New Techniques for Non-interactive Zero-Knowledge

Jens Groth* Rafail Ostrovsky† Amit Sahai‡

UCLA Computer Science Department
4732 Boelter Hall
Los Angeles, CA 90095-1596, USA
{jg,rafail,sahai}@cs.ucla.edu

Abstract

Non-interactive zero-knowledge (NIZK) proof systems are fundamental primitives used in many cryptographic constructions, including CCA2-secure cryptosystems, digital signatures, and various cryptographic protocols. We introduce new techniques for constructing NIZK proofs based on groups with a bilinear map. In comparison with previous constructions of NIZK proofs, our techniques yield significant reductions in the length of the common reference string and the size of the proofs. The new techniques also allow us to answer long standing open questions in the theory of non-interactive zero-knowledge.

- We construct the first perfect NIZK argument system for all NP languages.
- We construct the first UC-secure NIZK argument for all NP languages in the presence of an adaptive adversary.
- We construct the first non-interactive zap for all NP languages based on a standard cryptographic security assumption.

Keywords: Non-interactive zero-knowledge, witness indistinguishability, universal composability, groups with bilinear map, decision subgroup assumption, decisional linear assumption.

*Supported by NSF ITR/Cybertrust grant No. 0456717.

†Supported in part by a gift from Teradata, Intel equipment grant, NSF Cybertrust grant No. 0430254, OKAWA research award, B. John Garrick Foundation and Xerox Innovation Group Award.

‡Supported by grant No. 0456717 from the NSF ITR and Cybertrust programs, an equipment grant from Intel, and an Alfred P. Sloan Foundation Research Fellowship.

1 Introduction

Non-interactive zero-knowledge (NIZK) proofs allow a prover to create a proof of membership of an NP language. The proof can be used to convince anybody that indeed the statement in question belongs to the language, but the zero-knowledge property ensures that the proof will reveal nothing but the truth of the statement.

NIZK proofs are fundamental cryptographic primitives used in many constructions, including CCA2-secure cryptosystems, digital signatures, and various cryptographic protocols. The main contribution of this paper is a set of new techniques for constructing NIZK proofs based on groups with a bilinear map. In comparison with previous constructions of NIZK proofs, our techniques yield significant reductions in the length of the common reference string and the size of the proofs.

The new techniques also allow us to answer long standing open questions in the theory of non-interactive zero-knowledge.

- We construct the first perfect NIZK argument system for all NP languages.
- We construct the first UC-secure NIZK argument for all NP languages in the presence of an adaptive adversary.
- We construct the first non-interactive zap for all NP languages based on a standard cryptographic security assumption.

We now describe our contributions in more detail.

1.1 Efficient Non-interactive Zero-Knowledge Proofs

Blum, Feldman, and Micali [BFM88] introduced the notion of NIZK in the common random string model and showed how to construct *computational* NIZK proof systems for proving a single statement about any NP language. The first computational NIZK proof system for multiple theorems was constructed by Blum, De Santis, Micali, and Persiano [BDMP91]. Both [BFM88] and [BDMP91] based their NIZK systems on certain number-theoretic assumptions (specifically, the hardness of deciding quadratic residues modulo a composite number). Feige, Lapidot, and Shamir [FLS99] showed how to construct computational NIZK proofs based on any trapdoor permutation.

Much research has been devoted to the construction of efficient NIZK proofs, but until now the only known method to do so has been the "hidden random bits" method. By this we mean a method where the prover has a string of random bits, which are secret to the verifier. By revealing a subset of these bits, and keeping the rest secret, the prover can convince the verifier of the truth of the statement in question. Improvements in the efficiency of NIZK proofs have come in the form of various ways to set up a hidden random bits model and how to use it optimally.

From a birds eye perspective, the main contribution of this paper is to suggest a set of completely different techniques to construct NIZK proofs. We show that a special type of homomorphic commitment scheme, where it is possible to prove that a commitment contains 0 or 1, implies NIZK proofs for all NP languages. This yields very simple and efficient NIZK proof systems. We show that these proof commitments can be constructed from specific number theoretic assumptions related to groups equipped with a bilinear map.

For comparison with the most efficient previous work, please see Table 1.

1.2 Perfect NIZK Arguments

The plethora of research on NIZK mainly considers NIZK where the zero-knowledge property is only true *computationally*; that is, a computationally bounded party cannot extract any information beyond the correctness of the theorem being proven. In the case of *interactive* zero-knowledge, it has long been known that all NP statements can in fact be proven using *perfect* (or statistical) zero knowledge arguments [BC86, BCC88]; that is, even a computationally unbounded party would not learn anything beyond the correctness of the

Reference	CRS size	Proof Size	Assumption
Damgård [Dam92]	$O(\|C\|k^2 + k^3)$	$O(\|C\|k^2 + k^3)$	Quadratic Residuosity
Kilian-Petrank [KP98]	$O(\|C\|k^2)$	$O(\|C\|k^2)$	Trapdoor Permutations
Kilian-Petrank [KP98]	$O(k^3)$	$O(\|C\|k^3)$	Trapdoor Permutations
De Santis et al. [DDP99, DDP02]	$O(k + \|C\|^\varepsilon)$	poly$(\|C\|k)$	NIZK & One-Way Functions
This paper	$O(k)$	$O(\|C\|k)$	Subgroup Decision [BGN05]
This paper	$O(k)$	$O(\|C\|k)$	Decisional Linear [BBS04]

Table 1: Comparison of CRS size and NIZK proof size for efficient-prover NIZK proof systems for circuit satisfiability.

theorem being proven; though we must assume that the prover, *only during the execution of the protocol*, is computationally bounded to ensure soundness[1].

Achieving perfect or statistical NIZK has been an elusive goal. The original work of [BFM88] showed how a computationally unbounded prover can prove to a polynomially bounded verifier that a number is a quadratic-residue, where the zero-knowledge property is perfect. Statistical ZK (including statistical NIZK[2]) for any non-trivial language for both proofs and arguments were shown to imply the existence of a one-way function by Ostrovsky [Ost91]. Statistical NIZK proof systems were further explored by De Santis, Di Crescenzo, Persiano, and Yung [DDPY98] and Goldreich, Sahai, and Vadhan [GSV99], who gave complete problems for the complexity class associated with statistical NIZK proofs. However, these works came far short of working for all NP languages, and in fact NP-complete languages cannot have (even interactive) statistical zero-knowledge proof systems unless the polynomial hierarchy collapses [For87, AH91][3]. Unless our computational complexity beliefs are wrong, this leaves open only the possibility of argument systems.

Do there exist *statistical* NIZK arguments for all NP languages? Despite nearly two decades of research on NIZK, the answer to this question was not known.

Here, we answer this question in the affirmative. A simple modification to the common reference string in our NIZK proof system transforms the protocol into one with perfect zero-knowledge.

1.3 Universally Composable NIZK Arguments

We generalize our techniques to construct perfect NIZK arguments that satisfy Canetti's UC definition of security. Canetti introduced the universal composability (UC) framework [Can01] as a general method to argue security of protocols in an arbitrary environment. It is a strong security definition; in particular it implies non-malleability [DDN00], and security when arbitrary protocols are executed concurrently.

We define an ideal functionality that captures the notion of NIZK proofs. We then suggest a protocol that securely realizes this functionality against adaptive adversaries in the erasure-free model. In the erasure-free model the adversary can adaptively choose which parties to corrupt and when corrupting a party it learns the internal state and the entire computational history of this party. Not only do we obtain this degree of security, our protocol is also perfect zero-knowledge at the same time.

Prior to our result, no NIZK protocol was known to be UC-secure against adaptive adversaries. In [CLOS02], it was observed that De Santis et al. [DDO$^+$02] achieve UC-security, but only for the setting with *static* adversaries. And [CLOS02] and [DN02] both suggest UC secure zero-knowledge proofs, but these protocols are interactive.

[1] Such systems where the soundness holds computationally have come to be known as *argument systems*, as opposed to *proof systems* where the soundness condition must hold unconditionally.

[2] We note that the result of [Ost91] is for *honest-verifier* SZK, and does not require the simulator to produce the verifier's random tape, and therefore it includes NIZK, even for the common reference string which is not uniform. See also [PS05] for an alternative proof.

[3] See also [GOP98] appendix regarding subtleties of this proof, and [SV03] for an alternative proof.

1.4 Non-interactive Zaps

In 2000, Dwork and Naor [DN00] proved a very surprising result: that there exist "zaps", two-round witness-indistinguishable (WI) proofs in the plain model without a common reference string, where the verifier asks a single question and the prover sends back a single answer. Furthermore, [DN00] showed that their constructions allowed for the first message (from verifier to prover) to be reused – so that between a particular pair of prover and verifier, only one message from verifier to prover is required even if many statements are to be proven. Such zaps were shown to have a number of fascinating and important applications, beyond the numerous applications of WI proofs already present in the literature. Dwork and Naor's work left open the following tantalizing question: does there exist a *non-interactive* witness-indistinguishable proof, where the prover sends a single message to the verifier for some non-trivial NP-language?

Barak, Ong, and Vadhan [BOV03] constructed the first non-interactive zaps for any NP relation by applying derandomization techniques to the construction of Dwork and Naor, based on trapdoor permutations and the assumption that (very good) Hitting Set Generators (HSG) against co-nondeterministic circuits exist. It is known that such HSG's can be built if there is a function in E that requires exponential-size *nondeterministic* circuits – *i.e.* the assumption states that some uniform exponential deterministic computations can (only) be sped up by at most a constant power (Time 2^{cn} becomes $2^{\varepsilon n}$), when given the added power of nondeterminism and advice specific to the length of the input.

We give a new affirmative answer to the question by constructing a non-interactive zap. Our construction is completely different from the construction of Barak, Ong and Vadhan and uses a different, number-theoretic computational assumption. Furthermore, our construction is much more efficient than both the constructions of Dwork-Naor and Barak-Ong-Vadhan (even if these constructions were instantiated with very efficient NIZK proofs from this paper).

A further point of comparison would be to look more closely at the assumptions used, for instance in the context of Naor's classification of assumption based on falsifiability [Nao03]. While our assumption, the decisional linear assumption, is an "efficiently falsifiable" assumption according to Naor's classification, it appears that the assumption about the existence of HSG's against co-nondeterministic circuits, or the assumption about functions in E with large nondeterministic circuits, are "none of the above" assumptions according to Naor's classification, since we wouldn't have time to actually "run" a suggested nondeterministic (or co-nondeterministic) circuit that claims to break the assumption.[4]

2 Definitions: Non-interactive Proofs

Let R be an efficiently computable binary relation. For pairs $(x, w) \in R$ we call x the statement and w the witness. Let L be the language consisting of statements in R.

A non-interactive proof system [BFM88] for a relation R consists of a common reference string generation algorithm K, a prover P and a verifier V. We require that they all be probabilistic polynomial time algorithms, *i.e.*, we are looking at *efficient prover* proofs. The common reference string generation algorithm produces a common reference string σ of length $\Omega(k)$. The prover takes as input (σ, x, w) and produces a proof π. The verifier takes as input (σ, x, π) and outputs 1 if the proof is acceptable and 0 if rejecting the proof. We call (K, P, V) a proof system for R if it has the completeness and soundness properties described below.

[4] We note that there is some uncertainty as to how to interpret Naor's classification with respect to these derandomization-style assumptions. We take a view that we think is consistent with the spirit of Naor's classification by asking the question – if the assumption is false, then is there necessarily a reasonably efficient (PPT) algorithmic demonstration of the falsehood of this assumption? To us, it appears that the answer is "Yes" for our assumption, but appears to be "No" for the [BOV03] assumptions; this is simply because for the latter assumptions, it is important that the breaking algorithm could be non-deterministic – and if it is, then how can we efficiently verify that it indeed does break the assumption? It would be very interesting if in fact there were a positive answer to this. Of course the question of falsifiability is less important than the question of whether an assumption is actually true; alas, we find ourselves unequipped to address this issue.

PERFECT COMPLETENESS. For all adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k); (x,w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow P(\sigma,x,w) : V(\sigma,x,\pi) = 1 \text{ if } (x,w) \in R\right] = 1.$$

PERFECT OR COMPUTATIONAL SOUNDNESS. For all $x \notin L$ and all adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k); \pi \leftarrow \mathcal{A}(\sigma) : V(\sigma,x,\pi) = 1\right] = 0.$$

In computational soundness, we only quantify over non-uniform polynomial-time adversaries, and we only require the above probability to be negligible in k.[5]

PERFECT KNOWLEDGE EXTRACTION. We call (K,P,V) a proof of knowledge for R if there exists a probabilistic polynomial time knowledge extractor $E = (E_1, E_2)$ which allows us to extract a witness from a proof.

For all adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k) : \mathcal{A}(\sigma) = 1\right] = \Pr\left[(\sigma,\xi) \leftarrow E_1(1^k) : \mathcal{A}(\sigma) = 1\right],$$

and

$$\Pr\left[(\sigma,\xi) \leftarrow E_1(1^k); (x,\pi) \leftarrow \mathcal{A}(\sigma); w \leftarrow E_2(\sigma,\xi,x,\pi) : (x,w) \in R \text{ if } V(\sigma,x,\pi) = 1\right] = 1.$$

Since perfect knowledge extraction implies the existence of a witness for the statement being proven, it implies perfect soundness.

COMPUTATIONAL OR PERFECT (ADAPTIVE MULTI-THEOREM) ZERO-KNOWLEDGE [FLS99]. We say a non-interactive proof (K,P,V) is zero-knowledge if there exists a polynomial time simulator $S = (S_1, S_2)$ with the following zero-knowledge property. For all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k) : \mathcal{A}^{P(\sigma,\cdot,\cdot)}(\sigma) = 1\right] \approx \Pr\left[(\sigma,\tau) \leftarrow S_1(1^k) : \mathcal{A}^{S(\sigma,\tau,\cdot,\cdot)}(\sigma) = 1\right],$$

where $S(\sigma,\tau,x,w) = S_2(\sigma,\tau,x)$ for $(x,w) \in R$ and both oracles output `failure` if $(x,w) \notin R$.

If the two probabilities are equal, we say that (K,P,V) is *perfect* zero-knowledge.

NON-ERASURE ZERO-KNOWLEDGE. In modeling adaptive UC security without erasures, an honest prover may be corrupted at some time. To handle such cases, we want to extend the zero-knowledge property such that not only can we simulate an honest party making a proof, we also want to be able to simulate how it constructed the proof. Once the party is corrupted, the adversary will learn the witness and the randomness used; we want to create convincing randomness so that it looks like the simulated proof was constructed by an honest prover using this randomness.

We say a non-interactive proof (K,P,V) is a non-erasure NIZK argument or proof for R if there exists a probabilistic polynomial time simulator $S = (S_1, S_2, S_3)$ so for all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k) : \mathcal{A}^{PR(\sigma,\cdot,\cdot)}(\sigma) = 1\right] \approx \Pr\left[(\sigma,\tau) \leftarrow S_1(1^k) : \mathcal{A}^{SR(\sigma,\tau,\cdot,\cdot)}(\sigma) = 1\right],$$

where $PR(\sigma,x,w)$ picks randomness r, runs $\pi \leftarrow P(\sigma,x,w;r)$ and returns π, r, and where SR picks randomness ρ, runs $\pi \leftarrow S_2(\sigma,\tau,x;\rho); r \leftarrow S_3(\sigma,\tau,x,w,\rho)$ and returns π, r, both of the oracles outputting `failure` if $(x,w) \notin R$.

If the two probabilities are equal, we speak of perfect non-erasure zero-knowledge. Obviously, non-erasure zero-knowledge implies zero-knowledge, and perfect non-erasure zero-knowledge implies perfect zero-knowledge.

[5] We call a function $f : \mathbb{N} \to [0,1]$ negligible if for all $c > 0$ there exists a K so for all $k > K$ we have $f(k) \leq k^{-c}$. We write $f(k) \approx g(k)$ if $|f(k) - g(k)|$ is negligible.

WITNESS-INDISTINGUISHABILITY AND NON-INTERACTIVE ZAP. We call a (P, V) a non-interactive zap for R if (I, P, V), where $I(1^k) = 1^k$, is a non-interactive proof, and for all non-uniform polynomial time interactive adversaries $\mathcal{A}$ we have

$$\begin{aligned} & \Pr\Big[(x, w_0, w_1) \leftarrow \mathcal{A}(1^k); \pi \leftarrow P(1^k, x, w_0) : \mathcal{A}(\pi) = 1 \text{ and } (x, w_0), (x, w_1) \in R\Big] \\ \approx \quad & \Pr\Big[(x, w_0, w_1) \leftarrow \mathcal{A}(1^k); \pi \leftarrow P(1^k, x, w_0) : \mathcal{A}(\pi) = 1 \text{ and } (x, w_0), (x, w_1) \in R\Big]. \end{aligned}$$

A hybrid argument shows that this definition is equivalent to one where we give the adversary access to multiple proofs using either witness w_0 or w_1.

3 Homomorphic Proof Commitments

We will use a non-interactive commitment scheme with some special properties that we define in this section. Recall first that in a non-interactive commitment scheme there is a key generator, which generates a public commitment key ck. The commitment key ck defines a message space $\mathcal{M}_{ck}$, a randomizer space $\mathcal{R}_{ck}$ and a commitment space $\mathcal{C}_{ck}$. We will require that the key generation algorithm is probabilistic polynomial time and outputs keys of length $\theta(n)$. It will in general be obvious which key we are using, so we will sometimes omit it in our notation. There is an efficient commitment algorithm com that takes as input the commitment key, a message and a randomizer and outputs a commitment, $c = \mathrm{com}(m; r)$. We call (m, r) an opening of c.

The commitment scheme must be binding and hiding. Binding means that it is infeasible to find two openings with different messages of the same commitment. Hiding means that given a commitment it is infeasible to guess which message is inside the commitment. We want a commitment scheme that has two different flavors of keys. The commitment key can be perfectly binding, in which case a valid commitment uniquely defines one possible message. Alternatively, the commitment key can be perfectly hiding, in which case the commitment reveals no information whatsoever about the message. In fact, we can create perfect hiding keys together with some trapdoor information such that we can open a commitment to any message. We require that these two kinds of keys are computationally indistinguishable.

We will consider commitments, where both the message space $(\mathcal{M}, +, 0)$, the randomizer space $(\mathcal{R}, +, 0)$ and the commitment space $(\mathcal{C}, \cdot, 1)$ are finite abelian groups. The commitment scheme should be homomorphic, *i.e.*, for all messages and randomizers we have

$$\mathrm{com}(m_1 + m_2; r_1 + r_2) = \mathrm{com}(m_1; r_1)\mathrm{com}(m_2; r_2).$$

We will require that the message space has a generator 1, and also that it has at least order 3. The property that sets proof commitments apart from other commitments, is that there is a way to prove that a commitment contains 0 or 1. More precisely, if the key is of the perfect binding type, then it is possible to prove that there exists an opening $(m, r) \in \{0, 1\} \times \mathcal{R}$. On the other hand, if it is a perfect hiding key, then the proof will be perfectly witness-indistinguishable, *i.e.*, it is impossible to tell whether the message is 0 or 1.

HOMOMORPHIC PROOF COMMITMENT. $(K_{\text{binding}}, K_{\text{hiding}}, \mathrm{com}, \mathrm{Topen}, P_{01}, V_{01})$ is a homomorphic proof commitment scheme if it satisfies the following properties for all non-uniform polynomial time adversaries $\mathcal{A}$.

Key indistinguishability:

$$\Pr\Big[(ck, xk) \leftarrow K_{\text{binding}}(1^k) : \mathcal{A}(ck) = 1\Big] \approx \Pr\Big[(ck, tk) \leftarrow K_{\text{hiding}}(1^k) : \mathcal{A}(ck) = 1\Big].$$

Homomorphic property:

$$\begin{aligned} \Pr\Big[\text{mode} \leftarrow \{\text{binding}, \text{hiding}\}; (ck, *) \leftarrow K_{\text{mode}} : \\ \forall (m_1, r_1), (m_2, r_2) \in \mathcal{M} \times \mathcal{R} : \mathrm{com}(m_1 + m_2; r_1 + r_2) = \mathrm{com}(m_1; r_1)\mathrm{com}(m_2; r_2)\Big] = 1. \end{aligned}$$

Perfect binding:

$$\Pr\Big[(ck, xk) \leftarrow K_{\text{binding}}(1^k) : \exists (m_1, r_1), (m_2, r_2) \in \mathcal{M}\times\mathcal{R} \text{ so } m_1 \neq m_2 \text{ and } \text{com}(m_1; r_1) = \text{com}(m_2; r_2)\Big] = 0.$$

Perfect trapdoor opening:

$$\begin{aligned}\Pr\Big[&(ck, tk) \leftarrow K_{\text{hiding}}(1^k); (m_1, m_2) \leftarrow \mathcal{A}(ck); r_1 \leftarrow \mathcal{R}; r_2 \leftarrow \text{Topen}(m_1, r_1, m_2) : \\ &\text{com}(m_1; r_1) = \text{com}(m_2; r_2) \text{ if } m_1, m_2 \in \mathcal{M}\Big] = 1.\end{aligned}$$

Perfect trapdoor opening indistinguishability:

$$\begin{aligned}&\Pr\Big[(ck, tk) \leftarrow K_{\text{hiding}}(1^k); m \leftarrow \mathcal{A}(ck); r_1 \leftarrow \mathcal{R}; r_2 \leftarrow \text{Topen}(m, r_1, m) : \mathcal{A}(r_2) = 1\Big] \\ = \quad &\Pr\Big[(ck, tk) \leftarrow K_{\text{hiding}}(1^k); m \leftarrow \mathcal{A}(ck); r_1 \leftarrow \mathcal{R} : \mathcal{A}(r_1) = 1\Big].\end{aligned}$$

Perfect completeness:

$$\begin{aligned}\Pr\Big[&\text{mode} \leftarrow \{\text{binding}, \text{hiding}\}; (ck, *) \leftarrow K_{\text{mode}}(1^k); (m, r) \leftarrow \mathcal{A}(ck); \pi \leftarrow P_{01}(m, r) : \\ &V_{01}(\text{com}(m; r), \pi) = 1 \text{ if } (m, r) \in \{0, 1\} \times \mathcal{R}\Big] = 1.\end{aligned}$$

Perfect soundness:

$$\Pr\Big[(ck, xk) \leftarrow K_{\text{binding}}(1^k); (c, \pi) \leftarrow \mathcal{A}(ck) : \exists (m, r) \in \{0, 1\}\times\mathcal{R} \text{ so } c = \text{com}(m; r) \text{ if } V_{01}(c, \pi) = 1\Big] = 1.$$

Perfect witness indistinguishability:

$$\begin{aligned}&\Pr\Big[(ck, tk) \leftarrow K_{\text{hiding}}(1^k); (r_0, r_1) \leftarrow \mathcal{A}(ck); \pi \leftarrow P_{01}(0, r_0) : \\ &\qquad\qquad r_0, r_1 \in \mathcal{R} \text{ and } \text{com}(0; r_0) = \text{com}(1; r_1) \text{ and } \mathcal{A}(\pi) = 1\Big] \\ = \quad &\Pr\Big[(ck, tk) \leftarrow K_{\text{hiding}}(1^k); (r_0, r_1) \leftarrow \mathcal{A}(ck); \pi \leftarrow P_{01}(ck, 1, r_1) : \\ &\qquad\qquad r_0, r_1 \in \mathcal{R} \text{ and } \text{com}(0; r_0) = \text{com}(1; r_1) \text{ and } \mathcal{A}(\pi) = 1\Big].\end{aligned}$$

Perfect extractability. We can strengthen the definition of a proof commitment by requiring that we generate perfect binding keys such that we also have an extraction key that permits extraction of the message inside the commitment. We say the commitment scheme has perfect extractability if there is an extraction algorithm Ext such that

$$\Pr\Big[(ck, xk) \leftarrow K_{\text{binding}}(1^k) : \forall (m, r) \in \{0, 1\} \times \mathcal{R} : \text{Ext}(ck, xk, \text{com}(m; r)) = m\Big].$$

Perfect non-erasure witness indistinguishability. Consider a multi-party computation protocol, where we want to prove adaptive security. The adversary may corrupt some party, and in the security proof we may have to simulate the randomness and a computational history for this party that would explain its public view. If this party has computed some commitment and a proof that the commitment contains 0 or 1, then we can explain the randomness in the commitment by making a trapdoor opening. Here we will strengthen the witness indistinguishability such that we can also explain the proof. Given a proof, which may be created with one witness and some randomness, we want to come up with convincing randomness

that could explain the use of another witness. Let $\mathcal{R}_{\text{proof}}$ be the randomizer space used in the proof. We say the commitment scheme has perfect non-erasure witness indistinguishability if there is a polynomial time simulator S_{01} such that for all interactive adversaries $\mathcal{A}$ we have

$$\begin{aligned}
&\Pr\Big[(ck,tk) \leftarrow K_{\text{hiding}}(1^k); (m,r_0,r_1) \leftarrow \mathcal{A}(ck); \rho_0 \leftarrow \mathcal{R}_{\text{proof}}; \pi \leftarrow P_{01}(m,r_0;\rho_0); \rho_1 \leftarrow S_{01}(m,r_0,r_1,\rho_0) : \\
&\qquad (m,r_0,r_1) \in \{0,1\} \times \mathcal{R} \times \mathcal{R} \text{ and } \text{com}(m;r_0) = \text{com}(1-m;r_1) \text{ and } \mathcal{A}(\pi,\rho_1) = 1\Big] \\
= \quad &\Pr\Big[(ck,tk) \leftarrow K_{\text{hiding}}(1^k); (m,r_0,r_1) \leftarrow \mathcal{A}(ck); \rho_1 \leftarrow \mathcal{R}_{\text{proof}}; \pi \leftarrow P_{01}(1-m,r_1;\rho_1) : \\
&\qquad (m,r_0,r_1) \in \{0,1\} \times \mathcal{R} \times \mathcal{R} \text{ and } \text{com}(m;r_0) = \text{com}(1-m;r_1) \text{ and } \mathcal{A}(\pi,\rho_1) = 1\Big].
\end{aligned}$$

In the following two sections we give two examples of candidates for commitment schemes with these properties.

4 Homomorphic Proof Commitments based on the Subgroup Decision Assumption

Boneh, Goh and Nissim [BGN05] suggest a cryptosystem with interesting homomorphic properties that can be used to build a proof commitment scheme.

Bilinear groups. We use two cyclic groups $\mathbb{G}, \mathbb{G}_T$ of order n, where $n = pq$ and $p < q$ are primes. We make use of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$ we have $e(u^a, v^b) = e(u,v)^{ab}$. We require that $e(g,g)$ is a generator of $\mathbb{G}_T$ if g is a generator of $\mathbb{G}$. We also require that group operations, group membership, sampling of a random generator for $\mathbb{G}$ and the bilinear map be efficiently computable.

[BGN05] suggest the following example. Pick large primes $p < q$ and let $n = pq$. Find the smallest ℓ so $P = \ell n - 1$ is prime and equal to 2 modulo 3. Consider the points on the elliptic curve $y^2 = x^3 + 1$ over $\mathbb{F}_P$. This curve has $P + 1 = \ell n$ points, so it has a subgroup $\mathbb{G}$ of order n. We let $\mathbb{G}_T$ be the order n subgroup of $\mathbb{F}^*_{P^2}$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be the modified Weil-pairing from Boneh and Franklin [BF03].

The subgroup decision problem. Let $\mathcal{G}_{\text{BGN}}$ be a randomized algorithm that outputs $(p, q, \mathbb{G}, \mathbb{G}_T, e, g)$ such that $p < q$ are primes and $n = pq$ is a k-bit number and $\mathbb{G}, \mathbb{G}_T$ are descriptions of groups of order n, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map and g is a random generator for $\mathbb{G}$.

Let $\mathbb{G}_q$ be the subgroup of $\mathbb{G}$ of order q. The subgroup decision problem is to distinguish elements of $\mathbb{G}$ from elements of $\mathbb{G}_q$.

Definition 1 *The subgroup decision assumption holds for generator $\mathcal{G}_{\text{BGN}}$ if for any non-uniform polynomial time adversary $\mathcal{A}$ we have*

$$\begin{aligned}
&\Pr\Big[(p,q,\mathbb{G},\mathbb{G}_T,e,g) \leftarrow \mathcal{G}_{\text{BGN}}(1^k); n = pq; r \leftarrow \mathbb{Z}^*_n; h = g^r : \mathcal{A}(n,\mathbb{G},\mathbb{G}_T,e,g,h) = 1\Big] \\
\approx \quad &\Pr\Big[(p,q,\mathbb{G},\mathbb{G}_T,e,g) \leftarrow \mathcal{G}_{\text{BGN}}(1^k); n = pq; r \leftarrow \mathbb{Z}^*_q; h = g^{pr} : \mathcal{A}(n,\mathbb{G},\mathbb{G}_T,e,g,h) = 1\Big].
\end{aligned}$$

We remark that we have changed the wording of the subgroup decision problem slightly in comparison with [BGN05], but the definitions are equivalent.

4.1 A Homomorphic Proof Commitment Scheme

We will use the subgroup decision assumption to create a proof commitment scheme. To create a perfectly binding key, we set up groups with a bilinear map, where the subgroup decision problem is hard. We pick a generator g and an element h of order q. To commit to $m \in \mathbb{Z}_p$, we form $g^m h^r$ for random $r \in \mathbb{Z}_n$. This is the cryptosystem from [BGN05]. On the other hand, if we want to make perfectly hiding commitments, we choose h of order n, in which case we have a standard Pedersen commitment [Ped91].

It is possible to make a non-interactive proof that a commitment contains 0 or 1. Consider the commitment $c = g^m h^r$. If $h \in \mathbb{G}_q$, this uniquely defines $m \in \mathbb{Z}_p$. Observe, $m \in \{0, 1\}$ if and only if one of c or cg^{-1} has order q. Our task therefore reduces to proving that $e(c, cg^{-1})$ has order q. Since

$$e(c, cg^{-1}) = e(g^m h^r, g^{m-1} h^r) = e(g^m, g^{m-1}) e(h^r, g^{2m-1} h^r) = e(h, (g^{2m-1} h^r)^r),$$

we can simply reveal the proof $\pi = (g^{2m-1} h^r)^r$ and the verifier can check the above equation. Since h has order q, this implies $e(c, cg^{-1})$ has order q. With these ideas in mind, we offer a proof commitment scheme in Figure 1.

Perfectly binding key generation:

1. $(p, q, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\mathrm{BGN}}(1^k)$
2. $n = pq$
3. $x \leftarrow \mathbb{Z}_q^*$
4. $h = g^{px}$
5. Let $ck = (n, \mathbb{G}, \mathbb{G}_T, e, g, h)$
6. Let $xk = q$
7. Return (ck, xk)

Perfectly hiding key generation:

1. $(p, q, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\mathrm{BGN}}(1^k)$
2. $n = pq$
3. $x \leftarrow \mathbb{Z}_n^*$
4. $h = g^x$
5. Let $ck = (n, \mathbb{G}, \mathbb{G}_T, e, g, h)$
6. Let $tk = x$
7. Return (ck, tk)

Commitment: The key ck defines message space $\mathcal{M} = \mathbb{Z}_p$, randomizer space $\mathbb{Z}_n$, and commitment space $\mathbb{G}$. To commit to message $m \in \mathbb{Z}_p$ do

1. $r \leftarrow \mathbb{Z}_n$
2. Return $c = g^m h^r$

Extraction: On a perfect binding key we can use $xk = q$ to extract m of length $\mathcal{O}(\log k)$ from c as follows. Compute $c^q = (g^m h^r)^q = (g^q)^m$ and exhaustively search for m.

Trapdoor opening: Given a commitment $c = g^m h^r$ under a perfectly hiding commitment key we have $c = g^{m'} h^{r-(m'-m)/x}$. So we can create a perfectly hiding commitment and open it to any value we wish if we have the trapdoor key $tk = x$.

WI proof: Given $m, r \in \{0, 1\} \times \mathbb{Z}_n$ we make the WI proof for commitment to 0 or 1 as $\pi = (g^{2m-1} h^r)^r$.

Verification: To verify a WI proof π of commitment c containing 0 or 1, check $e(c, cg^{-1}) = e(h, \pi)$.

Figure 1: Homomorphic proof commitment scheme based on the subgroup decision assumption.

Theorem 2 *The protocol described in Figure 1 is a homomorphic proof commitment scheme with perfect extraction and perfect non-erasure witness indistinguishability if the subgroup decision assumption holds for* $\mathcal{G}_{\text{BGN}}$.

Proof. The subgroup decision assumption implies that it is hard to distinguish perfect binding keys and perfect hiding keys. It is straightforward to see that on either type of key the commitment scheme is homomorphic. When h has order q, we have perfect binding and perfect extraction. When h has order n, we have a unique trapdoor opening to any message. This leaves to demonstrate that we have a witness-indistinguishable proof of a commitment containing 0 or 1.

Let us first prove that we have perfect completeness. No matter whether ck is a perfect binding key or a perfect hiding key, it is the case that for $m \in \{0,1\}$ we have $e(c, cg^{-1}) = e(g^m h^r, g^{m-1}h^r) = e(g,g)^{m(m-1)} e(h^r, g^{2m-1}h^r) = e(h, \pi)$.

Let us now demonstrate that we have perfect soundness on perfect binding keys. We can always write $c = g^m h^r$ for some uniquely defined $m \in \mathbb{Z}_p$. We have $e(c, cg^{-1}) = e(g,g)^{m(m-1)} e(h, (g^{2m-1}h^r)^r)$. Since h has order q, $e(h,\pi)$ has order 1 or q. The verification $e(c, cg^{-1}) = e(h,\pi)$ implies that $e(c, cg^{-1})$ has order 1 or q. Since $e(c, cg^{-1}) = e(g,g)^{m(m-1)} e(h, (g^{2m-1}h^r)^r)$, we see that $e(g,g)^{m(m-1)}$ has order 1 or q. Since $e(g,g)$ is a generator for $\mathbb{G}_T$ this means that $m(m-1) = 0 \bmod p$, and thus $m = 0 \bmod p$ or $m = 1 \bmod p$.

Finally, let us show that we have perfect non-erasure witness indistinguishability on a perfect hiding key. Suppose we have $c = g^0 h^{r_0} = g^1 h^{r_1}$. Since h is a generator for $\mathbb{G}$ there is a unique proof π so $e(c, cg^{-1}) = e(h,\pi)$ and both witnesses make us produce the same proof. So we have perfect witness indistinguishability. Furthermore, since the prover algorithm is deterministic we automatically get perfect non-erasure witness indistinguishability since there is no randomness to reconstruct. □

5 Homomorphic Proof Commitments based on the Decisional Linear Assumption

We will now describe another example of a proof commitment scheme based on a cryptosystem from [BBS04].

BILINEAR GROUPS. We use two cyclic groups $\mathbb{G}, \mathbb{G}_T$ of order p, where p is a prime. We make use of a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$ we have $e(u^a, v^b) = e(u,v)^{ab}$. We require that $e(g,g)$ is a generator of $\mathbb{G}_T$ if g is a generator of $\mathbb{G}$. We also require that group operations, group membership and the bilinear map be efficiently computable.

Boneh and Franklin [BF03] give an example of such a bilinear group. Let $p = 2 \bmod 3$ be a k-bit prime, and choose a small ℓ so $q = \ell p - 1$ is prime. Then the elliptic curve $y^2 = x^3 + 1$ over $\mathbb{Z}_q$ has ℓp points. We can let $\mathbb{G}$ be the order p subgroup of this curve and $\mathbb{G}_T = \mathbb{F}^*_{q^2}$. The bilinear map is the modified Weil-pairing. To get a random generator g for this group, pick x at random such that $x^3 + 1$ is a square and let y be a randomly chosen square root. Then $g = (x,y)^\ell$ is a random generator for $\mathbb{G}$ provided $g \neq 1$.

Let $\mathcal{G}_{\text{DLIN}}$ be a randomized algorithm that takes a security parameter as input and outputs $(p, \mathbb{G}, \mathbb{G}_T, e, g)$ such that p is a prime, $\mathbb{G}, \mathbb{G}_T$ are descriptions of groups of order p, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map as described above and g is a random generator of $\mathbb{G}$.

Definition 3 (Decisional Linear Assumption) *We say the decisional linear assumption holds for the bilinear group generator* $\mathcal{G}_{\text{DLIN}}$ *if for all non-uniform polynomial time adversaries* $\mathcal{A}$ *we have*

$$\begin{aligned} & \Pr\left[(p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\text{DLIN}}; x, y \leftarrow \mathbb{Z}_p^*; r, s \leftarrow \mathbb{Z}_p : \mathcal{A}(p, \mathbb{G}, \mathbb{G}_T, e, g, g^x, g^y, g^{xr}, g^{ys}, g^{r+s}) = 1\right] \\ \approx \quad & \Pr\left[(p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\text{DLIN}}; x, y \leftarrow \mathbb{Z}_p^*; r, s, d \leftarrow \mathbb{Z}_p : \mathcal{A}(p, \mathbb{G}, \mathbb{G}_T, e, g, g^x, g^y, g^{xr}, g^{ys}, g^d) = 1\right]. \end{aligned}$$

The decisional linear assumption was first introduced by Boneh, Boyen and Shacham [BBS04] and has since been used in several cryptographic constructions. We call a tuple of the form (f^r, h^s, g^{r+s}) a *linear tuple* with respect to (f, h, g). When the basis (f, h, g) is obvious from context, we omit mention of it.

5.1 A Homomorphic Proof Commitment Scheme

We will use the decisional linear assumption to create a homomorphic proof commitment. The idea is to let g, f, h be generators of $\mathbb{G}$ and u, v, w another triple of elements in $\mathbb{G}$. A perfect hiding commitment key will contain (u, v, w), which is a linear tuple with respect to g, f, h. Then for any message $m \in \mathbb{Z}_p$ and randomizer $(r, s) \in \mathbb{Z}_p \times \mathbb{Z}_p$ we have a commitment $(u^m f^r, v^m h^s, w^m g^{r+s})$, which is a random linear tuple itself, and therefore reveals nothing about m. On the other hand, if (u, v, w) is not a linear tuple, then the commitment is perfectly binding. The decisional linear assumption implies that it is hard to distinguish perfect binding keys and perfect hiding keys.

Theorem 4 *The protocol in Figure 2 is a homomorphic proof commitment scheme with perfect extraction and perfect non-erasure witness-indistinguishability if the decision linear assumption holds for* $\mathcal{G}_{\mathrm{DLIN}}$.

Proof. Under the decisional linear assumption for $\mathcal{G}_{\mathrm{DLIN}}$ no non-uniform polynomial time adversary can distinguish between (u, v, w) being a linear tuple or not, and hence perfectly hiding keys and perfectly binding keys are indistinguishable.

The commitment scheme is homomorphic under entry-wise multiplication because for either type of commitment key we have

$$\begin{aligned} & \mathrm{com}(m_1 + m_2; r_1 + r_2, s_1 + s_2) = (u^{m_1+m_2} f^{r_1+r_2}, v^{m_1+m_2} h^{s_1+s_2}, w^{m_1+m_2} g^{r_1+r_2+s_1+s_2}) \\ = \quad & (u^{m_1} f^{r_1}, v^{m_1} h^{s_1}, w^{m_1} g^{r_1+s_1})(u^{m_2} f^{r_2}, v^{m_2} h^{s_2}, w^{m_2} g^{r_2+s_2}) = \mathrm{com}(m_1; r_1, s_1)\mathrm{com}(m_2; r_2, s_2). \end{aligned}$$

It is straightforward to see that the protocol is perfectly binding and has perfect extraction, when (u, v, w) is not a linear tuple. On the other hand, when (u, v, w) is a linear tuple then any commitment to any message is a linear tuple and thus perfectly hiding. We can compute a unique trapdoor opening of such a commitment to any value.

Perfect completeness of the witness-indistinguishable proof on either type of keys follows from direct verification. Let us now prove that the proof is perfectly sound on perfect binding keys.

The commitment uniquely defines $m, r, s \in \mathbb{Z}_p$ so $c_1 = u^m f^r, c_2 = v^m h^r, c_3 = w^m g^{r+s}$. We wish to prove that given a valid proof π it must be the case that $m \in \{0, 1\}$. Define r_0, s_0, t_0 and r_1, s_1, t_1 so $c = (f^{r_0}, h^{s_0}, g^{t_0})$ and $c = (uf^{r_1}, vh^{s_1}, wg^{t_1})$. For $i = 1, 2$ let

$$m_{i1} = \log_f(\pi_{i1}) \qquad m_{i2} = \log_h(\pi_{i1}) \qquad m_{i3} = \log_g(\pi_{i3}).$$

Let

$$m_{31} = m_{11} + m_{21} \qquad m_{32} = m_{12} + m_{22} \qquad m_{33} = m_{13} + m_{23}.$$

From the verification we get

$$\begin{aligned} m_{11} &= r_0 r_1 & m_{12} + m_{21} &= r_0 s_1 + s_0 r_1 \\ m_{22} &= s_0 s_1 & m_{13} + m_{31} &= r_0 t_1 + t_0 r_1 \\ m_{33} &= t_0 t_1. & m_{23} + m_{32} &= s_0 t_1 + t_0 s_1 \end{aligned}$$

This means

$$\begin{aligned} & (r_0 + s_0 - t_0)(r_1 + s_1 - t_1) \\ = \quad & r_0 r_1 + r_0 s_1 + s_0 r_1 + s_0 s_1 + t_0 t_1 - (r_0 t_1 + t_0 r_1 + s_0 t_1 + t_0 s_1) \\ = \quad & m_{11} + m_{12} + m_{21} + m_{22} + m_{33} - m_{13} - m_{31} - m_{23} - m_{32} = 0. \end{aligned}$$

We conclude

$$t_0 = r_0 + s_0 \quad \text{or} \quad t_1 = r_1 + s_1,$$

so at least one of (c_1, c_2, c_3) and $(c_1 u^{-1}, c_2 v^{-1}, c w^{-1})$ must be a linear tuple. This shows that c or $c\mathrm{com}(-1; 0, 0)$ is a commitment to 0.

Perfectly binding key generation:

1. $(p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\mathrm{DLIN}}(1^k)$
2. $x, y \leftarrow \mathbb{Z}_p^*$
3. $f = g^x, h = g^y$
4. $r_u, s_v \leftarrow \mathbb{Z}_p$
5. $(u, v, w) = (f^{r_u}, h^{s_v}, g^{r_u+s_v+z})$, where $z \leftarrow \mathbb{Z}_p^*$
6. Let $ck = (p, \mathbb{G}, \mathbb{G}_T, e, g, f, h, u, v, w)$
7. Let $xk = (x, y, z)$ and return (ck, xk)

Perfectly hiding key generation:

1. $(p, \mathbb{G}, \mathbb{G}_T, e, g) \leftarrow \mathcal{G}_{\mathrm{DLIN}}(1^k)$
2. $x, y \leftarrow \mathbb{Z}_p^*$
3. $f = g^x, h = g^y$
4. $r_u, s_v \leftarrow \mathbb{Z}_p$
5. $(u, v, w) = (f^{r_u}, h^{s_v}, g^{r_u+s_v}$
6. Let $ck = (p, \mathbb{G}, \mathbb{G}_T, e, g, f, h, u, v, w)$
7. Let $tk = (r_u, s_v)$ and return (ck, tk)

Commitment: The key ck defines message space $\mathcal{M} = \mathbb{Z}_p$, randomizer space $\mathbb{Z}_p \times \mathbb{Z}_p$, and commitment space $\mathbb{G}^3$. To commit to message $m \in \mathbb{Z}_p$ pick $(r, s) \leftarrow \mathbb{Z}_p \times \mathbb{Z}_p$ and return

$$c = (c_1, c_2, c_3) = \mathrm{com}(m; r, s) = (u^m f^r, v^m h^s, w^m g^{r+s}).$$

Extraction: On a perfect binding key we can extract m of length $\mathcal{O}(\log k)$ from $c = (c_1, c_2, c_3)$ as follows. Compute $(g^z)^m = c_3 c_1^{-1/x} c_2^{-1/y}$ and exhaustively search for m.

Trapdoor opening: Given a commitment $c = (u^m f^r, v^m h^s, w^m g^{r+s})$ under a perfectly hiding commitment key we have $c = (u^{m'} f^{r-(m'-m)r_u}, v^{m'} h^{s-(m-m')s_v}, w^{m'} g^{r+s-(m-m')(r_u+s_v)})$. So we can create a perfectly hiding commitment and open it to any value we wish if we have the trapdoor key $tk = (r_u, s_v)$.

WI proof: Given witness consisting of an opening $(m, r, s) \in \{0, 1\} \times \mathbb{Z}_p \times \mathbb{Z}_p$ we make a proof as follows. Choose $t \leftarrow \mathbb{Z}_p$ and let

$$\begin{array}{lll} \pi_{11} = (u^{2m-1} f^r)^r & \pi_{12} = v^{(2m-1)r} h^{rs-t} & \pi_{13} = w^{(2m-1)r} g^{(r+s)r+t} \\ \pi_{21} = u^{(2m-1)s} f^{rs+t} & \pi_{22} = (v^{2m-1} h^s)^s & \pi_{23} = w^{(2m-1)s} g^{(r+s)s-t} \end{array}$$

Return the proof $\pi = (\pi_{11}, \pi_{12}, \pi_{13}, \pi_{21}, \pi_{22}, \pi_{23})$.

Verification: On input (ck, c, π) compute $\pi_{3j} = \pi_{1j}\pi_{2j}$ for $j = 1, 2, 3$. Accept the proof if and only if

$$\begin{array}{ll} e(f, \pi_{11}) = e(c_1, c_1 u^{-1}) & e(f, \pi_{12}) e(h, \pi_{21}) = e(c_1, c_2 v^{-1}) e(c_2, c_1 u^{-1}) \\ e(h, \pi_{22}) = e(c_2, c_2 v^{-1}) & e(f, \pi_{13}) e(g, \pi_{31}) = e(c_1, c_3 w^{-1}) e(c_3, c_1 u^{-1}) \\ e(g, \pi_{33}) = e(c_3, c_3 w^{-1}) & e(h, \pi_{23}) e(g, \pi_{32}) = e(c_2, c_3 w^{-1}) e(c_3, c_2 v^{-1}). \end{array}$$

Figure 2: Homomorphic proof commitment scheme based on the decisional linear assumption.

On a perfect hiding key, both c and $c\mathrm{com}(-1; 0, 0)$ are linear tuples. Define, (r_0, s_0) and $(r_1, s_1) = (r_0 - r_u, s_0 - s_v)$ so $c = (f^{r_0}, h^{s_0}, g^{r_0+s_0})$ and $c\mathrm{com}(-1; 0, 0) = (f^{r_1}, h^{s_1}, g^{r_1+s_1})$. The witness is on the form $(0, r_0, s_0)$ or $(1, r_1, s_1)$. In the proof we pick $t \leftarrow \mathbb{Z}_p$ at random. All we need to observe now is that witness (r_0, s_0) with randomness t gives the same proof as using (r_1, s_1) using randomness $t' = t + r_0 s_1 - s_0 r_1$. Perfect non-erasure witness indistinguishability now follows from the observation that if we have a proof generated with randomness t, then once we get the witness there are two possibilities. It can be the same witness as

used in the proof, in which case we are done. Or it can be the other witness, which we with knowledge of r_u, s_v can compute and which also allows us to compute randomness t' corresponding to this witness. □

6 Non-interactive Zero-Knowledge Proof for Circuit SAT

In this section, we will describe a NIZK proof for Circuit SAT. We use a public key for a homomorphic proof commitment scheme as the common reference string. In the proofs, the common reference string will be a perfect binding key, while in the simulation it will be a perfect hiding key. The prover gets as input a circuit C, which without loss of generality consists of NAND-gates. He also gets a witness w, consisting of wires $w_1, \ldots, w_{\text{out}}$ such that the wires respect the circuit and the output wire is true, $w_{\text{out}} = 1$. We write $C(w) = 1$, when this is the case.

The prover's strategy is straightforward. He commits to each wire and for each commitment makes a proof that it contains 0 or 1. This way, the verifier is guaranteed that the prover has committed to truth-values for each wire. The prover makes a trivial commitment to the output wire, using randomness $r = 0$, so the verifier can easily check that indeed the output is 1. What remains is to convince the verifier that the committed wires respect the NAND-gates of the circuit.

We make the following observation, leaving the proof to the reader.[6]

Lemma 5 *Let $\mathcal{M}$ be a finite cyclic group with neutral element 0 and generator 1. Let $b_0, b_1, b_2 \in \{0,1\}$.*
If the order of the group is at least 4, then

$$b_2 = \neg(b_0 \wedge b_1) \qquad \text{if and only if} \qquad b_0 + b_1 + 2(b_2 - 1) \in \{0,1\}.$$

If the order of the group is 3, then

$$b_2 = \neg(b_0 \wedge b_1) \qquad \text{if and only if} \qquad b_0 + b_1 + 2(b_2 - 1) \in \{0,1\} \text{ and } b_0 + b_1 + b_2 - 1 \in \{0,1\}.$$

In the following, we focus on the case where the message space of the commitment scheme has order at least 4, leaving the case of order 3 to the reader. Given commitments c_0, c_1, c_2 containing plaintexts b_0, b_1, b_2 the homomorphic property of the commitment scheme implies that $\left(c_0 \cdot c_1 \cdot c_2^2 \cdot \text{com}(-1;0)^2\right)$ is a commitment to $b_0 + b_1 + 2(b_2 - 1)$. A proof that this commitment contains 0 or 1 shows that $b_2 = \neg(b_0 \wedge b_1)$. The prover will make such a proof for each NAND-gate in the circuit.

Theorem 6 *The protocol in Figure 3 is an NIZK proof of knowledge for circuit satisfiability. It has perfect completeness, perfect soundness, and computational zero-knowledge. If the proof commitment scheme has perfect extraction, then it is a perfect proof of knowledge. If the commitment scheme has perfect non-erasure witness indistinguishability, then it is a proof with computational non-erasure zero-knowledge.*

Proof. Knowing a satisfying assignment w for C, we have truth-values for all wires that are consistent with the NAND-gates and with the output wire being 1. Perfect completeness follows from the homomorphic property of the commitment scheme and the perfect completeness of the proofs of committed messages being either 0 or 1.

We prove in Lemma 7 that we have perfect soundness. If the commitment scheme is extractable, then we can extract the wire-values from the commitments, which by the perfect soundness corresponds to a witness w so $C(w) = 1$.

By the indistinguishability of perfect binding keys and perfect hiding keys for the commitment scheme, we have

$$\Pr\left[\sigma \leftarrow K(1^k) : \mathcal{A}^{P(\sigma,\cdot,\cdot)}(\sigma) = 1\right] \approx \Pr\left[(\sigma,\tau) \leftarrow S_1(1^k) : \mathcal{A}^{P(\sigma,\tau,\cdot,\cdot)}(\sigma) = 1\right],$$

where both oracles output `failure` if $(\sigma, x, w) \notin R$.

[6] Similar equations exist for all possible binary gates, so it is not necessary to restrict the circuit to NAND-gates. We only restrict ourselves to NAND-gates because it simplifies the exposition.

Common reference string:

1. $(ck, xk) \leftarrow K_{\text{binding}}(1^k)$
2. The common reference string is $\sigma = ck$.

Statement: The statement is a circuit C built from NAND-gates. The claim is that there exist wires $w = (w_1, \ldots, w_{\text{out}})$ so $C(w) = 1$.

Proof: Input (σ, C, w) so $C(w) = 1$

1. Commit to each bit w_i as $r_i \leftarrow \mathcal{R}; c_i = \text{com}(w_i; r_i)$
2. For the output wire let $r_{\text{out}} = 0$ and $c_{\text{out}} = \text{com}(1; 0)$
3. For all c_i make a WI proof π_i of existence of an opening (w_i, r_i) so $w_i = \{0, 1\}$ and $c_i = \text{com}(w_i; r_i)$
4. For all NAND-gates, we do the following. We have input wires numbered i, j and output wire k. Using message $w_i + w_j + 2w_k - 2$ and randomness $r_i + r_j + 2r_k$ make a WI proof π_{ijk} for $c_i c_j c_k^2 \text{com}(-1; 0)^2$ containing message 0 or 1.
5. Return the proof π consisting of all the commitments and proofs

Verification: Input (σ, C, π).

1. Check that all wires have a corresponding commitment and $c_{\text{output}} = \text{com}(1; 0)$.
2. Check that all commitments have a proof of the message being 0 or 1.
3. Check that all NAND-gates with input wires i, j and output wire k have a proof π_{ijk} for $c_i c_j c_k^2 \text{com}(-1; 0)^2$ containing 0 or 1
4. Return 1 if all checks pass, else return 0

Figure 3: NIZK proof for circuit satisfiability.

Lemma 8 shows that if we simulate the common reference string by $(\sigma, \tau) = (ck, tk) \leftarrow K_{\text{hiding}}(1^k)$ then we have perfect zero-knowledge, so we conclude that we have computational zero-knowledge for (K, P, V). Lemma 8 also shows that we get perfect non-erasure zero-knowledge on a simulated common reference string, so we conclude that (K, P, V) has computational non-erasure zero-knowledge. □

Lemma 7 *The triple (K, P, V) has perfect soundness.*

Proof. Since we prove for each wire that the commitment contains either 0 or 1, we have made a perfectly binding commitment to a truth-value for each wire. By Lemma 5, the WI proofs for the gates imply that all committed truth-values respect the NAND-gates. Finally, we know that the output commitment is $\text{com}(1; 0)$, so the output bit is 1. □

Lemma 8 *The triple (S_σ, P, V) has perfect zero-knowledge, where S_σ is K_{hiding} restricted to the first part of its output. If the proof commitment scheme has perfect non-erasure witness indistinguishability, then we get perfect non-erasure zero-knowledge.*

Proof. Let us first describe the three simulator algorithms. $S_1 = K_{\text{hiding}}$ generates the common reference string $\sigma = ck$ as well as the simulation key $\tau = tk$

S_2 on input (σ, τ, C) sets $c_{\text{output}} = \text{com}(1; 0)$. For all other wires, it selects a commitment $c_i = \text{com}(0; r_i)$ with $r_i \leftarrow \mathcal{R}$. Later, when S_3 learns a witness w, it can compute the corresponding messages $w_i \in \{0, 1\}$ for all these ciphertexts, and open them as $r'_i = \text{Topen}(0, r_i, w_i)$.

For each of these commitments S_2, using witness $(0, r_i)$, makes a WI proof that they contain 0 or 1. For each NAND-gate, with wires i, j and k, it trapdoor opens c_k to 1 as $r'_k \leftarrow \text{Topen}(tk, r_k, 1)$. Using the witness $(0, r_i + r_j + 2r'_j)$ it can now make a WI proof that $c_i c_j c_k^2 \text{com}(-2; 0)$ contains 0 or 1.

Later, upon learning the witness w, S_3 knows the messages $w_i \in \{0, 1\}$. It trapdoor opens all commitments as $r'_i \leftarrow \text{Topen}(0, r_i, w_i)$. Now $c_i = \text{com}(w_i; r'_i)$, where $C(w) = 1$. By the perfect witness indistinguishability of the proofs, we cannot distinguish the simulation from a proof having used this witness. Furthermore, if the WI proofs have perfect non-erasure witness indistinguishability, we can reconstruct randomness in the WI proofs that corresponds to these openings of the commitments. □

Corollary 9 *If the subgroup decision assumption holds for $\mathcal{G}_{\text{BGN}}$, then there exists a NIZK proof for circuit SAT with perfect completeness, perfect soundness, perfect knowledge extraction and computational non-erasure zero-knowledge. The common reference string has size $\mathcal{O}(k)$ and the proofs have size $\mathcal{O}(|C|k)$.*

Corollary 10 *If the decisional linear assumption holds for $\mathcal{G}_{\text{DLIN}}$, then there exists a NIZK proof for circuit SAT with perfect completeness, perfect soundness, perfect knowledge extraction and computational non-erasure zero-knowledge. The common reference string has size $\mathcal{O}(k)$ and the proofs have size $\mathcal{O}(|C|k)$.*

Remark on the uniform random string model. Consider the proof commitment in Section 5 based on the decisional linear assumption. We note that in the common reference string, if p is a prime number, then if we let g, f, h, u, v, w be randomly chosen elements of $\mathbb{G}$, with overwhelming probability they will form a viable common reference string such that (u, v, w) are a non-linear tuple with respect to (f, h, g), and therefore the resulting commitment scheme is perfectly binding. If, for instance, the group is the one suggested by Boneh and Franklin [BF03], then all that is needed to define $\mathbb{G}$ is the prime p. Thus, we can implement our NIZK proofs in the uniform random string model, where the random string is first used to obtain a k-bit prime p using standard methods (just dividing up the uniform random string into k-bit chunks and checking one-by-one if they are prime will do), and then the remaining randomness is used to randomly determine g, f, h, u, v, w (by picking random order p points on the curve). Such an NIZK proof will not have perfect soundness, but statistical soundness, since the probability of (u, v, w) being a linear tuple is exponentially small in k. In the uniform random string model this is optimal, since for any NIZK proof system with a uniform random string there is a risk of accidentally selecting a simulation string.

7 Perfect NIZK Argument for Circuit SAT

In this section, we construct an NIZK argument for circuit satisfiability with perfect zero-knowledge. The main idea is a simple modification of the NIZK proof for circuit satisfiability in Figure 3. Instead of using a perfect binding key as the common reference string, we use a perfect hiding key as the common reference string.

Theorem 11 *(S_σ, P, V) is an NIZK argument for circuit satisfiability with perfect completeness, computational soundness and perfect zero-knowledge, where S_σ is K_{hiding} restricted to the first part of its output. If the proof commitment scheme has perfect non-erasure witness indistinguishability, then the protocol is perfect non-erasure zero-knowledge.*

Proof. As in the proof of Theorem 6, we can show that the protocol has perfect completeness. Perfect zero-knowledge and perfect non-erasure zero-knowledge follows from Lemma 8. This leaves us with the question of soundness.

Consider a non-uniform polynomial time adversary $\mathcal{A}$ and an arbitrary family of polynomially bounded size unsatisfiable circuits $\{C_k\}$. From the key indistinguishability of the homomorphic proof commitment scheme and the perfect soundness of (K, P, V) we have

$$\begin{aligned} & \Pr\left[\sigma \leftarrow S_\sigma(1^k); \pi \leftarrow \mathcal{A}(\sigma, C_k) : V(\sigma, C_k, \pi) = 0\right] \\ \approx \quad & \Pr\left[\sigma \leftarrow K(1^k); \pi \leftarrow \mathcal{A}(\sigma, C_k) : V(\sigma, C_k, \pi) = 0\right] \;=\; 0. \end{aligned}$$

□

7.1 Adaptive Soundness

In the examples based on the subgroup decision assumption and the decisional linear assumption, we do not know whether (S_σ, P, V) is a NIZK argument with computational *adaptive* soundness, where the adversary can choose the statement x after seeing the common reference string. When an adversary can choose a statement adaptively, it may express properties about the common reference string itself. For instance, C could be the statement that *ck* is a perfectly hiding commitment key. Now we can no longer argue soundness on the basis that the two common reference strings are indistinguishable, since switching from S_σ to K also switches to a setting where C is satisfiable.

On the other hand, it turns out that the schemes we have constructed are sufficient to construct a UC NIZK argument, as we shall see in the next section. Due to the great generality of the UC framework, we take this as an indication that adaptive soundness may not be so important after all. Nonetheless, we do prove in Appendix A that using complexity leveraging techniques we can obtain adaptive soundness for non-trivial size circuits, although the circuits must be very small in comparison with the security parameter.

We do not know whether our perfect NIZK argument system has adaptive soundness. However, we can prove that the perfect NIZK argument has an adaptive soundness property, which we will call co-soundness. Consider a language of circuits in co-NP. Co-soundness says that it is infeasible for an adversary to find an unsatisfiable circuit, and a witness for unsatisfiability, and at the same time form a valid perfect NIZK argument for satisfiability. In other words, in some sense, the adversary cannot *know* that it succeeded in proving a false statement.

Adaptive co-soundness. Let R_{co} be a binary relation consisting of circuits C and witnesses w_{co} for C being unsatisfiable. We say (S_σ, P, V) has adaptive co-soundness if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow S_\sigma(1^k); (C, \pi, w_{\text{co}}) \leftarrow A(\sigma) : V(\sigma, C, \pi) = 1 \text{ and } (C, w_{\text{co}}) \in R_{\text{co}}\right] \approx 0.$$

Theorem 12 *The perfect NIZK argument (S_σ, P, V) based on homomorphic proof commitments enjoys adaptive co-soundness.*

Proof. By the indistinguishability of perfect binding keys and perfect hiding keys and the perfect soundness of (K, P, V), we have for all non-uniform polynomial time adversaries $\mathcal{A}$ that

$$\begin{aligned} & \Pr\left[\sigma \leftarrow S_\sigma(1^k); (C, \pi, w_{\text{co}}) \leftarrow A(\sigma) : V(\sigma, C, \pi) = 1 \text{ and } (C, w_{\text{co}}) \in R_{\text{co}}\right] \\ \approx \quad & \Pr\left[\sigma \leftarrow K(1^k); (C, \pi, w_{\text{co}}) \leftarrow A(\sigma) : V(\sigma, C, \pi) = 1 \text{ and } (C, w_{\text{co}}) \in R_{\text{co}}\right] \;=\; 0. \end{aligned}$$

□

Let us give an example to illustrate the significance of Theorem 12. Consider a setup, where we have a cryptosystem and an encrypted message. An adversary might try to cheat and create a fake NIZK argument that the plaintext has some property, which it does not have. However, according to Theorem 12 this is not possible, because the randomness in the key generation algorithm serves as a coNP-witness that the statement is false. From the randomness of the key generation algorithm, we can generate the public key and the decryption key, and we can therefore decrypt the ciphertext and check that the statement is false. In the next section we will see that the coNP-soundness of the perfect NIZK argument is exactly what we need to build a UC NIZK argument with perfect zero-knowledge.

8 Universally Composable Non-interactive Zero-Knowledge

8.1 Modeling Non-interactive Zero-Knowledge Arguments

The universal composability (UC) framework (see [Can01] for a detailed description) is a strong security model capturing security of a protocol under concurrent execution of arbitrary protocols. We model all other things not directly related to the protocol through an environment $\mathcal{Z}$. The environment can at its own choosing give inputs to the parties running the protocol, and according to the protocol specification, the parties can give outputs to the environment. In addition, there is an adversary $\mathcal{A}$ that attacks the protocol. $\mathcal{A}$ can communicate freely with the environment. It can also corrupt parties, in which case it learns the entire history of that party and gains complete control over the actions of this party. The environment learns whenever a party is corrupted.

To model security we use a simulation paradigm. We specify the functionality $\mathcal{F}$ that the protocol should realize. The functionality $\mathcal{F}$ can be seen as a trusted party that handles the entire protocol execution and tells the parties what they would output if they executed the protocol correctly. In the ideal process, the parties simply pass on inputs from the environment to $\mathcal{F}$ and whenever receiving a message from $\mathcal{F}$ they output it to the environment. In the ideal process, we have an ideal process adversary $\mathcal{S}$. $\mathcal{S}$ does not learn the content of messages sent from $\mathcal{F}$ to the parties, but is in control of when, if ever, a message from $\mathcal{F}$ is delivered to the designated party. $\mathcal{S}$ can corrupt parties, at the time of corruption it will learn all inputs the party has received and all outputs it has sent to the environment. As the real world adversary, $\mathcal{S}$ can freely communicate with the environment.

We now compare these two models and say that the protocol securely realizes $\mathcal{F}$ if no environment can distinguish between the two worlds. This means, the protocol is secure, if for any polynomial time $\mathcal{A}$ running in the real world, there exists a polynomial time $\mathcal{S}$ running in the ideal process with $\mathcal{F}$, so no non-uniform polynomial time environment can distinguish between the two worlds.

The standard zero-knowledge functionality $\mathcal{F}_{\text{ZK}}$ as defined in [Can01] goes as follows: On input $(\textbf{prove}, P, V, sid, ssid, x, w)$ from P the functionality $\mathcal{F}_{\text{ZK}}$ checks that $(x, w) \in R$ and in that case sends $(\textbf{proof}, P, V, sid, ssid, x)$ to V.[7] It is thus part of the model that the prover will send the proof to a particular receiver and that this receiver will learn who the prover is. This is a very reasonable model when we talk about interactive zero-knowledge proofs of knowledge. We remark that with only small modifications in the UC NIZK argument that we are about to suggest we can securely realize this functionality.

When we talk about NIZK arguments we do not always know who is going to receive the NIZK argument. We simply create a string π, which is the NIZK argument. We may create this string in advance and later decide to whom to send it. Furthermore, anybody who intercepts the string π can verify the truth of the statement and can use the string to convince others about the truth of the statement. The NIZK argument is not deniable; quite on the contrary, it is transferable [Pas03]. For this reason, and because the protocol and the security proof becomes a little simpler, we suggest a different functionality $\mathcal{F}_{\text{NIZK}}$ to capture the essence of NIZK arguments, see Figure 4.

8.2 Tools

We will use a perfect non-erasure NIZK argument with adaptive coNP-soundness $(S_\sigma, P, V, S_1, S_2, S_3)$ as described in the previous section. In addition, we use the following cryptographic tools to securely realize $\mathcal{F}_{\text{NIZK}}$.

Perfectly hiding commitment scheme with extraction. A perfectly hiding commitment scheme with extraction has the following property. We can run a key generation algorithm $hk \leftarrow K_{\text{hiding}}(1^k)$ to get a hiding key hk, or we can alternatively run a key generation algorithm $(hk, xk) \leftarrow K_{\text{extract}}(1^k)$ in which case we get both a hiding key hk and an extraction key xk. $(K_{\text{hiding}}, \text{com})$ constitute a perfectly hiding

[7] The role of the session identifier *sid* and subsession identifier *ssid* is to distinguish different functionalities $\mathcal{F}$ and calls to these functionalities.

Parameterized with relation R and running with parties $P_1, \ldots, P_n$ and adversary $\mathcal{S}$.

Proof: On input (**prove**,$sid, ssid, x, w$) from party P ignore if $(x, w) \notin R$. Send (**prove**,x) to $\mathcal{S}$ and wait for answer (**proof**, π). Upon receiving the answer store (x, π) and send (**proof**, $sid, ssid, \pi$) to P.

Verification: On input (**verify**, $sid, ssid, x, \pi$) from V check whether (x, π) is stored. If not send (**verify**,x, π) to $\mathcal{S}$ and wait for an answer (**witness**,w). Upon receiving the answer, check whether $(x, w) \in R$ and in that case, store (x, π). If (x, π) has been stored return (**verification**,$sid, ssid$,1) to V, else return (**verification**,$sid, ssid$,0).

Figure 4: NIZK argument functionality $\mathcal{F}_{\text{NIZK}}$.

commitment scheme. On the other hand, $(K_{\text{extract}}, \text{com}, \text{dec})$ constitute a public key cryptosystem with errorless decryption, *i.e.*,

$$\Pr\left[(hk, xk) \leftarrow K_{\text{extract}}(1^k) : \forall (m, r) : \text{dec}_{xk}(\text{com}_{hk}(m; r)) = m\right] = 1. \tag{1}$$

We demand that no non-uniform polynomial time adversary $\mathcal{A}$ can distinguish between the two key generation algorithms. This implies that the cryptosystem is semantically secure against chosen plaintext attack since the perfectly hiding commitment does not reveal what the message is. Observe that proof commitment schemes imply the existence perfectly hiding commitment schemes with extraction.

Pseudorandom cryptosystem. A cryptosystem $(K_{\text{pseudo}}, E, D)$ has pseudorandom ciphertexts of length $\ell_E(k)$ if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\begin{aligned} & \Pr\left[(pk, dk) \leftarrow K_{\text{pseudo}}(1^k) : \mathcal{A}^{E_{pk}(\cdot)}(pk) = 1\right] \\ \approx \quad & \Pr\left[(pk, dk) \leftarrow K_{\text{pseudo}}(1^k) : \mathcal{A}^{R_{pk}(\cdot)}(pk) = 1\right], \end{aligned} \tag{2}$$

where $R_{pk}(m)$ runs $c \leftarrow \{0,1\}^{\ell_E(k)}$ and returns c. We require that the cryptosystem have errorless decryption as defined earlier.

Trapdoor permutations imply pseudorandom cryptosystems, we can use the Goldreich-Levin hard-core bit [GL89] of a trapdoor permutation to make a one-time pad. In our concrete setting, we note that the subgroup decision assumption for $\mathcal{G}_{\text{BGN}}$ implies hardness of factorization. Rabin encryption [Rab79] is secure if factorization is hard, and it is possible to transform Rabin-encryption into a pseudorandom cryptosystem.

Assuming the decisional linear assumption for the elliptic curve example in Section 5, we can also get a pseudorandom cryptosystem. The observation here is that ciphertexts are on the form (u, v, w), which look like random group elements. These are elements in the order p subgroup of an elliptic curve of order ℓp, where ℓ is a small integer. We can pick random elements in $\mathbb{G}$ by choosing a random point on the curve and raising it to the power ℓ. It is possible to work backwards as well in this process, and come up with plausible randomness that would cause us to select the points u, v, w.

Tag-based simulation-sound trapdoor commitment. A tag-based commitment scheme has four algorithms. The key generation algorithm $K_{\text{tag}-\text{com}}$ produces a commitment key ck as well as a trapdoor key tk. There is a commitment algorithm that takes as input the commitment key ck, a message m and any tag tag and outputs a commitment $c = \text{commit}_{ck}(m, tag; r)$. To open a commitment c with tag tag we reveal m and the randomness r. Anybody can now verify $c = \text{commit}_{ck}(m, tag; r)$. As usual, the commitment scheme must be both hiding and binding.

In addition, to these two algorithms there are also a couple of trapdoor algorithms Tcom, Topen that allow us to create an equivocal commitment and later open this commitment to any value we prefer. We create an equivocal commitment and an equivocation key as $(c, ek) \leftarrow \text{Tcom}_{ck,tk}(tag)$. Later we can open it to any message m as $r \leftarrow \text{Topen}_{ck}(ek, c, m, tag)$, such that $c = \text{commit}_{ck}(m, tag; r)$. We require that equivocal

commitments and openings are indistinguishable from real openings. For all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\begin{aligned} & \Pr\Big[(ck,tk) \leftarrow K_{\text{tag}-\text{com}}(1^k) : \mathcal{A}^{\mathcal{R}(\cdot,\cdot)}(ck) = 1\Big] \\ \approx \quad & \Pr\Big[(ck,tk) \leftarrow K_{\text{tag}-\text{com}}(1^k) : \mathcal{A}^{\mathcal{O}(\cdot,\cdot)}(ck) = 1\Big], \end{aligned} \tag{3}$$

where $\mathcal{R}(m, tag)$ returns a randomly selected randomizer and $\mathcal{O}(m, tag)$ computes $(c, ek) \leftarrow \text{Tcom}_{ck,tk}(m, tag); r \leftarrow \text{Topen}_{ck}(ek, c, m, tag)$ and returns r. Both oracles ignore tags that have already been submitted once.

The tag-based simulation-soundness property means that a commitment using *tag* remains binding even if we have made equivocations for commitments using different tags. For all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\begin{aligned} \Pr\Big[(ck,tk) \leftarrow K(1^k); (c, tag, m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(ck) : tag \notin Q \text{ and} \\ c = \text{commit}_{ck}(m_0, tag; r_0) = \text{commit}_{ck}(m_1, tag; r_1) \text{ and } m_0 \neq m_1\Big] \approx 0, \end{aligned} \tag{4}$$

where $\mathcal{O}(\text{commit}, tag)$ computes $(c, ek) \leftarrow \text{Tcom}_{ck,tk}(tag)$, returns c and stores (c, tag, ek), and $\mathcal{O}(\text{open}, c, m, tag)$ returns $r \leftarrow \text{Topen}_{ck}(ek, c, m, tag)$ if (c, tag, ek) has been stored, and where Q is the list of tags for which equivocal commitments have been made by $\mathcal{O}$.

The term tag-based simulation commitment comes from Garay, MacKenzie and Yang [GMY03], while the definition presented here is from MacKenzie and Yang [MY04]. The latter paper offers a construction based on one-way functions.

Strong one-time signatures. We remind the reader that strong one-time signatures allow a non-uniform polynomial time adversary to ask an oracle for a signature on one arbitrary message. Then it must be infeasible to forge a signature on any different message and infeasible to come up with a different signature on the same message. Strong one-time signatures can be constructed from one-way functions.

8.3 UC NIZK

The standard technique to prove that a protocol securely realizes a functionality in the UC framework is to show that the ideal model adversary $\mathcal{S}$ can simulate everything that happens on top of the ideal functionality. So $\mathcal{S}$ simulates $\mathcal{A}$ and the parties $P_1, \ldots, P_n$ running the protocol. We use the notation $\tilde{P}_i$ for a real party in the ideal process, which simply forwards inputs and outputs between the environment and the ideal functionality, and P_i for a simulated party. In our case, there are two tricky parts. First, $\mathcal{S}$ may learn that a statement C has been proved by $\tilde{P}$ and has to simulate a UC NIZK argument π that P will output without knowing the witness. Furthermore, if P is corrupted later then we can corrupt $\tilde{P}$ and learn the witness but must now simulate the randomness of P that would lead it to produce π. The second problem is that whenever $\mathcal{S}$ sees an acceptable UC NIZK argument π for a statement C, then an honest verifier would accept. We must therefore, input a witness w to $\mathcal{F}_{\text{NIZK}}$ so it can instruct $\tilde{V}$ to accept.

The main idea in overcoming these hurdles is to commit to the witness w and make a non-erasure NIZK argument that indeed we have committed to a witness w so $C(w) = 1$. This way we can simulate NIZK arguments and the prover's random coins.

This leaves us with the commitment scheme. On one hand, when we simulate UC NIZK arguments we want to make equivocal commitments that can be opened to anything since we do not know the witness yet. On the other hand, when we see a UC NIZK argument that we did not construct ourselves we want to be able to extract the witness, since we have to give it to $\mathcal{F}_{\text{NIZK}}$.

We construct such a commitment scheme from the tools specified in the previous section in a manner related to the construction of a UC commitment by Canetti et al. [CLOS02]. We use a tag-based simulation-sound trapdoor commitment scheme to commit to each bit of w. If w has length ℓ this gives us commitments $c_1, \ldots, c_\ell$. For honest provers we can use the trapdoor key tk to create equivocal commitments that can be

opened to any bit we like. This enables us to simulate the commitments of the honest provers, and when we learn w upon corruption, we can simulate the randomness they could have used to commit to the witness w.

We still have an extraction problem, it is not clear that we can extract a witness from tag-based commitments created by a malicious adversary. To solve this problem we choose to encrypt the openings of the commitments. Now we can extract witnesses, but we have reintroduced the problem of equivocation. In a simulated commitment we may know two different openings of a commitment c_i to respectively 0 and 1, however, if we encrypt the opening then we are stuck with one possible opening. This is where the pseudorandomness property of the cryptosystem comes in handy. We can simply make two encryptions, one of an opening to 0 and one of an opening to 1. Since the ciphertexts are pseudorandom, we can open the ciphertext containing the opening we want and claim that the other ciphertext was chosen as a random string. To recap, the idea so far to commit to a bit b is to make a commitment c_i to this bit, and create a ciphertext $c_{i,b}$ containing an opening of c_i to b, while choosing $c_{i,1-b}$ as a random string.

The commitment scheme is equivocable, however, again we must be careful that we can extract a message from an adversarial commitment. The problem is that since we equivocate commitments for honest provers it may be the case that the adversary can produce equivocable commitments. This means, the adversary can produce some simulation sound commitment c_i and encryptions $c_{i,0}, c_{i,1}$ of openings to respectively 0 and 1. To resolve this issue we will select the tags for the commitments in a way so the adversary is forced to use a tag that has not been used to make an equivocable commitment. When an honest prover is making a commitment, we select keys for a strong one-time signature scheme $(vk, sk) \leftarrow K_{\text{sign}}(1^k)$. We use $tag = (vk, C)$ when making the commitment c_i. The verification key vk will be published together with the commitment, and we will sign the commitment (as well as something else) using this key. Since the adversary cannot forge signatures, it must use a different tag, and therefore the commitment is binding and only one of the ciphertexts can contain an opening of c_i.

If the adversary corrupts a party P that has used vk earlier, then it may indeed sign messages using vk and can therefore use vk in the tag for commitments. However, since we also include the statement C in the tag for the commitment using vk, the adversary can only create an equivocable commitment in a UC NIZK argument for the same statement C. We will observe that in this particular case we do not need to extract the witness w, because we can get it during the corruption of the prover $\tilde{P}$.

Finally, in order to make the UC NIZK argument perfect zero-knowledge we wrap all the commitments c_i and the ciphertexts $c_{i,b}$ inside a perfectly hiding commitment c. In the simulation, however, we generate the key for this commitment scheme in a way such that it is instead a cryptosystem and we can extract the plaintext. This last step is only added to make the UC NIZK argument perfect zero-knowledge, it can be omitted if perfect zero-knowledge is not needed.

The resulting protocol can be seen in Figure 5. We use the notation from Section 8.2.

Theorem 13 *The protocol in Figure 7 securely realizes $\mathcal{F}_{\text{NIZK}}$ in the $\mathcal{F}_{\text{CRS}}$-hybrid model.*

Proof. Let $\mathcal{A}$ be any polynomial time adversary. We will describe an ideal adversary $\mathcal{S}$ so no non-uniform polynomial time environment can distinguish whether it is running in the $\mathcal{F}_{\text{CRS}}$-hybrid model with parties $P_1, \ldots, P_n$ and adversary $\mathcal{A}$ or in the ideal process with $\mathcal{F}_{\text{NIZK}}$, $\mathcal{S}$ and dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$.

$\mathcal{S}$ starts by invoking a copy of $\mathcal{A}$. It will run a simulated interaction of $\mathcal{A}$, the parties and the environment. In particular, whenever the simulated $\mathcal{A}$ communicates with the environment, $\mathcal{S}$ just passes this information along. And whenever $\mathcal{A}$ corrupts a party P_i, $\mathcal{S}$ corrupts the corresponding dummy party $\tilde{P}_i$.

SIMULATING $\mathcal{F}_{\text{CRS}}$. $\mathcal{S}$ chooses the common reference string in the following way. It selects, $(hk, xk) \leftarrow K_{\text{extract}}(1^k); (ck, tk) \leftarrow K_{\text{tag-com}}(1^k); (pk, dk) \leftarrow K_{\text{pseudo}}(1^k)$ and $(\sigma, \tau) \leftarrow S_1(1^k)$. This means $\mathcal{S}$ is capable of extracting plaintexts committed under hk, able to create and equivocate simulation sound trapdoor commitments, decrypt pseudorandom ciphertexts, and simulate NIZK arguments and later upon learning a witness simulate convincing randomness.

Let $\Sigma = (hk, ck, pk, \sigma)$. $\mathcal{S}$ simulates $\mathcal{F}_{\text{CRS}}$ sending $(\mathbf{crs}, sid, \Sigma)$ to all parties. Whenever $\mathcal{A}$ decides to deliver such a message to a party P_i, $\mathcal{S}$ will simulate P_i receiving this string.

Common reference string generation:

1. $hk \leftarrow K_{\text{hiding}}(1^k)$
2. $(ck, tk) \leftarrow K_{\text{tag-com}}(1^k)$
3. $(pk, dk) \leftarrow K_{\text{pseudo}}(1^k)$
4. $(\sigma, \tau) \leftarrow S_1(1^k)$
5. Return $\Sigma = (hk, ck, pk, \sigma)$

Proof: On input (Σ, C, w) so $C(w) = 1$ do

1. $(vk, sk) \leftarrow K_{\text{sign}}(1^k)$
2. For $i = 1$ to ℓ select r_i at random and let $c_i = \text{commit}_{ck}(w_i, (vk, C); r_i)$
3. For $i = 1$ to ℓ select R_{w_i} at random and set $c_{i,w_i} = E_{pk}(r_i; R_{w_i})$ and choose $c_{i,1-w_i}$ as a random string.
4. Choose r at random and let $c = \text{com}_{hk}(c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}; r)$
5. Create an NIZK argument π for the statement that there exists w and randomness so c has been produced as described in steps 3,4 and 5 and $C(w) = 1$.
6. $s \leftarrow \text{sign}_{sk}(C, vk, c, \pi)$
7. Return $\Pi = (vk, c, \pi, s)$

Verification: On input (Σ, C, Π)

1. Parse $\Pi = (vk, c, \pi, s)$
2. Verify that s is a signature on (C, vk, c, π) under vk.
3. Verify the NIZK argument π
4. Return 1 if all checks work out, else return 0

Figure 5: UC NIZK argument $(K^{\text{UC}}, V^{\text{UC}}, P^{\text{UC}})$.

Common reference string: On input (**start**,*sid*) run $\Sigma \leftarrow K^{\text{UC}}(1^k)$.

Send (**crs**,*sid*, Σ) to all parties and halt.

Figure 6: Protocol for UC NIZK common reference string generation.

Proof: Party P waits until receiving (**crs**,*sid*, Σ) from $\mathcal{F}_{\text{CRS}}$.

On input (**prove**,*sid*, *ssid*, C, w) so $C(w) = 1$ run $\Pi \leftarrow P^{\text{UC}}(\Sigma, C, w)$. Output (**proof**,*sid*, *ssid*, Π).

Verification: Party V waits until receiving (**crs**,*sid*, Σ) from $\mathcal{F}_{\text{CRS}}$.

On input (**verify**,*sid*, *ssid*, C, Π) run $b \leftarrow V^{\text{UC}}(\Sigma, C, \Pi)$. Output (**verification**,*sid*, *ssid*, b).

Figure 7: Protocol for UC NIZK argument.

Simulating uncorrupted provers. Suppose $\mathcal{S}$ receives (**proof**,*sid*, *ssid*, C) from $\mathcal{F}_{\text{NIZK}}$. This means that some dummy party $\tilde{P}$ received input (**prove**,*sid*, *ssid*, C, w), where $C(w) = 1$. We must simulate the output a real party P would make, however, we may not know w.

We create $(vk, sk) \leftarrow K_{\text{sign}}(1^k)$. Let $tag = (vk, C)$ and form ℓ equivocal commitments $(c_i, ek_i) \leftarrow$

$\text{Tcom}_{pk,tk}(tag)$. We simulate openings of the c_i's to both 0 and 1. For all $i = 1$ to ℓ and $b = 0$ to 1 compute $\rho_{i,b} \leftarrow \text{Topen}_{ck}(ek_i, c_i, b, tag)$. Select $r_{i,b}$ at random and set $c_{i,b} = E_{pk}(\rho_{i,b}; r_{i,b})$. Compute $c = E_{hk}(c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}; r)$ for a random r. Let x be the statement that there exists a witness w and randomness such that c has been correctly generated using w and $C(w) = 1$. Choose randomness ρ and simulate the NIZK argument as $\pi \leftarrow S_2(\sigma, \tau, x; \rho)$. Finally, create a one-time signature s on C, vk, c, π.

Let $\Pi = (vk, c, \pi, s)$ and return (**proof**,Π) to $\mathcal{F}_{\text{NIZK}}$. $\mathcal{F}_{\text{NIZK}}$ subsequently sends (**proof**,$sid, ssid, \Pi$) to $\tilde{P}$ and we deliver this message so it gets output to the environment.

Simulating uncorrupted verifiers. Suppose $\mathcal{S}$ receives (**verify**,C, Π) from $\mathcal{F}_{\text{NIZK}}$. This means an honest dummy party $\tilde{V}$ has received (**verify**,$sid, ssid, C, \Pi$) from the environment.

$\mathcal{S}$ checks the UC NIZK argument, $b \leftarrow V^{\text{UC}}(\Sigma, C, \Pi)$. If invalid, it sends (**witness**,`no witness`) to $\mathcal{F}_{\text{NIZK}}$ and delivers the consequent message (**verification**,$sid, ssid, 0$) to $\tilde{V}$ that outputs this rejection to the environment.

On the other hand, if Π is valid we must try to extract a witness w. If C has ever been proved by an honest prover that was later corrupted, we already know the witness and do not need to run the following extraction procedure. If the witness is not known already $\mathcal{S}$ uses the extraction key xk to extract a plaintext $c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}$ from c. Since it knows the decryption key dk, it can then decrypt all $c_{i,b}$. This gives us plaintexts $\rho_{i,b}$. We check whether $c_i = \text{commit}_{ck}(b, (vk, C); \rho_{i,b})$ and in that case b is a possible candidate for the i-th bit of w.

If successful in all of this, $\mathcal{S}$ lets w be these bits. However, if any of the bits are ambiguous, *i.e.*, w_i could be both 0 and 1, or if any of them are inextractable, then it sets $w =$ `no witness`. It sends (**witness**,w) to $\mathcal{F}_{\text{NIZK}}$. It delivers the resulting output message to $\tilde{V}$ that outputs it to the environment.

We will later argue that the probability of the UC NIZK argument being valid, yet us not being able to extract a witness to give to $\mathcal{F}_{\text{NIZK}}$ is negligible. That means, with overwhelming probability we input a valid witness w to $\mathcal{F}_{\text{NIZK}}$ when Π is an acceptable UC NIZK argument for satisfiability of C.

Simulating corruption. Suppose a simulated party P_i is corrupted by $\mathcal{A}$. Then we have to simulate the transcript of P_i. We start by corrupting $\tilde{P}_i$ thereby learning all UC NIZK arguments it has verified. It is straightforward to simulate P_i's internal tapes when running these verification processes.

We also learn all statements C that it has proved together with the corresponding witnesses w. Recall, the UC NIZK arguments Π have been provided by $\mathcal{S}$. Here is how we can simulate the randomness that would lead P_i to produce such a UC NIZK argument Π. Since $\mathcal{S}$ created $c_i, c_{i,0}, c_{i,1}$ such that $c_{i,0}$ contains a 0-opening of c_i and $c_{i,1}$ contains a 1-opening of c_i it can produce good looking randomness to claim that it committed to w_i. This also gives us convincing randomness for constructing all these commitments and for producing the ciphertext c, so we can run the simulator algorithm S_3 to simulate randomness that would lead the prover to produce π.

Hybrids. We wish to argue that no environment can distinguish between the adversary $\mathcal{A}$ running with parties executing the UC NIZK protocol in the $\mathcal{F}_{\text{CRS}}$-hybrid model and the ideal adversary $\mathcal{S}$ running in the $\mathcal{F}_{\text{NIZK}}$-hybrid model with dummy parties. In order to do so we define several hybrid experiments and show that the environment cannot distinguish between any of them.

H0: This is the $\mathcal{F}_{\text{CRS}}$-hybrid model running with adversary $\mathcal{A}$ and parties $P_1, \ldots, P_n$.

H1: We modify H0 by running $(hk, xk) \leftarrow K_{\text{extract}}(1^k)$ instead of $hk \leftarrow K_{\text{hiding}}(1^k)$ when generating the common reference string Σ.

H0 and H1 are indistinguishable, because otherwise we could build a distinguisher from $\mathcal{A}, \mathcal{Z}$ that could tell which key generation algorithm created hk contradicting Equation (4).

H2: We modify H1 in the way an uncorrupted prover P creates tag-based simulation sound trapdoor commitments $c_1, \ldots, c_\ell$ to the bits of the witness. Let $tag = (vk, C)$ as chosen in the protocol. Instead of creating c_i by selecting r_i at random and setting $c_i = \text{commit}_{ck}(w_i, tag; r_i)$, we create an equivocal commitment $(c_i, ek_i) \leftarrow \text{Tcom}_{ck,tk}(tag)$ and subsequently produce randomness $\rho_{i,w_i} \leftarrow \text{Topen}_{ck}(ek_i, c_i, w_i, tag)$. We continue the proof using ρ_{i,w_i} instead of r_i.

H1 and H2 are indistinguishable. If they were distinguishable, then we could distinguish real commitments and openings from equivocal commitments and equivocated openings, in violation of Equation (6) of the definition of trapdoor commitments.

H3: In H3, we make another modification to the procedure followed by an honest prover. We are already creating c_i as an equivocal commitment and equivocating it with randomness ρ_{i,w_i} that would open it to contain w_i. We run the equivocation procedure once more to also create convincing randomness that would explain c_i as a commitment to $1 - w_i$. This means, we compute $\rho_{i,1-w_i} \leftarrow \text{Topen}_{ck}(ek_i, c_i, 1 - w_i, tag)$. Instead of selecting $c_{i,1-w_i}$ as a random string, we choose to encrypt $\rho_{i,1-w_i}$ as $c_{i,1-w_i} = E_{pk}(\rho_{i,1-w_i}; r_{i,1-w_i})$ for a randomly chosen $r_{i,1-w_i}$. We still pretend that $c_{i,1-w_i}$ is a randomly chosen string when we carry out the NIZK proof π or if the prover is ever corrupted.

H2 and H3 are indistinguishable because of the pseudorandomness property of the cryptosystem, see Equation (5). Suppose we could distinguish H2 and H3, then we could distinguish between an encryption oracle and an oracle that supplies randomly chosen strings.

H4: Consider the case where an honest party V receives (**verify**,$sid, ssid, C, \Pi$). Suppose Π is indeed an acceptable UC NIZK argument and the one-time signature scheme has verification key vk. If vk was selected by an honest party in making a UC NIZK argument and this party is still uncorrupted, yet (C, Π) differ from the UC NIZK argument this honest party produced, then we output `failure` to the environment.

To argue that H3 and H4 are indistinguishable we need to show that the probability of failure is negligible. This follows from the fact that outputting `failure` corresponds to creating a forgery of the strong one-time signature scheme.

H5: Again, we look at the case of an uncorrupted verifier that has an acceptable UC NIZK argument Π for some C to verify. If (C, Π) were produced by an uncorrupted prover we do not change the protocol, neither do we modify the protocol if C has been proved by an honest prover that has later been corrupted. In all other cases, we use the extraction key xk in an attempt to decrypt c to get a plaintext on the form $c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}$. Then we use the decryption key dk to attempt to decrypt the $c_{i,b}$'s to get $\rho_{i,b}$ so $c_{i,b} = \text{commit}_{ck}(b, (vk, C); \rho_{i,b})$. We output `failure` if we encounter a $c_i = \text{commit}_{ck}(0, (vk, C), \rho_{i,0}) = \text{commit}_{ck}(1, (vk, C), \rho_{i,1})$.

Tag-based simulation soundness, see Equation (7), of the commitment scheme implies that H4 and H5 are indistinguishable. Consider the tag (vk, C). Outputting `failure` corresponds to breaking the binding property of the commitment scheme, unless we have previously created an equivocal commitment with tag (vk, C). In H4, we ruled out the possibility of vk coming from a UC NIZK argument of a party that is still uncorrupted. This leaves us with the possibility of $\mathcal{A}$ corrupting an honest prover P, learning the secret key sk corresponding to vk and making a UC NIZK argument using the tag (vk, C). If we have ever created an equivocal commitment using this tag, we did it for this prover. However, this means that C stems from the same honest prover that has now been corrupted, and in that case we do not try to extract $\rho_{i,b}$'s.

H6: As in H5, we try to extract $\rho_{i,0}, \rho_{i,1}$'s. We output `failure` if we cannot decrypt c to get $c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}$. We also output `failure` if there is an i so we cannot decrypt either $c_{i,0}$ or $c_{i,1}$ to give us $\rho_{i,b}$ so $c_i = \text{commit}_{ck}(b, (vk, C); \rho_{i,b})$. We ruled out the possibility of both $\rho_{i,0}$ and $\rho_{i,1}$ being an opening of c_i in H5, so if everything is OK so far we have a uniquely defined w so for all i we have $c_i = \text{commit}_{ck}(w_i, (vk, C); \rho_{i,w_i})$. We output `failure` if $C(w) \neq 1$.

Call c well-formed if we can extract $c_1, c_{1,0}, c_{1,1}, \ldots, c_\ell, c_{\ell,0}, c_{\ell,1}$ from c using xk, and for all $i = 1$ to ℓ at least one of the $c_{i,0}, c_{i,1}$ will have a proper $\rho_{i,b}$ so $c_i = \text{commit}_{ck}(b, (vk, C); \rho_{i,b})$, and if all of these openings are unique then the bits constitute a witness w for $C(w) = 1$. Observe, from the perfect extractability of the commitment scheme and the errorless decryption property of the pseudorandom cryptosystem, we have that the randomness used in creating (hk, xk) and (pk, dk) is a witness to c

being malformed unless indeed it is well-formed. The coNP-soundness of the NIZK argument therefore tells us that with overwhelming probability c is well-formed and we have negligible chance of outputting `failure`. This means H5 and H6 are indistinguishable.

H7: Instead of making real NIZK arguments for uncorrupted provers we use the non-erasure zero-knowledge simulators. We use $\pi \leftarrow S_2(\sigma, \tau, \cdot; \rho)$ with ρ random to simulate the honest provers' NIZK arguments that c has been correctly generated. Finally, if any such prover is corrupted we use $r \leftarrow S_3(\sigma, \tau, x, \pi, \cdot, \rho)$ to create convincing randomness that would make the prover output π on the witness for c being correctly generated.

The non-erasure zero-knowledge property of the NIZK proof implies that H6 and H7 are indistinguishable.

SIM: This is the ideal process running with $\mathcal{F}_{\text{NIZK}}$ and $\mathcal{S}$.

H7 is already very similar to the ideal process. Honest provers in H7 make UC NIZK arguments in the same way as $\mathcal{S}$ without using the knowledge of the witness w for anything. It therefore makes no difference that $\mathcal{S}$ only learns w upon corruption of a party P when it has to simulate the random tape of said party.

Whenever an honest verifier has to verify a proof C, Π we are also very close to what happens in the simulation. If C, Π has been produced by an honest prover it returns 1, as will the dummy verifier in the ideal process. If C is a statement proved by an honest prover, but this prover has later been corrupted, then in H8 the verifier will return 1 if Π is an acceptable UC NIZK argument. $\mathcal{S}$ in a similar situation will have corrupted the dummy prover that made the UC NIZK argument, and therefore it will know the witness. If Π is an acceptable UC NIZK argument, it can therefore give this witness to $\mathcal{F}_{\text{NIZK}}$ that will make the dummy verifier output an acceptance to the environment. Finally, in the remaining case we have argued in H7 that we manage to extract a witness w if Π is acceptable and this extraction procedure is carried out exactly as it is done by $\mathcal{S}$. Therefore, $\mathcal{S}$ can submit this witness to $\mathcal{F}_{\text{NIZK}}$.

In conclusion, H7 is perfectly indistinguishable from the ideal process. Our path from H0 to SIM shows us that H0 and SIM are indistinguishable.

□

Theorem 14 *The UC NIZK argument in Figure 5 is perfect zero-knowledge.*

Proof. We start by describing the simulator $S^{\text{UC}} = (S_1^{\text{UC}}, S_2^{\text{UC}})$. S_1^{UC} runs $hk \leftarrow K_{\text{hiding}}(1^k); (ck, tk) \leftarrow K_{\text{tag-com}}(1^k); (pk, sk) \leftarrow K_{\text{pseudo}}(1^k); (\sigma, \tau) \leftarrow S_1(1^k)$. Let $\Sigma = (hk, ck, pk, \sigma)$. S_1^{UC} outputs (Σ, τ).

Consider next S_2^{UC} that is given a circuit C on which to simulate a UC NIZK argument Π for satisfiability. It generates keys for the strong one-time signature scheme $(vk, sk) \leftarrow K_{\text{sign}}(1^k)$. Then generates a perfectly hiding commitment $c \leftarrow \text{com}_{hk}(0)$. It simulates an argument $\pi \leftarrow S_2(\sigma, \tau, x)$ for the statement x that c has been correctly formed and contains a witness w so $C(w) = 1$. Finally, it creates a one-time signature on everything, $s \leftarrow \text{sign}_{sk}(C, vk, c, \pi)$. It outputs the simulated UC NIZK argument $\Pi = (vk, c, \pi, s)$.

Perfect zero-knowledge of the NIZK argument system implies that for all adversaries $\mathcal{A}$ we have

$$\Pr\left[\Sigma \leftarrow K^{\text{UC}}(1^k) : \mathcal{A}^{P(\Sigma,\cdot,\cdot)}(\Sigma) = 1\right] = \Pr\left[(\Sigma, \tau) \leftarrow S_1^{\text{UC}}(1^k) : \mathcal{A}^{PS(\Sigma,\tau,\cdot,\cdot)}(\Sigma) = 1\right],$$

where PS is an oracle that on input (Σ, τ, C, w) outputs `failure` if $C(w) = 0$ and otherwise creates a UC NIZK argument $\Pi = (vk, c, \pi, s)$ by following the provers algorithm for creating vk, c, s but simulating the NIZK argument π.

Next, we argue that for all adversaries $\mathcal{A}$ we have

$$\Pr\left[(\Sigma, \tau) \leftarrow S_1^{\text{UC}}(1^k) : \mathcal{A}^{PS(\Sigma,\tau,\cdot,\cdot)}(\Sigma) = 1\right] = \Pr\left[(\Sigma, \tau) \leftarrow K^{\text{UC}}(1^k) : \mathcal{A}^{S'(\Sigma,\tau,\cdot,\cdot)}(\Sigma) = 1\right],$$

where $S'(\Sigma, \tau, C, w)$ checks that $C(w) = 1$ and in that case returns $\Pi \leftarrow S_2^{\mathrm{UC}}(\Sigma, \tau, C)$.

The only difference between the two oracles PS and S' is the message inside the commitment c. However, since the commitment scheme is perfectly hiding, this does not change the distributions. □

Corollary 15 *Proof commitment schemes with perfect extraction and pseudorandom cryptosystems imply the existence of a non-interactive perfect zero-knowledge protocol that securely realizes $\mathcal{F}_{\mathrm{NIZK}}$.*

Corollary 16 *Bilinear groups as described in Section 4 for which the subgroup decision assumption holds imply the existence of a non-interactive perfect zero-knowledge protocol that securely realizes $\mathcal{F}_{\mathrm{NIZK}}$.*

Corollary 17 *If the decisional linear assumption holds for the example of bilinear groups based on elliptic curves as described in Section 5 then there exists a non-interactive perfect zero-knowledge protocol that securely realizes $\mathcal{F}_{\mathrm{NIZK}}$.*

9 Non-Interactive Zaps for Circuit Satisfiability

We now give a construction of non-interactive zaps for circuit satisfiability. The main idea is to select two correlated common reference strings in such a way that the verifier can check that at least one of them is a perfect binding commitment key. The prover then forms two proofs, one for each common reference string. Since one of them is perfect binding, this means we get perfect soundness. On the other hand, we do want to be able to generate these two correlated common reference strings in such a way that one of them can be used to simulate proofs and the adversary cannot tell which one of them is a perfect hiding key. This is what will give us witness indistinguishability.

Verifiable correlated key generation. We say a proof commitment scheme has verifiable correlated key generation, if there exists two efficient algorithms K_2, V_2 with the following characteristics. $K_2(1^k, b)$ generates a perfect binding key ck_{1-b} and a perfect hiding key ck_b together with the trapdoor key tk_b. It outputs (ck_0, ck_1, tk_b). V_2 takes as input two commitment keys ck_0, ck_1 and outputs 1 if and only if at least one of the commitment keys is a perfect binding commitment key. We require that it must be hard to tell, which one of ck_0 and ck_1 is perfectly hiding. Formally, for all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\Pr\Big[(ck_0, ck_1, tk) \leftarrow K_2(1^k, 0) : \mathcal{A}(ck_0, ck_1) = 1\Big] \approx \Pr\Big[(ck_0, ck_1, tk) \leftarrow K_2(1^k, 1) : \mathcal{A}(ck_0, ck_1) = 1\Big].$$

If the decisional linear assumption holds for the elliptic curve example of a bilinear group from [BF03] in Section 5, then that proof commitment scheme has verifiable correlated key generation.[8] To generate a pair of keys, we generate $(ck_b, tk_b) = ((p, \mathbb{G}, \mathbb{G}_T, e, g, f, h, u, , v, w), (r_u, s_v)) \leftarrow K_{\mathrm{hiding}}(1^k, b)$. Then we set $ck_{1-b} = (p, \mathbb{G}, \mathbb{G}_T, e, g, f, h, u, v, w^{2b-1})$. The verifier first checks that $(p, \mathbb{G}, \mathbb{G}_T, e)$ describes an elliptic curve with a bilinear map, this is straightforward because it simply corresponds to verifying that $p = 2 \bmod 3$ is a prime. Next, it checks that g, f, h, u, v, w are points on the curve. Then it checks that g, f, h are non-trivial points on the curve. Finally, it checks that ck_0, ck_1 are the same strings, except for the last elements w_0, w_1 and that $w_1 = w_0 g$. At least one of u, v, w_0 and $u, v, w_0 g$ must be a non-linear tuple, and therefore at least one of the keys is perfectly binding.

Theorem 18 *The protocol in Figure 8 is a non-interactive proof in the plain model for Circuit Satisfiability with perfect completeness, perfect soundness and computational witness indistinguishability if the proof commitment scheme has verifiable correlated key generation.*

Proof. The protocol is perfectly complete because the NIZK proofs for Circuit Satisfiability are perfectly complete both on perfect binding keys and perfect hiding keys.

[8] We do not believe that the proof commitment scheme based on the subgroup decision assumption in Section 4 has verifiable correlated key generation.

Proof: The prover given $1^k, C$ and wires w such that $C(w) = 1$ proceeds as follows.

1. Generate a *verifiable* pair of commitment keys $(ck_0, ck_1, tk_0) \leftarrow K_2(1^k, 0)$
2. Use the NIZK prover to obtain a proof π_0 of the statement with respect to the common reference string ck_0
3. Use the NIZK prover to obtain a proof π_1 of the statement with respect to the common reference string ck_1
4. The resulting proof is $\pi = (ck_0, ck_1, \pi_0, \pi_1)$

Verification: On input C and a proof π as described above, accept if and only if the following procedure succeeds

1. Verify that at least one of ck_0, ck_1 is a perfect binding key $V_2(ck_0, ck_1) = 1$
2. Verify π_0 with respect to the common reference string ck_0
3. Verify π_1 with respect to the common reference string ck_1

Figure 8: Non-interactive zap for circuit satisfiability.

Perfect soundness follows from the fact that given $V_2(ck_0, ck_1) = 1$ at least one of ck_0 and ck_1 must be a perfect binding key. The perfect soundness from the proof system over this common reference string then implies that C must be satisfiable.

We now argue (computational) witness indistinguishability assuming verifiable correlated key generation for the proof commitment scheme by means of a hybrid argument. The adversary generates a circuit C and two witnesses w_0 and w_1.

1. The first hybrid is simply the prover algorithm above using witness w_0.
2. The second hybrid proceeds as in the first, except that for π_0, it uses the NIZK prover with witness w_1 to obtain π_0 instead of using witness w_0.

   Hybrid 1 and Hybrid 2 are identically distributed, by means of an intermediate hybrid using the NIZK simulator for π_0, and the fact that the NIZK simulator is a perfect simulator in the case where the common reference string is a perfectly hiding commitment key.
3. The third hybrid proceeds as the second, except that it uses $(ck_0, ck_1, tk_1) \leftarrow K_2(1^k, 1)$ to generate the common reference strings. Now ck_0 is a perfectly binding key and ck_1 is perfectly hiding.

   Hybrid 2 and Hybrid 3 are computationally indistinguishable since no non-uniform polynomial time adversary can distinguish between generating the keys using $K_2(1^k, 0)$ or $K_2(1^k, 1)$.
4. The fourth hybrid proceeds as the third, except that for π_1, it uses the NIZK prover with witness w_1 to obtain π_1 instead of using witness w_0.

   Hybrid 3 and Hybrid 4 are identically distributed for the same reasons Hybrids 1 and 2 were identically distributed.
5. Finally, the fifth hybrid proceeds as the fourth, except that it uses $(ck_0, ck_1, tk_0) \leftarrow K_2(1^k, 0)$ to generate the common reference strings. This is precisely the WI prover algorithm using witness w_1.

   Hybrid 4 and Hybrid 5 are computationally indistinguishable by the same argument showing that Hybrids 2 and 3 were computationally indistinguishable.

□

Corollary 19 *If the decisional linear assumption holds for the elliptic curve based bilinear group in [BF03], then non-interactive zaps exist.*

10 Acknowledgment

We wish to thank Brent Waters for important collaboration. He brought our attention to the fact that witness-indistinguishable proofs for a commitment to 0 or 1 suffice to build NIZK proofs for circuit satisfiability. He also participated at an early stage of the research on building NIZK proof systems based on the decisional linear assumption.

References

[AH91] William Aiello and Johan Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42(3):327–345, 1991.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *proceedings of CRYPTO '04, LNCS series, volume 3152*, pages 41–55, 2004.

[BC86] Gilles Brassard and Claude Crèpeau. Non-transitive transfer of confidence: A perfect zero-knowledge interactive protocol for sat and beyond. In *Proceedings of FOCS '86*, pages 188–195, 1986.

[BCC88] Gilles Brassard, David Chaum, and Claude Crèpeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[BDMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM Jornal of Computation*, 20(6):1084–1118, 1991.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *proceedings of STOC '88*, pages 103–112, 1988.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *proceedings of TCC '05, LNCS series, volume 3378*, pages 325–341, 2005.

[BOV03] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. In *proceedings of CRYPTO '03, LNCS series, volume 2729*, pages 299–315, 2003.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *proceedings of FOCS '01*, pages 136–145, 2001. Full paper available at `http://eprint.iacr.org/2000/067`.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *proceedings of STOC '02*, pages 494–503, 2002. Full paper available at `http://eprint.iacr.org/2002/140`.

[Dam92] Ivan Damgård. Non-interactive circuit based proofs and non-interactive perfect zero-knowledge with proprocessing. In *proceedings of EUROCRYPT '92, LNCS series, volume 658*, pages 341–355, 1992.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. *SIAM Journal of Computing*, 30(2):391–437, 2000.

[DDO+02] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In *proceedings of CRYPTO '01, LNCS series, volume 2139*, pages 566–598, 2002.

[DDP99] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Non-interactive zero-knowledge: A low-randomness characterization of np. In *proceedings of ICALP '99, LNCS series, volume 1644*, pages 271–280, 1999.

[DDP02] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Randomness-optimal characterization of two np proof systems. In *proceedings of RANDOM '02, LNCS series, volume 2483*, pages 179–193, 2002.

[DDPY98] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. Image density is complete for non-interactive-szk. In *proceedings of ICALP '98, LNCS series, volume 1443*, pages 784–795, 1998.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *proceedings of FOCS '00*, pages 283–293, 2000.

[DN02] Ivan Damgård and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In *proceedings of CRYPTO '02, LNCS series, volume 2442*, pages 581–596, 2002. Full paper available at `http://www.brics.dk/RS/01/41/index.html`.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs under general assumptions. *SIAM Journal of Computing*, 29(1):1–28, 1999.

[For87] Lance Fortnow. The complexity of perfect zero-knowledge. In *Proceedings of STOC '87*, pages 204–209, 1987.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *proceedings of STOC '89*, pages 25–32, 1989.

[GMY03] Juan A. Garay, Philip D. MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. In *proceedings of EUROCRYPT '03, LNCS series, volume 2656*, pages 177–194, 2003. Full paper available at `http://eprint.iacr.org/2003/037`.

[GOP98] Oded Goldreich, Rafail Ostrovsky, and Erez Petrank. Computational complexity and knowledge complexity. *SIAM Journal of Computing*, 27:1116–1141, 1998.

[GSV99] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Can statistical zero knowledge be made non-interactive? or on the relationship of szk and niszk. In *CRYPTO '99, LNCS series, volume 1666*, pages 467–484, 1999.

[KP98] Joe Kilian and Erez Petrank. An efficient noninteractive zero-knowledge proof system for np with general assumptions. *Journal of Cryptology*, 11(1):1–27, 1998.

[MY04] Philip D. MacKenzie and Ke Yang. On simulation-sound trapdoor commitments. In *proceedings of EUROCRYPT '04, LNCS series, volume 3027*, pages 382–400, 2004. Full paper available at `http://eprint.iacr.org/2003/252`.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *proceedings of CRYPTO '03, LNCS series, volume 2729*, pages 96–109, 2003.

[Ost91] Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of Structure in Complexity Theory Conference*, pages 133–138, 1991.

[Pas03] Rafael Pass. On deniability in the common reference string and random oracle model. In *proceedings of CRYPTO '03, LNCS series, volume 2729*, pages 316–337, 2003.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *proceedings of CRYPTO '91, LNCS series, volume 576*, pages 129–140, 1991.

[PS05] Rafael Pass and Abhi Shelat. Characterizing non-interactive zero-knowledge in the public and secret parameter models. In *proceedings of CRYPTO '05, LNCS series*, 2005.

[Rab79] Michael O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[SV03] Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003.

A Perfect NIZK Argument with Adaptive Soundness

ADAPTIVE SOUNDNESS. (K, P, V) is said to have adaptive soundness, if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k); (x, \pi) \leftarrow \mathcal{A}(\sigma) : V(\sigma, x, \pi) = 1 \text{ and } x \notin L\right] \approx 0.$$

Consider the perfect NIZK argument (S_σ, P, V) from Section 7. We will bound the probability of an adversary breaking the adaptive soundness on circuits of size less than $\ell(k)$.

Theorem 20 *If perfect binding keys and perfect hiding keys can be distinguished with at most probability $\nu_{\text{KeyDist}}(k) < \ell(k)^{-\ell(k)}\nu(k)$, where ν is a negligible function, then (S_σ, P, V) has adaptive soundness for circuits of size $\ell(k)$.*

Proof.

$$\begin{aligned}
& \Pr\left[\sigma \leftarrow S_\sigma(1^k); (C, \pi) \leftarrow \mathcal{A}(\sigma) : C \notin L \text{ and } |C| \leq \ell(k) \text{ and } V(\sigma, C, \pi) = 1\right] \\
= & \sum_{C \notin L : |C| \leq \ell(k)} \Pr\left[\sigma \leftarrow S_\sigma(1^k); (C', \pi) \leftarrow \mathcal{A}(\sigma) : C' = C \text{ and } V(\sigma, C', \pi) = 1\right] \\
\leq & \sum_{C \notin L : |C| \leq \ell(k)} \Pr\left[\sigma \leftarrow S_\sigma(1^k); (C', \pi) \leftarrow \mathcal{A}(\sigma) : V(\sigma, C, \pi) = 1\right] \\
\leq & \sum_{C \notin L : |C| \leq \ell(k)} \nu_{\text{KeyDist}}(1^k) \quad < \quad \sum_{C \notin L : |C| \leq \ell(k)} \ell(k)^{-\ell(k)}\nu(1^k) \quad \leq \quad \nu(1^k).
\end{aligned}$$

□

Let us make a back of the envelope estimate of how large circuits we can hope to have adaptive soundness for if we are using the subgroup decision assumption, where $\nu_{\text{KeyDist}} = \nu_{\text{SD}}$, the upper bound on the advantage in deciding the subgroup decision problem. If the subgroup decision assumption is broken, it is still not clear whether it leads to an attack on adaptive soundness. However, let us be conservative and aim for circuits of size $\ell(k)$ so $\ell(k)^{\ell(k)}\nu_{\text{SD}}(k)$ is negligible.

The best attack on the subgroup decision assumption we can think of consists of factoring n. The number field sieve algorithms factors n in heuristically

$$e^{(1.92+o(1))(\ln n)^{1/3}(\ln\ln n)^{2/3}} = 2^{(\log e)^{2/3}(1.92+o(1))k^{1/3}(\ln(\frac{k}{\log e}))^{2/3}}$$

steps. From this we make a guess that there exists an algorithm that decides the subgroup decision problem with advantage $2^{-(\log e)^{2/3}(1.92+o(1))k^{1/3}(\ln(\frac{k}{\log e}))^{2/3}}$. This implies that $2^{-(\log e)^{2/3}(1.92+o(1))k^{1/3}(\ln(\frac{k}{\log e}))^{2/3}} < \nu_{\text{SD}}(k)$, so

$$\ell(k)^{\ell(k)}2^{-(\log e)^{2/3}(1.92+o(1))k^{1/3}(\ln(\frac{k}{\log e}))^{2/3}}$$

must be negligible. Letting $\ell(k) = k^\varepsilon$ gives us

$$2^{k^\varepsilon \log(k^\varepsilon) - (\log e)^{2/3}(1.92+o(1))k^{1/3}(\ln(\frac{k}{\log e}))^{2/3}}$$

must be negligible, which is true for any constant $\varepsilon < 1/3$.

Picking for instance $\varepsilon = 1/4$ works. Requiring that the common reference string is at least of size $\ell(k)^4$ bits is not unreasonable in comparison with earlier constructions of computational NIZK proofs. However, security relies on the very strong assumption that any non-uniform polynomial time adversary has at most $\nu_{\text{SD}}(k) = 2^{-(\log k/4)k^{1/4}}$ advantage in the subgroup decision problem.