part of the honestly generated CRS), and thus the probability that there exists a $x \in L$ such that $x \oplus x^* \in L$ is at most $\delta^2 \cdot 2^\ell$. Thus, if there exists an adversary that can break defamation free, we can use the adversary to break the soundness of the underlying NIZK: on an honestly generated $\mathsf{CRS}_\Pi \leftarrow \mathsf{GenCRS}'(1^\lambda)$ we choose a random x^* and run $\mathcal{A}$ on (CRS_Π, x^*). When $\mathcal{A}$ outputs (x, w, π) such that $\mathsf{Judge}((\mathsf{CRS}_\Pi, x^*), (x, w, \pi)) = \mathsf{corrupted}$ then with non-negligible probability it holds that $x \oplus x^* \notin L$, but $\mathsf{Verify}'(\mathsf{CRS}_\Pi, x^* \oplus x, \pi) = 1$, in contradiction to the soundness property of the underlying NIZK system. □

7.2.1 Examples of Sparse Languages

We next present two examples of languages that satisfy the necessary sparseness requirements.

PRG language. As an example of a language that is captured by this construction, consider the langage which consists of outputs of pseudorandom generators. Let $G : \{0,1\}^\lambda \to \{0,1\}^{\ell(\lambda)}$ be a pseudorandom generator with expansion factor $\ell(\lambda)$. Consider the language:

$$L = \{y \in \{0,1\}^{\ell(\lambda)} \mid \exists x \in \{0,1\}^\lambda \text{ s.t. } G(x) = y\} \ .$$

$\mathcal{DR}_{\mathsf{yes}}$ is constructed in the natural way (sample x and apply $G(x)$). From the pseudo-randomness property of the PRG we have that $M(\mathcal{DR}_{\mathsf{yes}})$ is indistinguishable from $\mathcal{U}_{\ell(\lambda)}$, and we can construct non-interactive arguments with accountable-soundness as long as $\ell(\lambda) = 2\lambda + \mathsf{poly}\log(\lambda)$.

Sequential Composition of Hash. Let $H : \{0,1\}^\lambda \to \{0,1\}^{\lambda/2}$ be an *unkeyed* hash function (e.g., SHA-256) where λ is sufficiently large even number. We emphasize that this is just a hash function and not a keyed hash family as typically required for complexity theoretic anaylsis. For some repetition parameter T, consider the following language:

$$L = \left\{x_{T+1} \in \{0,1\}^{\lambda/2} : \exists x_0 = (0^{\lfloor 7\lambda/8 \rfloor} \mid\mid y) \in \{0,1\}^\lambda \text{ s.t. } \forall i \in [T]\ x_i = H(0^{\lambda/2} || x_{i-1})\right\} \ .$$

In essence, a non-interactive argument of knowledge for L proves knowledge of a structured pre-image of a hash output obtained via repeated hashing. This language is a prominent benchmark for designing time- and space-efficient arguments for RAM computations [BHR+20, BHR+21], and also can be viewed as proving knowledge of a T-sized chain in a blockchain. Firstly, observe that L has $2^{\lceil \lambda/8 \rceil}$ instances, thereby it is δ-sparse for $\delta = 2^{-\lfloor 3\lambda/8 \rfloor}$. Secondly, for some specific hash function H, we can potentially conjecture that the distribution that generates y as above but for a uniformly random x_0 not of the form $(0^\lambda || *)$ is indistinguishable from the uniform distribution over L. Therefore, any adaptively sound SNARG for L can then be lifted to achieve accountable soundness w.r.t. the uniform distribution.

7.3 Construction for Negligibly Sparse Languages

The construction in Section 7.2 only covers a small subset of languages due to the required sparsity. In this section, we present a construction for δ-sparse languages for any negligible δ. However, this construction is worse than the above construction in two aspects: (a) the construction (in particular, Judge) depends on the distribution $\mathcal{D}$ for which accountability is to be shown, and (b) the defamation-freeness proof requires subexponential hardness assumptions. We note that the constructions presented in Section 7.1 as well as in this section do not suffer from these limitations: in particular, the construction is independent of the distribution $\mathcal{D}$ and the security proof only requires polynomial hardness from the building blocks.

Let L be any such δ-sparse NP language. Let $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be a non-interactive argument system for L. Let $\mathcal{D}'$ be some distribution over L using $t(\lambda)$ bits of randomness, for some appropriate t (that depends on δ). Towards the end of this section, we show how to realize such a $\mathcal{D}'$ from any distribution $\mathcal{D}$ additionally assuming subexponentially secure PRG.

We now give the construction $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ of a non-interactive argument system with accountable soundness:

Construction 7.6: NIZK with accountable soundness

$\mathsf{GenCRS}(1^\lambda)$:

1. Run $\mathsf{CRS}' = \mathsf{GenCRS}'(1^\lambda)$.
2. Sample $x^* \leftarrow \mathcal{U}_\ell$.
3. Output $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.

$\mathsf{Prove}(\mathsf{CRS}, x, w)$ where $\mathsf{CRS} = (\mathsf{CRS}', x^*)$:

1. Output $\pi = \mathsf{Prove}'(\mathsf{CRS}', x, w)$.

$\mathsf{Verify}(\mathsf{CRS}, x, \pi)$ where $\mathsf{CRS} = (\mathsf{CRS}', x^*)$:

1. Output $b = \mathsf{Verify}'(\mathsf{CRS}', x, \pi)$.

$\mathsf{Judge}\left(\mathsf{CRS} = (\mathsf{CRS}', x^*), \tau = (x', r, \pi)\right)$:

1. If CRS is not well-formed (i.e., $\mathsf{CRS} = (\mathsf{CRS}', x^*)$) then output $\mathsf{corrupted}$.
2. Otherwise, output $\mathsf{corrupted}$ iff $x' = \mathcal{D}(r)$, and $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus x', \pi) = 1$.

Theorem 7.7. *Let δ be some negligible function and let L be $\delta(\lambda)$-sparse language L over $\ell(\lambda)$-bit strings, and let $\mathcal{D}'$ be a distribution over L using $t(\lambda)$ bits of randomness. Then, if $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be a non-interactive adaptively sound argument for L, then the construction Π as described above achieves satisfies accountable soundness for L w.r.t. $\mathcal{D}'$ as long as $\mathcal{D}' \approx_c \mathcal{U}_\ell$ and $\delta \cdot 2^t$ is negligible in λ.*
Further, if Π' is a NIZK (resp., SNARG) then Π is a NIZK (resp., SNARG) with accountable soundness for the distribution $\mathcal{D}$.

Proof of Theorem 7.7. The completeness and soundness properties hold by the completeness and soundness of the non-interactive argument Π'. Moreover, GenCRS is identical to GenCRS' except that we add random string x^* of length ℓ in the CRS, and Prove and Verify algorithms ignore x^* and behave identically to Prove' and Verify'. Therefore, if Π' is a NIZK (resp., SNARG), we can conclude that so is Π.

Next, we proceed to show accountable soundness. Here, accountability follows identically to the proof of accountability in Theorem 7.4. We now proceed to discuss defamation-freeness.

Recall that to show defamation-freeness, we need to show that the no PPT adversary $\mathcal{A}$ when given a honestly generated CRS, can output a certificate $\tau = (x', r, \pi)$ that the Judge accepts with overwhelming probability. Recall that the Judge algorithm accepts τ iff π is an accepting proof for $x = x' \oplus x^*$ and that x' is chosen from the support of the distribution $\mathcal{D}'$.

Towards this, we first show that, over the choice of x^*, it is only with negligible probability that there exists any x' in the support of the distribution $\mathcal{D}'$ for which $x = x^* \oplus x'$ is even in L: By the δ-sparseness of L we have that for every x' in the support of $\mathcal{D}'$

$$\Pr_{x^* \in \mathcal{U}_\ell}[x^* \oplus x' \in L] \leq \delta \ . \tag{1}$$

Then, by a union bound over all 2^t values of x' in the support of $\mathcal{D}'$ we have that,

$$\Pr_{x^* \in \mathcal{U}_\ell}[\exists x' \in [\mathcal{D}'(\cdot)] : x^* \oplus x' \in L] \leq \delta \cdot 2^t \ , \tag{2}$$

which is negligible.

Therefore, for any $\mathcal{A}$, that breaks defamation-freeness with non-negligible probability p, with probability at least $p/2$, we have that $\mathcal{A}$ outputs a false statement $x = x' \oplus x^*$ along with an accepting proof for it. Then, we can build a reduction to the adaptive soundness of Π'. □

Existence of the distribution $\mathcal{D}'$. Next, we argue that given any distribution $\mathcal{D}$ using $\mathsf{poly}(\lambda)$ bits of randomness, we can construct a distribution $\mathcal{D}'$ using t bits of randomness by assuming sufficiently strong PRGs.

Claim 7.8. *Let $\delta(\lambda)$ be negligible function, L be a δ-sparse language L over ℓ bit strings, and $\mathcal{D}$ be a distribution over L using ℓ_r bits of randomness. Then, for $t = \log(1/\sqrt{\delta})$, assuming the existence of a PRG G from t bits to ℓ_r there exists a distribution $\mathcal{D}'$ over L that uses t bits of randomness. Further, $\mathcal{D} \approx_c \mathcal{U}_\ell$ implies $\mathcal{D}' \approx_c \mathcal{U}_\ell$.*

The construction of the distribution $\mathcal{D}'$ is straightforward: $\mathcal{D}'(r)$ outputs $\mathcal{D}(G(r))$. For inverse subexponential functions $\delta = 2^{-\lambda^\epsilon}$ for $0 < \epsilon < 1$, a polynomial stretch PRG would be sufficient (i.e., $t = (\lambda^\epsilon)/2$.). Such a PRG can be instantiated from any polynomially-secure PRG. However, for larger δ's (e.g., $\delta = 2^{-\log^2 \lambda}$) one would need a "super-polynomial" stretch PRG G that maps seeds of length $\mathsf{polylog}(\lambda)$ bits to strings of length $\mathsf{poly}(\lambda)$. Such a PRG can be instantiated from any sub-exponentially-secure PRG by appropriately scaling its security parameter. The indistinguishability of $\mathcal{D}'$ from $\mathcal{U}_\ell$ then follows readily from the pseudorandomness of the PRG as well as the fact that $\mathcal{D} \approx_c \mathcal{U}_\ell$.

7.4 General Feasibility for Accountable Soundness

So far in this section we presented compilers to lift any non-interactive argument to additionally satisfy accountable soundness. However, these compilers relied crucially on algebraic structures of the language L: in Section 7.1 required the ability to sample NO instances along with a witness where as in Section 7.2 the language was assumed to be sparse. In this section, we focus on understanding what cryptographic assumptions are sufficient to achieve accountable soundness for *any* NP language.

In particular, for any distribution $\mathcal{D}$, we present a construction of a non-interactive argument that achieves accountable soundness w.r.t. $\mathcal{D}$ under subexponential hardness assumptions. The description of the construction (more specifically, the Judge) algorithm depends on the distribution $\mathcal{D}$ and we require $\mathcal{D}$ to satisfy some natural but strong properties. We proceed to give the construction.

Construction. Let L be ***any*** language in NP on ℓ-bit strings and let $\mathcal{D}$ be a distribution over L using ℓ_r bits of randomness. Our non-interactive argument $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ depends on another non-interactive argument $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ for L and a PRG G from t bits to ℓ_r bits for parameter t to be specified shortly. The construction of Π is as follows:

Construction 7.9: NIZK with accountable soundness

$\mathsf{GenCRS}(1^\lambda)$:

1. Run $\mathsf{CRS}' = \mathsf{GenCRS}'(1^\lambda)$.
2. Sample $x^* \leftarrow \mathcal{U}_\ell$.

3. Output $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.

Prove(CRS, x, w) where $\mathsf{CRS} = (\mathsf{CRS}', x^*)$:

1. Output $\pi = \mathsf{Prove}'(\mathsf{CRS}', x, w)$.

Verify (CRS, x, π) where $\mathsf{CRS} = (\mathsf{CRS}', x^*)$:

1. Output $b = \mathsf{Verify}'(\mathsf{CRS}', x, \pi)$.

Judge $\big(\mathsf{CRS} = (\mathsf{CRS}', x^*), \tau = (x', r, \pi)\big)$:

1. If CRS is not well-formed (i.e., $\mathsf{CRS} = (\mathsf{CRS}', x^*)$) then output corrupted.
2. Otherwise, output corrupted iff $x' = \mathcal{D}(G(r))$, and $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus x', \pi) = 1$.

Theorem 7.10. *Let L be an NP language over ℓ bit strings. Let $\mathcal{D}$ be a distribution over L with the following two properties: (a) $\mathcal{D} \approx_c \mathcal{U}_\ell$, (b) there exists $0 < \epsilon < 1$ and a distribution $\mathcal{DR}_{\mathsf{no}}$ over $\bar{L}$ such that no PPT adversary can distinguish $\mathcal{DR}_{\mathsf{no}}$ from $\mathcal{U}_\ell$ with advantage better than $2^{-\lambda^\epsilon}$. Let $\Pi' = (\mathsf{GenCRS}', \mathsf{Prove}', \mathsf{Verify}')$ be a non-interactive argument for L for which no PPT adversary can break adaptive soundness with probability better than $2^{-\lambda^\epsilon}$ and G be a PRG. Then, the above construction Π is a non-interactive argument for L that satisfies accountable-soundness w.r.t. $\mathcal{D}$. Further, if Π' is a NIZK (resp., SNARG) then Π is a NIZK (resp., SNARG) with accountable soundness for the distribution $\mathcal{D}$.*

Proof. The completeness and soundness properties hold by the completeness and soundness of the non-interactive argument Π'. Moreover, GenCRS is identical to GenCRS' except that we add random string x^* of length ℓ in the CRS, and Prove and Verify algorithms ignore x^* and behave identically to Prove' and Verify'. Therefore, if Π' is a NIZK (resp., SNARG), we can conclude that so is Π.

Next, we show that Π satisfies accountable soundness w.r.t. $\mathcal{D}$. For this, we first show accountability and then proceed to show defamation-freeness.

Accountability. We exhibit the required extractor Ext below. Let $\mathcal{A}$ be some PPT adversary. The extractor has one black-box query to the adversary $\mathcal{A}$, after $\mathcal{A}$ gives out CRS^*.

1. The extractor is invoked on an input CRS^*. If CRS^* is not of the form of (CRS_Π, x^*) then output $\tau = \bot$.
2. Otherwise, it samples a random seed $r \leftarrow \mathcal{U}_{\ell_r}$ for the PRG, and uses $G(r)$ as the randomness to sample a statement x' from $\mathcal{D}$. That is, $x' = \mathcal{D}(G(r))$.
3. It queries $\mathcal{A}$ on $x = x^* \oplus x'$ to get back π, and outputs $\tau = (x', r, \pi)$.

We now show that $\mathsf{AccSnd.REAL}_{\Pi,\mathcal{A}}(\lambda)$, $\mathsf{AccSnd.IDEAL}_{\Pi,\mathcal{A},\mathcal{E}}(\lambda)$ are negligibly close. Recall that in AccSnd.REAL, $\mathcal{A}$ is queried on some random x sampled from $\mathcal{D}$ whereas in AccSnd.IDEAL the extractor samples some x' from $\mathcal{D}$ using randomness $G(r)$ and then queries $\mathcal{A}$ on the input $x = x' \oplus x^*$. To show accountability, we consider the following set of intermediate hybrids.

Hybrid $\boldsymbol{H_0(\lambda)}$: This is identical to the AccSnd.REAL game. More specifically,

1. Run $\mathcal{A}$ to get $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.

2. $x \leftarrow \mathcal{D}$.

3. Query $\mathcal{A}$ on x to get π.

4. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$.

Hybrid $H_1(\lambda)$: This is identical to H_0 except that $x \leftarrow \mathcal{U}_\ell$ instead of $x \leftarrow \mathcal{D}$.

Hybrid $H_2(\lambda)$: This is identical to H_1 except that $\mathcal{A}$ is queried on $x = x' \oplus x^*$ for $x' \leftarrow \mathcal{U}_\ell$.

Hybrid $H_3(\lambda)$: This is identical to H_2 except that $x' \leftarrow \mathcal{D}$ instead of $x' \leftarrow \mathcal{U}_\ell$.

Hybrid $H_4(\lambda)$: This is identical to H_3 except that x' is computed as $\mathcal{D}(G(r))$ for a uniformly random string r instead of $x' \leftarrow \mathcal{D}$.

Hybrid $H_5(\lambda)$: This is identical to H_4 except that the winning condition is changed. In particular, this hybrid outputs 1 iff $\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$ and $x' = \mathcal{D}(G(r))$. This hybrid is identical to the $\mathsf{AccSnd.IDEAL}$ game. We explicitly write down this hybrid for clarity.

1. Run $\mathcal{A}$ to get $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.

2. Set $x' = \mathcal{D}(G(r))$ for $r \leftarrow \mathcal{U}_{\ell_r}$.

3. Query $\mathcal{A}$ on $x = x^* \oplus x'$ to get back π.

4. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$ and $x' = \mathcal{D}(G(r))$.

For H_i, let p_i be the probability that H_i outputs 1. We need to show that there exists a negligible function μ such that $| p_0 - p_5 | \leq \mu(\lambda)$.

First, note that there exists a negligible function μ_{03} such that $| p_0 - p_3 | \leq \mu_{03}(\lambda)$. This follows from the indistinguishability of $\mathcal{D}$ and $\mathcal{U}_\ell$: p_0 and p_1 are negligibly close due to $\mathcal{D} \approx_c \mathcal{U}_\ell$. The hybrids Hyb_0 and Hyb_1 are identical and hence we conclude that p_1 and p_2 are equal, and finally p_2 and p_3 are negligible close due $\mathcal{D} \approx_c \mathcal{U}_\ell$.

Secondly, the only difference between H_3 and H_4 is that in the former x' is sampled from $\mathcal{D}$ using a uniformly random ℓ_r bit string as the randomness whereas in the latter x' is sampled from $\mathcal{D}$ using $G(r)$ as a the randomness for a uniformly random t bit string r. Therefore, by the security of the PRG G, there exists a negligible function μ_{34} such that $| p_3 - p_4 | \leq \mu_{34}(\lambda)$.

Finally, the only difference between H_4 and H_5 is the condition on which the hybrids output 1. Specifically, H_4 outputs 1 only if $\mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1$ whereas H_5 additionally requires that $x' = \mathcal{D}(G(r))$ when outputting 1. However, note that the view of $\mathcal{A}$ is identical across both experiments and the additional condition checked in H_5 is always satisfied in H_5. Therefore, we can conclude that p_4 and p_5 are identical. This concludes the proof of accountability.

Defamation-freeness. We need to show that for a honestly generated $\mathsf{CRS} = (\mathsf{CRS}', x^*)$, no PPT adversary $\mathcal{A}$ can output a certificate $\tau = (x', r, \pi)$ that the Judge algorithm accepts with non-negligible probability. To show this we consider the following set of hybrids and highlight the changes across hybrids in red.

Hybrid $H_0(\lambda)$: This hybrid is identical to the defamation-freeness game for Π. More specifically,

1. $\mathsf{CRS}' \leftarrow \mathsf{GenCRS}'(\lambda)$, $x^* \leftarrow \mathcal{U}_\ell$.

2. Set $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.

3. Run $\mathcal{A}$ on input CRS to get back $\tau = (x', r, \pi)$.
4. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus, x', \pi) = 1$ and $x' = \mathcal{D}(G(r))$.

Hybrid $H_1(\lambda)$: This hybrid is identical to H_0 except that the hybrid samples a guess s for the randomness r output by $\mathcal{A}$, and outputs 1 only if its guess for r is correct. More specifically,

1. $\mathsf{CRS}' \leftarrow \mathsf{GenCRS}'(\lambda)$, $x^* \leftarrow \mathcal{U}_\ell$.
2. Set $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.
3. Compute $s \leftarrow \mathcal{U}_{\ell_r}$.
4. Run $\mathcal{A}$ on input CRS to get back $\tau = (x', r, \pi)$.
5. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus, x', \pi) = 1$ and $x' = \mathcal{D}(G(r))$ and $s = r$.

Hybrid $H_2(\lambda)$: This hybrid is identical to H_1 except that sample x^* from a syntactically different distribution. More specifically,

1. $\mathsf{CRS}' \leftarrow \mathsf{GenCRS}'(\lambda)$.
2. $\tilde{x} \leftarrow \mathcal{U}_\ell$.
3. $s \leftarrow \mathcal{U}_{\ell_r}$, $y = \mathcal{D}(G(s))$.
4. Set $x^* = \tilde{x} \oplus y$.
5. Set $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.
6. Run $\mathcal{A}$ on input CRS to get back $\tau = (x', r, \pi)$.
7. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus, x', \pi) = 1$ and $x' = \mathcal{D}(G(r))$ and $s = r$.

Hybrid $H_3(\lambda)$: This hybrid is identical to H_2 except that $\tilde{x}$ is sampled from $\mathcal{DR}_{\mathsf{no}}$ distirbution instead of $\mathcal{U}_\ell$. More specifically,

1. $\mathsf{CRS}' \leftarrow \mathsf{GenCRS}'(\lambda)$.
2. $\tilde{x} \leftarrow \mathcal{DR}_{\mathsf{no}}$.
3. $s \leftarrow \mathcal{U}_{\ell_r}$, $y = \mathcal{D}(G(s))$.
4. Set $x^* = \tilde{x} \oplus y$.
5. Set $\mathsf{CRS} = (\mathsf{CRS}', x^*)$.
6. Run $\mathcal{A}$ on input CRS to get back $\tau = (x', r, \pi)$.
7. Output 1 iff $\mathsf{Verify}(\mathsf{CRS}, x^* \oplus, x', \pi) = 1$ and $x' = \mathcal{D}(G(r))$ and $s = r$.

For each H_i, let p_i be the probability that H_i outputs 1. We want to show that p_0 is negligible for all $\mathcal{A}$. Towards this, first note that $p_1 = p_0/2^t$. The hybrids H_2 and H_1 are identical, hence $p_2 = p_1$.

Claim 7.11. *By the subexponential indistinguishability of $\mathcal{U}_\ell$ and $\mathcal{DR}_{\mathsf{no}}$, we have that* $\mid p_2 - p_3 \mid \leq 2^{-\lambda^\epsilon}$.

The only difference between H_2 and H_3 is that in H_2 $\tilde{x}$ is sampled from $\mathcal{U}_\ell$ whereas in H_3 it is sampled from $\mathcal{DR}_{\mathsf{no}}$. The claim follows.

We conclude the proof by upperbounding the probability p_3 relying on the adaptive soundness of Π'. At a high level, since $\tilde{x}$ is sampled from $\mathcal{DR}_{\mathsf{no}}$, if H_3 outputs 1 with probability p_3 this implies that $\mathcal{A}$ was able to come up with an accepting proof π for the statement $x^* \oplus x'$ and that $s = r$. Combining these, we can conclude that $x' = y$ and furthermore $x^* \oplus x' = \tilde{x}$ which is a NO instance. Therefore, we can reduce to the adaptive soundness of Π'.

Claim 7.12. *By adaptive soundness of Π', we have that $p_3 \leq 2^{-\lambda^\epsilon}$.*

Therefore, by combining the above claims, we have that $p_0 \leq 2 \cdot 2^{-\lambda^\epsilon} \cdot 2^t$ which is negligible for $t = \lambda^\delta$ for $0 < \delta < \epsilon$. □

8 Combining Subversion Advice-ZK and Accountable Soundness

In this section, we build NIZKs that satisfy both subversion advice-ZK and accountable soundness. This results in the first NIZKs that satisfy meaningful notions of privacy as well as soundness for malicious CRS.

For languages that fit $\mathsf{NP} \cap \mathsf{co\text{-}NP}$'s generalization defined in Section 7.1, we start by plugging in an adaptively sound NIZK argument Π' for NP in Theorem 5.3 to get a NIZK argument Π that satisfies subversion advice-ZK. Then, instantiate Theorem 7.2 with such a Π to get a subversion advice-ZK NIZK $\widetilde{\Pi}$ with accountable soundness: the compiler in Theorem 7.2 preserves subversion advice-ZK property as it doesn't change Π's $\mathsf{Prove}, \mathsf{Verify}, \mathsf{GenCRS}$ algorithms.

Theorem 8.1. *Let L be **any** NP language, R_L be its relation. For $L_{\mathsf{No}} \subseteq \bar{L}$, let R_{No} be its relation. For some super-polynomial function T, assume an adaptively sound NIZK for NP, a non-interactive perfectly binding T-extractable commitment Com, a T-hard trapdoor generation protocol, and a two-message delayed input, publicly verifiable WI argument for NP with T-adaptive soundness.*
Then, there exists a subversion advice-ZK NIZK argument for L that satisfies accountable soundness w.r.t. distribution $\mathcal{D}$ as required in Theorem 7.2.

Similarly, we obtain a subversion advice-ZK NIZK with accountable soundness for sparse languages from Theorem 7.4 and Theorem 5.3.

Theorem 8.2. *Let L be some δ-sparse NP language over ℓ-bit strings such that $\delta^2 \cdot 2^\ell$ is negligible. For some super-polynomial function T, assume an adaptively sound NIZK for NP, a non-interactive perfectly binding T-extractable commitment Com, a T-hard trapdoor generation protocol, and a two-message delayed input, publicly verifiable WI argument for NP with T-adaptive soundness.*
Then, there exists a subversion advice-ZK NIZK argument for L that satisfies accountable soundness w.r.t. distribution $\mathcal{D}$ as required in Theorem 7.4.

Acknowledgements. This work was done partially when Pratik Soni was visiting Carnegie Mellon University, where he was supported by a DARPA SIEVE grant, and an ACE center award from the Algorand Foundation. Gilad Asharov and Hadar Kaner were supported by Israel Science Foundation (grant No. 2439/20), JP Morgan Faculty Research Award, and European Union's Horizon

2020 research and innovation programme under the Marie SklodowskaCurie grant agreement No. 891234.

References

[AADG21] Prabhanjan Ananth, Gilad Asharov, Hila Dahari, and Vipul Goyal. Towards accountability in CRS generation. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021*, volume 12698, pages 278–308. Springer, 2021.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 99–108, New York, NY, USA, 1996. Association for Computing Machinery.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334, 2018.

[BCPR14] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the existence of extractable one-way functions. Cryptology ePrint Archive, Paper 2014/402, 2014. https://eprint.iacr.org/2014/402.

[BFJ+20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical zap arguments. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 642–667, Cham, 2020. Springer International Publishing.

[BFS16] Mihir Bellare, Georg Fuchsbauer, and Alessandra Scafuro. Nizks with an untrusted CRS: security in the face of parameter subversion. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016*, volume 10032 of *Lecture Notes in Computer Science*, pages 777–804, 2016.

[BHR+20] Alexander R. Block, Justin Holmgren, Alon Rosen, Ron D. Rothblum, and Pratik Soni. Public-coin zero-knowledge arguments with (almost) minimal time and space overheads. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography*, pages 168–197, Cham, 2020. Springer International Publishing.

[BHR+21] Alexander R. Block, Justin Holmgren, Alon Rosen, Ron D. Rothblum, and Pratik Soni. Time- and space-efficient arguments from groups of unknown order. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 123–152, Cham, 2021. Springer International Publishing.

[BP04] Boaz Barak and Rafael Pass. On the possibility of one-message weak zero-knowledge. In Moni Naor, editor, *Theory of Cryptography*, pages 121–132, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[BSCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. Cryptology ePrint Archive, Paper 2014/349, 2014.

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 573–592, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: From practice to theory. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 1082–1090, New York, NY, USA, 2019. Association for Computing Machinery.

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270. Springer, 1994.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, pages 255–271, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[DN07] Cynthia Dwork and Moni Naor. Zaps and their applications. *SIAM J. Comput.*, 36(6):1513–1543, feb 2007.

[FLS90] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 308–317 vol.1, 1990.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 416–426, 1990.

[GGJS11] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Bringing people of different beliefs together to do uc. In Yuval Ishai, editor, *Theory of Cryptography*, pages 311–328, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. A generic approach to constructing and proving verifiable random functions. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography*, pages 537–566, Cham, 2017. Springer International Publishing.

[GKM+19] Rishab Goyal, Sam Kim, Nathan Manohar, Brent Waters, and David J. Wu. Watermarking public-key cryptographic primitives. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 367–398, Cham, 2019. Springer International Publishing.

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 660–670, New York, NY, USA, 2018. Association for Computing Machinery.

[GKWW21] Rishab Goyal, Sam Kim, Brent Waters, and David J. Wu. Beyond software watermarking: Traitor-tracing for pseudorandom functions. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 250–280, Cham, 2021. Springer International Publishing.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, New York, NY, USA, 1987. Association for Computing Machinery.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *J. Cryptol.*, 7(1):1–32, dec 1994.

[GO07] Jens Groth and Rafail Ostrovsky. Cryptography in the multi-string model. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007*, pages 323–341, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for nizk. In *Proceedings of the 26th Annual International Conference on Advances in Cryptology*, CRYPTO'06, page 97–111, Berlin, Heidelberg, 2006. Springer-Verlag.

[Goy07] Vipul Goyal. Reducing trust in the PKG in identity based cryptosystems. In *Advances in Cryptology - CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 430–447. Springer, 2007.

[KZ20] Benjamin Kuykendall and Mark Zhandry. Towards non-interactive witness hiding. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020*, volume 12550 of *Lecture Notes in Computer Science*, pages 627–656. Springer, 2020.

[LVW19] Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. 2-message publicly verifiable wi from (subexponential) lwe. Cryptology ePrint Archive, Paper 2019/808, 2019.

[NWZ15] Ryo Nishimaki, Daniel Wichs, and Mark Zhandry. Anonymous traitor tracing: How to embed arbitrary information in a key. Cryptology ePrint Archive, Paper 2015/750, 2015.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *Proceedings of the 22nd International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'03, page 160–176, Berlin, Heidelberg, 2003. Springer-Verlag.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for np from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 89–114, Cham, 2019. Springer International Publishing.

[TBM+20] Sri Aravinda Krishnan Thyagarajan, Adithya Bhat, Giulio Malavolta, Nico Döttling, Aniket Kate, and Dominique Schröder. Verifiable timed signatures made practical. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, page 1733–1750, 2020.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

[ZKC] Zero-knowledge contingency payment. accessed: February 2023.

A Instantiations of Trapdoor Generation

A.1 Trapdoor Generation from One-Way Functions with Efficient Recognizable Range

Let $f = \{f_\lambda : \mathcal{D}_\lambda \to \mathcal{R}_\lambda\}_\lambda$ be a T-one-way function. Further we assume that f has an efficient recognizable range, that is, we assume the existence of a PPT algorithm $\mathsf{CheckRange}$ that for all λ and $\mathsf{s}_{\mathsf{out}} \in \mathcal{R}_\lambda$:

$$\mathsf{CheckRange}(\mathsf{s}_{\mathsf{out}}) = 1 \quad \iff \quad \exists \mathsf{s}_{\mathsf{in}} \in \mathcal{D}_\lambda \quad \text{s.t.} \quad f_\lambda(\mathsf{s}_{\mathsf{in}}) = \mathsf{s}_{\mathsf{out}} \ .$$

We give a construction of the trapdoor generation protocol for such a one-way function:

1. $\mathsf{TDGen}(1^\lambda)$ samples a random $\mathsf{s}_{\mathsf{in}} \leftarrow \mathcal{D}_\lambda$ and output $\mathsf{s}_{\mathsf{out}} = f_\lambda(\mathsf{s}_{\mathsf{in}})$.
2. $\mathsf{TDValid}(\mathsf{s}_{\mathsf{out}})$ outputs 1 iff $\mathsf{CheckRange}(\mathsf{s}_{\mathsf{out}}) = 1$.
3. $\mathsf{TDCheck}(1^\lambda, \mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}})$ outputs 1 iff $\mathsf{s}_{\mathsf{out}} = f_\lambda(\mathsf{s}_{\mathsf{in}})$.
4. TDSol on input $\mathsf{s}_{\mathsf{out}} \in \mathcal{R}_\lambda$ finds an $\mathsf{s}_{\mathsf{in}} \in \mathcal{D}_\lambda$ such that $f_\lambda(\mathsf{s}_{\mathsf{in}}) = \mathsf{s}_{\mathsf{out}}$ via brute-force search.

Clearly, $\mathsf{TDValid}(\mathsf{TDGen}(1^\lambda)) = 1$. Moreover, for every $\mathsf{s}_{\mathsf{out}}$ we have that $\mathsf{TDValid}(\mathsf{s}_{\mathsf{out}}) = 1$ if and only if there exists an s_{in} such that $\mathsf{TDCheck}(\mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 1$, i.e., $\mathsf{s}_{\mathsf{out}} = f_\lambda(\mathsf{s}_{\mathsf{in}})$. T-hardness follows from the T-one-wayness of f. Finally, the correctness of TDSol is immediate.

Instantiating T-hard one-way function: For any T, a T-hard OWF g can be obtained from 2^{λ^ϵ}-secure OWF f by setting $\lambda_f = \omega(1) \cdot T/\epsilon$ where λ_f, λ are security parameters of f and g respectively. Then, T-hardness of g follows from the fact that for any polynomial p, $p(T) < 2^{\lambda_f^\epsilon}$. Additionally, g satisfies S-solvability for $S = 2^{\lambda_f} = 2^{\omega(1)T^{1/\epsilon}}$.

A.2 Trapdoor Generation from Collision-Resistant Hash Family

Let $H = \{h_\lambda : \{0,1\}^{\kappa(\lambda)} \times \mathcal{D}_\lambda \to \mathcal{R}_\lambda\}$ be a collision-resistant hash family, where for every $\lambda \in \mathbb{N}$ we have $|\mathcal{R}_\lambda| < |\mathcal{D}_\lambda|$. Then consider the following construction of trapdoor generation:

1. $\mathsf{TDGen}(1^\lambda)$ outputs a random hash function key $\mathsf{s}_{\mathsf{out}} \leftarrow \{0,1\}^{\kappa(\lambda)}$.
2. $\mathsf{TDValid}(\mathsf{s}_{\mathsf{out}})$ outputs 1 iff $\mathsf{s}_{\mathsf{out}} \in \{0,1\}^{\kappa(\lambda)}$.
3. $\mathsf{TDCheck}(1^\lambda, \mathsf{s}_{\mathsf{in}} = (x_0, x_1), \mathsf{s}_{\mathsf{out}})$ outputs 1 iff $h_\lambda(\mathsf{s}_{\mathsf{out}}, x_0) = h_\lambda(\mathsf{s}_{\mathsf{out}}, x_1)$ and $x_0 \neq x_1$.
4. TDSol on input $\mathsf{s}_{\mathsf{out}} \in \mathcal{R}_\lambda$ finds an $\mathsf{s}_{\mathsf{in}} \in \mathcal{D}_\lambda^2$ such that $\mathsf{TDCheck}(1^\lambda, \mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 1$ via brute-force search.

Clearly, $\mathsf{TDValid}(\mathsf{TDGen}(1^\lambda)) = 1$. Further, since h is compressing, for every $\mathsf{s}_{\mathsf{out}}$ there exists $\mathsf{s}_{\mathsf{in}} = (x_0, x_1)$ such that $\mathsf{TDCheck}(\mathsf{s}_{\mathsf{in}}, \mathsf{s}_{\mathsf{out}}) = 1$ and $x_0 \neq x_1$. Finally, T-hardness follows from the T-collision-resistance of H and the correctness of TDSol is immediate.

Instantiating T-collision resistant hash function: As described in the above subsection, for any T, a T-collision resistant hash function g can be obtained from 2^{λ^ϵ}-collision resistant hash function f by setting $\lambda_f = \omega(1) \cdot T/\epsilon$ where λ_f, λ are security parameters of f and g respectively. Then, T-hardness of g follows from the fact that for any polynomial p, $p(T) < 2^{\lambda_f^\epsilon}$.

B Subversion Advice-ZK NIZKs satisfy Subversion Witness Hiding, Subversion Function Hiding and More

In this section, we discuss the relation of our new notion of subversion advice-ZK for NIZKs with several different notions of privacy in the subversion-setting. In particular, we show that it implies the previously considered notion of subversion-witness-indistinguishability. We also introduce other relaxed notions including subversion witness-hiding and subversion-function-hiding, and show that subversion advice-ZK imply them all.

B.1 Subversion Witness Hiding (WH)

Informally, Subversion WH demands that even when the authority creates the CRS maliciously, it still cannot recover a valid witness for the instance from the proof it was given. The adversary is modeled as a two-stage algorithm: it first outputs a CRS CRS^* and a secret state τ passed to the second stage. The second stage is then defined like the honest-CRS WH game. We emphasize that WH is meaningful only for languages that are considered "hard" according to some distribution; that is, it is infeasible to find a valid witness for the instance, when it is drawn from the distribution. We first formally define "hard" distributions.

Definition B.1. *Let L be an NP language and R_L be its associated relation, let $\mathcal{D}$ be a distribution over instance-witness pairs of R_L. We say that L is* hard w.r.t. distribution $\mathcal{D}$ *if for every non-uniform* PPT *algorithm $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that for every auxiliary input $z \in \{0,1\}^*$:*

$$\Pr\left[(x,w') \in R_L : \ (x,w) \leftarrow \mathcal{D},\ w' \leftarrow \mathcal{A}(x,z)\right] \leq \mu(\lambda) \ .$$

Next, we formally define subversion-witness-hiding for NIZKs.

Definition B.2 (Witness-Hiding with Subversion CRS)**.** *Let $L \in \mathsf{NP}$ and a distribution $\mathcal{D}$ over its associated relation R_L, L is hard w.r.t. $\mathcal{D}$. We say that a NIZK argument for $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ for L is also* witness hiding with subversion CRS with respect to $\mathcal{D}$ *if for all non-uniform* PPT *adversaries $\mathcal{A}$ there exists a negligible function μ such that:*

$$\Pr[\mathsf{S\text{-}WH}_{\Pi,\mathcal{A},\mathcal{D}}(\lambda) = 1] \leq \mu(\lambda)$$

where the random variable $\mathsf{S\text{-}WH}_{\Pi,\mathcal{A},\mathcal{D}}(\lambda)$ is defined as follows:

$\underline{\mathsf{S\text{-}WH}_{\Pi,\mathcal{A},\mathcal{D}}(\lambda)}$*:*

1. *The adversary $\mathcal{A}(1^\lambda)$ outputs (subverted) CRS^* and secret state τ.*
2. *$(x,w) \leftarrow \mathcal{D}$ and then $\pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w)$.*
3. *$\mathcal{A}$ is given (x,π).*
4. *$\mathcal{A}$ outputs w'. The output of the experiment is 1 if $R_L(x,w') = 1$.*

Theorem B.3. *An subversion advice-ZK NIZK system $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ is also subversion witness hiding.*

Proof. Assume there exists a non-uniform PPT adversary $\mathcal{A}$ that wins in $\mathsf{S\text{-}WH}_{\Pi,\mathcal{A},\mathcal{D}}(\lambda)$ with some non-negligible probability $\epsilon(\lambda)$. That is, $\mathcal{A}$ does the following:

1. $\mathcal{A}$ outputs (CRS^*, τ).
2. On input (x, π) (where π is a valid proof for x), $\mathcal{A}$ outputs w'.

Consider the following adversary $\mathcal{B}$ for the subversion advice-ZK game of Π that depends on $\mathcal{A}$: It runs $\mathcal{A}(1^\lambda)$ and gets (CRS^*, τ). It outputs CRS^* as it's first message. It samples $(x, w) \leftarrow \mathcal{D}$ and queries on that instance-witness pair, and then receives back π. It sends (x, π) to $\mathcal{A}$ in order to obtain w' and outputs whatever it outputs.

Since Π is an subversion advice-ZK NIZK, there exists a non-uniform PPT simulator $\mathcal{S}$ and an advice distribution $\mathcal{D}_\mathcal{S}$ for which $\mathsf{REAL}_{\mathcal{B},\Pi}(\lambda)$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}_\mathcal{S}}(\lambda)$ are computationally indistinguishable.

By the subversion advice-ZK NIZK property of Π, we can claim that $\mathcal{B}$ outputs a valid witness for x with non-negligible probability. However, in order to reduce to the hardness of distribution $\mathcal{D}$, we need to somehow provide d as non-uniform advice to $\mathcal{B}$. This can actually be done via a standard averaging argument. Specifically, there exist some d in the support of $\mathcal{D}_\mathcal{S}$ such that conditioned on d being given to $\mathcal{B}$, it outputs a valid witness with non-negligible probability. Let us fix such a d. Then, $\mathcal{B}$ with such a d hardwired as non-uniform advice contradicts the hardness of the underlying language.

□

B.2 Subversion Witness Indistinguishability (WI)

Subversion WI demands that even when the authority creates the CRS maliciously, it still cannot decide which of two witnesses of its choice were used to create a proof. The adversary is modeled as a two-stage algorithm: it first outputs a CRS CRS^* and a secret state τ passed to the second stage. The second stage is then defined like the honest-CRS WI game.

Definition B.4. *Let $L \in \mathsf{NP}$ and a distribution $\mathcal{D}$ over its associated relation R_L, L is hard w.r.t. $\mathcal{D}$. We say that a NIZK argument $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ for L is also* witness indistinguishable for subversion CRS *if for all non-uniform PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ such that for all $x \in L$, witnesses w_0, w_1 of x:*

$$\Pr[\mathsf{S\text{-}WI}_{\Pi,\mathcal{A}}(\lambda, x, w_0, w_1) = 1] \leq \frac{1}{2} + \mu(\lambda)$$

where the random variable $\mathsf{S\text{-}WI}_{\Pi,\mathcal{A},\mathcal{D}}(\lambda, x, w_0, w_1)$ is defined as follows:

$\underline{\mathsf{S\text{-}WI}_{\Pi,\mathcal{A}}(\lambda, x, w_0, w_1)}$*:*

1. *If $(x, w_0) \notin R_L$ or $(x, w_1) \notin R_L$ then abort and output $\perp$.*
2. *The adversary $\mathcal{A}(1^\lambda)$ outputs (subverted) CRS^* and a secret state τ.*
3. *Sample $b \leftarrow \{0,1\}$ uniformly at random. Compute $\pi = \mathsf{Prove}(\mathsf{CRS}^*, x, w_b)$ and send (x, π) to $\mathcal{A}$.*
4. *$\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. The output of the experiment is 1 iff $b = b'$.*

Theorem B.5. *An subversion advice-ZK NIZK system $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ satisfies subversion witness indistinguishability.*

Proof. Let $\mathcal{A}$ be a non-uniform PPT adversary for the S-WI game. We show that $\mathcal{A}$ can be used to construct a non-uniform PPT adversary $\mathcal{A}'$ for the REAL game of the subversion advice-ZK of Π: On input 1^λ, $\mathcal{A}'$ runs $\mathcal{A}(1^\lambda)$ to obtain (CRS^*, τ). It outputs CRS^*. Eventually, $\mathcal{A}'$ receives (x, π) as an input. It runs $\mathcal{A}(x, \pi)$ and outputs its output b'.

$\mathcal{A}'$ is non-uniform and PPT since $\mathcal{A}$ is non-uniform and PPT. By the subversion advice-ZK property of Π, there exist a non-uniform PPT simulator $\mathcal{S}$ and an advice distribution $\mathcal{D}$ for which $\mathsf{REAL}_{\mathcal{A}',\Pi}(\lambda, x, w_b)$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda, x, w_b)$ are indistinguishable, where $b \in \{0,1\}$ is the uniform bit chosen in the game S-WI. Since the simulator's view is independent from the used witness, we also have that $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda, x, w_0)$ and $\mathsf{IDEAL}_{\mathcal{S},\Pi,\mathcal{D}}(\lambda, x, w_1)$ are indistinguishable. Thus, from transitivity of indistinguishability, we have that $\mathsf{REAL}_{\mathcal{A}',\Pi}(\lambda, x, w_0)$ and $\mathsf{REAL}_{\mathcal{A}',\Pi}(\lambda, x, w_1)$ are indistinguishable. Therefore, there exists a negligible function $\mu(\cdot)$ such that:

$$\Pr\left[b' = 0 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_0), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right] - \Pr\left[b' = 0 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_1), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right] \leq \mu(\lambda)$$

Overall, we have that:

$$\begin{aligned}
&\Pr\left[\mathsf{S\text{-}WI}_{\Pi,\mathcal{A}}(\lambda, x, w_0, w_1) = 1\right] = \\
&\quad \Pr[b' = b :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_b), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda), b \leftarrow \{0,1\}] \\
&\quad = \frac{1}{2} \cdot \Pr\left[b' = 0 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_0), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right] \\
&\quad + \frac{1}{2} \cdot \Pr\left[b' = 1 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_1), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right] \\
&\quad = \frac{1}{2} \cdot \Pr\left[b' = 0 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_0), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right] \\
&\quad + \frac{1}{2} \cdot \left(1 - \Pr\left[b' = 0 :\ b' \leftarrow \mathcal{A}(x,\pi), \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, x, w_1), \mathsf{CRS}^* \leftarrow \mathcal{A}(1^\lambda)\right]\right) \\
&\quad \leq \frac{1}{2} + \frac{1}{2} \cdot \mu(\lambda)
\end{aligned}$$

□

B.3 Subversion Function Hiding

We introduce a new notion of subversion-security that we call subversion function hiding. Intuitively, a NIZK proof system satisfies subversion function-hiding if no efficient adversary can compute any efficiently computable function $f(x, w)$ of *the prover's witness* when given an honestly generated NIZK proof for the statement x even using a maliciously sampled CRS. We emphasize that we are only interested in hiding functions of the prover's witness. While an extension to hide functions of all witnesses is interesting, it is not clear how such a definition should look like. We leave it as an open question for the future.

Definition B.6. *Let L be some* NP *language, $\mathcal{D}$ be a distribution over instance-witness pairs of L. We say that a NIZK* $(\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ *for L is subversion function-hiding if for all non-uniform* PPT *verifiers $\mathsf{V} = (\mathsf{V}_0, \mathsf{V}_1)$ there exists a non-uniform polynomial-time simulator $\mathcal{S}$ such that for all efficiently computable functions $f(\cdot)$*

if there exists a non-negligible function ϵ such that

$$\Pr[\mathsf{REAL}_{\mathsf{pred}}(\lambda, \mathcal{S}, f, \mathcal{D}) = 1] \geq \epsilon(\lambda) \ ,$$

then there exists a non-negligible function δ such that

$$\Pr[\mathsf{IDEAL}_{\mathsf{pred}}(\lambda, \mathsf{V}, f, \mathcal{D}) = 1] \geq \delta(\lambda) \ ,$$

where the games IDEAL *and* REAL *are defined below:*

$\underline{\mathsf{REAL}_{\mathsf{pred}}(\lambda, \mathsf{V}, f, \mathcal{D}):}$

1. $(\mathsf{CRS}, \tau) \leftarrow \mathsf{V}_0$.
2. $(x, w) \leftarrow \mathcal{D}(1^\lambda)$.
3. $\pi \leftarrow \mathsf{Prove}(\mathsf{CRS}, x, w)$.
4. $y_{\mathsf{real}} \leftarrow \mathsf{V}_1(\mathsf{CRS}, \tau, x, \pi)$.
5. *Output* 1 *iff* $y_{\mathsf{real}} = f(x, w)$.

$\underline{\mathsf{IDEAL}_{\mathsf{pred}}(\lambda, \mathcal{S}, f, \mathcal{D}):}$

1. $(x, w) \leftarrow \mathcal{D}(1^\lambda)$.
2. $y_{\mathsf{ideal}} \leftarrow \mathcal{S}(x)$.
3. *Output* 1 *iff* $y_{\mathsf{ideal}} = f(x, w)$.

Theorem B.7. *An subversion advice-ZK NIZK* $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ *satisfies subversion function hiding.*

Proof. The proof is very similar to the case of subversion witness-hiding but we include it for completeness. Let $\mathsf{V} = (\mathsf{V}_0, \mathsf{V}_1)$ be a non-uniform PPT corrupted verifier for the subversion function-hiding game. We show that V can be used to construct a non-uniform PPT adversary $\mathcal{A}$ for the subversion advice-ZK game of Π: $\mathcal{A}$ is given 1^λ as an input. It runs $\mathsf{V}_0(1^\lambda)$ and obtains its output (CRS^*, τ). It outputs CRS^*, and then being invoked on (x, π). It runs $\mathsf{V}_1(\mathsf{CRS}^*, \tau, x, \pi)$ and obtains y_{real}.

Since V is non-uniform and PPT, $\mathcal{A}$ is also non-uniform and PPT. From the subversion advice-ZK property of Π, there exists a PPT simulator $\mathcal{S}_{\mathcal{A}}$ and an advice distribution $\mathcal{D}_{\mathcal{A}}$ such that $\mathsf{REAL}_{\mathcal{A},\Pi}(\lambda, x, w)$ and $\mathsf{IDEAL}_{\mathcal{S}_{\mathcal{A}},\Pi,\mathcal{D}_{\mathcal{A}}}(\lambda, x, w)$ are indistinguishable.

Consider the following adversary $\mathcal{S}$ that depends on $\mathcal{A}$: It gets as input $x \leftarrow \mathcal{D}$ and a sample $d = (r, \mathsf{td})$ from $\mathcal{D}_{\mathcal{A}}$. It first sets $\mathsf{CRS}^* = \mathcal{A}(r)$. It then runs the $\mathcal{S}_{\mathcal{A}}$ to computes a simulated proof π' using the trapdoor td of CRS. It sends $\mathcal{A}$ on (x, π') to receive y, and outputs whatever it outputs.

By the subversion advice-ZK NIZK property of Π, we can claim that with non-negligible probability, $\mathcal{S}$'s output y is indeed $f(x, w)$. However, notice that $\mathcal{S}$ additionally receives a sample from the inefficient distribution $\mathcal{D}_{\mathcal{A}}$. To conclude the construction of the required simulator, we will have to fix some "good" advice d to $\mathcal{S}$ as non-uniform advice. This can actually be done via a standard averaging argument. Specifically, there exist some d in the support of $\mathcal{D}_{\mathcal{A}}$ such that conditioned on d being given to $\mathcal{S}$, it's output y is $f(x, w)$ with non-negligible probability. Let us fix such a d. Then, $\mathcal{S}$ with such a d hardwired as non-uniform advice gives us the required simulator to show subversion function-hiding.

□