Minimizing Non-interactive Zero-Knowledge Proofs Using Fully Homomorphic Encryption

Jens Groth*
University College London

January 6, 2011

Abstract

A non-interactive zero-knowledge proof can be used to demonstrate the truth of a statement without revealing anything else. It has been shown under standard cryptographic assumptions that non-interactive zero-knowledge proofs of membership exist for all languages in NP. However, known non-interactive zero-knowledge proofs of membership of NP-languages yield proofs that are larger than the corresponding membership witnesses.

We investigate the question of minimizing the communication overhead involved in making non-interactive zero-knowledge proofs and show that if fully homomorphic encryption exists then it is possible to minimize the size of non-interactive zero-knowledge proofs and get proofs that are of the same size as the witnesses.

Our technique is applicable to many types of non-interactive zero-knowledge proofs. We apply it to both standard non-interactive zero-knowledge proofs and to universally composable non-interactive zero-knowledge proofs. The technique can also be applied outside the realm of non-interactive zero-knowledge proofs, for instance to get witness-size interactive zero-knowledge proofs in the plain model without any setup.

Keywords: Non-interactive zero-knowledge proofs, fully homomorphic encryption.

1 Introduction

Non-interactive zero-knowledge (NIZK) proofs [BFM88] allow the construction of a proof that can convince others about the truth of a statement. We will consider statements of the form $x \in L$, where L can be an arbitrary language in NP. While the proof should guarantee the truth of the statement it should not reveal anything else. More precisely, we require that the NIZK proof is complete, sound and zero-knowledge.

Completeness: Given a witness w for the statement $x \in L$ there is an efficient algorithm to construct a convincing proof π.

Soundness: A malicious prover should not be able to convince anybody if the statement is false.

We focus on unconditional soundness, where even an adversary with infinite computing power cannot create a convincing proof π for $x \notin L$.

Zero-knowledge: A malicious verifier learns nothing but the truth of the statement. In particular, the proof π does not reveal the witness w that the prover used when constructing the proof π.

*Supported by Engineering and Physical Sciences Research Council grant number EP/G013829/1.

Only languages in BPP have NIZK proofs in the plain model without any setup [Ore87, GO94, GK96]. Blum, Feldman and Micali [BFM88] therefore suggested the common reference string model, where the prover and the verifier have access to a bit-string that is assumed to have been generated honestly according to a specific distribution. The common reference string can for instance be generated by a trusted third party or by a set of parties executing a multi-party computation protocol. Groth and Ostrovsky [GO07] has as an alternative suggested NIZK proofs in the multi-string model, where many parties generate a random string and the security of the NIZK proof relies on a majority of the strings being honestly generated.

1.1 Related work

NIZK proofs have many applications, ranging from early chosen-ciphertext secure public-key cryptosystems [DDN00] to recent advanced signature schemes [BW06, CGS07]. There is therefore a significant body of research dealing with NIZK proofs.

Blum, Feldman and Micali [BFM88] proposed an NIZK proof for all of NP based on a number theoretic assumption related to factoring. Feige, Lapidot and Shamir [FLS99] gave an NIZK proof for all of NP based on the existence of trapdoor permutations.

While these results established the existence of NIZK proofs based on general assumptions, other works have aimed at defining stronger security properties such as non-malleability [Sah01], robustness [DDO+02] and universal composability [Can01, GOS06b].

There has been significant progress in reducing the complexity of NIZK proofs based on general assumptions [Dam92, DDP02, KP98, Gro10] and Groth, Ostrovsky and Sahai [GOS06b, GOS06a, Gro06, GS08] have constructed practical NIZK proofs using techniques from pairing-based cryptography. Recently Gentry [Gen09b, Gen09a] proposed a fully homomorphic encryption scheme and demonstrated that fully homomorphic encryption can be used to construct NIZK proofs that are proportional to the size of the witness.

1.2 Our contribution

We construct NIZK proofs for arbitrary NP-languages. The size of the common reference string is $\text{poly}(k)$ and the size of the proof is essentially the same as the witness, i.e., $|\pi| = |w| + \text{poly}(k)$ where k is the security parameter.

In Table 1, we compare our NIZK proofs with the current state of the art NIZK proofs for Circuit Satisfiability based on respectively trapdoor permutations [Gro10] and specific cryptographic assumptions [GOS06b, GOS06a, Gro10, Gen09a]. All of these NIZK proofs are publicly verifiable by anybody who sees the common reference string, the statement and the proof.

	CRS size	Proof size	Assumption
Groth [Gro10]	$\|C\| \cdot \text{poly}(k)$	$\|C\| \cdot \text{poly}(k)$	Trapdoor perm.
GOS [GOS06b, GOS06a]	$\text{poly}(k)$	$\|C\| \cdot \text{poly}(k)$	Pairing-based
Groth [Gro10]	$\|C\| \cdot \text{polylog}(k) + \text{poly}(k)$	$\|C\| \cdot \text{polylog}(k) + \text{poly}(k)$	Naccache-Stern
Gentry [Gen09a]	$\text{poly}(k)$	$\|w\| \cdot \text{poly}(k) + \text{poly}(k)$	FHE and NIZK
This work	$\text{poly}(k)$	$\|w\| + \text{poly}(k)$	FHE and NIZK

Table 1: Comparison of NIZK proofs for security parameter k, circuit size $|C|$ and witness size $|w|$.

Our result is quite general and applies not only to standard NIZK proofs, but also to NIZK proofs with stronger security properties such as simulation soundness and non-malleability [Sah01] and

universal composability [Can01]. Universally composable NIZK proofs have the property that they retain their security properties in any environment, even an environment where arbitrary protocols are running concurrently with the NIZK proof. We propose a universally composable NIZK proof that is secure against adaptive malicious adversaries assuming provers are able to securely erase data from their system after constructing their proofs. The universally composable NIZK proofs consist of $|w| + \text{poly}(k)$ bits.

1.3 Our technique

Gentry in his seminal paper [Gen09b] presented a fully homomorphic public-key cryptosystem based on lattices. A fully homomorphic encryption scheme allows taking two ciphertexts and computing new ciphertexts containing the sums or products of their plaintext even if the secret key is unknown. More generally we can take t ciphertexts and compute a new ciphertext containing the evaluation of an arbitrary circuit on the plaintexts. Recent works have aimed at improving the efficiency and proposing fully homomorphic encryption schemes under other assumptions [SV10, vDGHV10, SS10].

There are many applications of fully homomorphic encryption schemes. It is not known whether they imply the existence of NIZK proofs though. However, if NIZK proofs do exist then fully homomorphic encryption can be used to reduce the size of the proofs. Gentry [Gen09a] showed that using fully homomorphic encryption it is possible to get NIZK proofs where the proof size is proportional to the witness size. If we are looking at the satisfiability of a large circuit with a few input gates, i.e., a small witness for satisfiability, this is a significant improvement over other NIZK proofs that tend to grow proportionally to the circuit size.

Gentry proposed to encrypt every bit of the witness using a fully homomorphic encryption scheme. Using the operations of the fully homomorphic cryptosystem it is then possible to evaluate the circuit on the plaintexts to get a ciphertext that contains the output. Using an NIZK proof the prover then constructs a proof for the public key being valid, the encrypted inputs being valid ciphertexts and the output ciphertext being an encryption of 1. Since the proof contains $|w|$ ciphertexts and $|w|$ proofs of their correctness the total complexity is $|w| \cdot \text{poly}(k)$.

In this paper, we present a simple modification of Gentry's NIZK proof that decreases the proof size to $|w| + \text{poly}(k)$. The idea is to encrypt the witness w using a symmetric key cryptosystem, for instance using a one-time pad with a pseudorandom string, and then use the fully homomorphic encryption both to decrypt the encrypted witness and then evaluate the circuit on the witness.

More precisely, the prover will given a witness w for the satisfiability of a circuit C construct $(u, pk, \bar{s})$, where u is an encryption of w and pk is a public key for the fully homomorphic encryption scheme and $\bar{s}$ is a fully homomorphic encryption of the secret seed s used to construct u. Now the prover gives an NIZK proof for pk being a valid public key, $\bar{s}$ being a valid encryption of a seed s and that after decrypting u using s and evaluating C on the resulting plaintext the output is 1. The length of u is $|w|$ and the polynomially many other components are of size $\text{poly}(k)$ each so the total size of the proof is $|w| + \text{poly}(k)$.

Using the fully homomorphic cryptosystem to first evaluate a symmetric key encryption and then use the resulting value afterwards is applicable in many situations. We apply it to non-interactive zero-knowledge proofs here, but one could for instance also use the technique to get interactive zero-knowledge proofs in the plain model with a communication complexity of $|w| + \text{poly}(k)$. This compares favorably with the current state of the art [KR08, IKOS09] that yield interactive proofs with communication complexity growing proportionally to the witness size or linearly in the circuit size.

2 Preliminaries

Given two functions $f, g : \mathbb{N} \to [0,1]$ we write $f(k) \approx g(k)$ when $|f(k) - g(k)| = O(k^{-c})$ for every constant $c > 0$. We say that f is *negligible* if $f(k) \approx 0$ and that f is *overwhelming* if $f(k) \approx 1$.

We write $y = A(x; r)$ when the algorithm A on input x and randomness r, outputs y. We write $y \leftarrow A(x)$ for the process of picking randomness r at random and setting $y = A(x; r)$. We also write $y \leftarrow S$ for sampling y uniformly at random from the set S.

We will now define non-interactive zero-knowledge proofs and describe three tools that will be used in our constructions of minimal size NIZK proofs, namely fully homomorphic encryption schemes, pseudorandom generators and strong one-time signatures.

2.1 Fully homomorphic public-key encryption

A fully homomorphic bit-encryption scheme enables computation on encrypted bits. There is an evaluation algorithm Eval that takes as input an arbitrary Boolean circuit and an appropriate number of ciphertexts and outputs a new ciphertext containing the output of the circuit evaluated on the plaintexts.

The cryptosystem consists of four algorithms $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$. The probabilistic polynomial time key generation algorithm K_{FHE} on input 1^k (and randomness $\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}}(k)}$) outputs a public key pk and a decryption key dk. The probabilistic polynomial time encryption algorithm E given a public key pk and a bit b outputs a ciphertext c. The deterministic polynomial time decryption algorithm D given a public key pk and a ciphertext c returns a bit b or an error symbol $\perp$. Finally, the deterministic polynomial time evaluation algorithm Eval takes a public key pk, a Boolean circuit C with t input gates, and t ciphertexts as input and returns a ciphertext. We require that the cryptosystem is compact, which means that there is a polynomial upper bound $\ell_C(k)$ on the size of the ciphertexts output by Eval.

We will often encrypt an entire bit-string one bit at a time. We therefore define $E_{pk}(m)$ to be the tuple $(E_{pk}(m_1), \ldots, E_{pk}(m_{|m|}))$, where $m_1, \ldots, m_{|m|}$ are the bits of m. When being explicit about the randomness used, we define $E_{pk}(m; \bar{r}) = (E_{pk}(m_1; r_1), \ldots, E_{pk}(m_{|m|}; r_{|m|}))$ for $\bar{r} = (r_1, \ldots, r_{|m|}) \in (\{0,1\}^{\ell_E(k)})^{|m|}$.

The properties we need from the fully homomorphic encryption scheme is correctness and indistinguishability under chosen plaintext attack as defined below.

Definition 1 (Correctness) $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ *is (perfectly) correct if for all Boolean circuits C and all inputs m of appropriate length we have*

$$\Pr\left[(pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k); \bar{m} \leftarrow E_{pk}(m); v = \mathrm{Eval}_{pk}(C; \bar{m}) : D_{dk}(v) = C(m)\right] = 1.$$

Definition 2 (IND-CPA security) $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ *is indistinguishable under chosen plaintext attack (IND-CPA secure) if for all non-uniform polynomial time $\mathcal{A}$ we have*

$$\Pr\left[(pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k); c \leftarrow E_{pk}(0) : \mathcal{A}(c) = 1\right] \approx \Pr\left[(pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k); c \leftarrow E_{pk}(1) : \mathcal{A}(c) = 1\right].$$

2.2 Pseudorandom generators

A length-flexible pseudorandom generator is a deterministic polynomial time algorithm G that on input (s, ℓ) where $s \in \{0,1\}^{\ell_G(k)}$ for a polynomial ℓ_G specified in the description of G returns an ℓ-bit string. Pseudorandomness means that G's output looks random, which we now define formally.

Definition 3 (Pseudorandom generator) *G is a pseudorandom generator if for all non-uniform polynomial time $\mathcal{A}$ and all polynomially bounded ℓ we have*

$$\Pr\left[s \leftarrow \{0,1\}^{\ell_G(k)}; y = G(s, \ell(k)) : \mathcal{A}(y) = 1\right] \approx \Pr\left[y \leftarrow \{0,1\}^{\ell(k)} : \mathcal{A}(y) = 1\right].$$

Length-flexible pseudorandom generators can be constructed from one-way functions [HILL99]. The existence of fully homomorphic encryption therefore implies the existence of length-flexible pseudorandom generators.

2.3 Strong one-time signatures

A strong one-time signature scheme consists of three algorithms $(K_{\mathrm{SIG}}, \mathrm{Sign}, \mathrm{Vfy})$. The key generation algorithm K_{SIG} is a probabilistic polynomial time algorithm that on input 1^k returns a verification key vk and a signing key sk. The signing algorithm Sign is a probabilistic polynomial time algorithm that on input sk and an arbitrary message m returns a signature sig. The signature verification algorithm Vfy is a deterministic polynomial time algorithm that given a verification key vk, a message m and a signature sig returns 1 (acceptance) or 0 (rejection). We require that the scheme is correct and strongly existentially unforgeable under a single chosen message attack as defined below.

Definition 4 (Correctness) *$(K_{\mathrm{SIG}}, \mathrm{Sign}, \mathrm{Vfy})$ is (perfectly) correct if for all $m \in \{0,1\}^*$ we have*

$$\Pr\left[(vk, sk) \leftarrow K_{\mathrm{SIG}}(1^k); \mathrm{sig} \leftarrow \mathrm{Sign}_{sk}(m) : \mathrm{Vfy}_{vk}(m, \mathrm{sig}) = 1\right] = 1.$$

Definition 5 (Strong existential unforgeability under one-time chosen message attack) *$(K_{\mathrm{SIG}}, \mathrm{Sign}, \mathrm{Vfy})$ is strongly existentially unforgeable under chosen message attack if for all non-uniform polynomial time $\mathcal{A}$ we have*

$$\Pr\Big[(vk, sk) \leftarrow K_{\mathrm{SIG}}(1^k); m \leftarrow \mathcal{A}(vk); \mathrm{sig} \leftarrow \mathrm{Sign}_{sk}(m); (m', \mathrm{sig}') \leftarrow \mathcal{A}(\mathrm{sig}) : (m', \mathrm{sig}') \neq (m, \mathrm{sig}) \ \wedge \ \mathrm{Vfy}_{vk}(m', \mathrm{sig}') = 1\Big] \approx 0.$$

We will use a strong one-time signature scheme that has fixed-length signatures, i.e., where there is a polynomial upper bound ℓ_{SIG} on the length of the signatures.

Fixed-length strong one-time signatures can be constructed from one-way functions (from universal one-way hash-functions and Lamport signatures used in combination with Merkle trees for instance). The existence of fully homomorphic encryption therefore implies the existence of fixed-length strong one-time signatures.

2.4 Non-interactive Zero-Knowledge Proofs

Let R be a polynomial time computable binary relation. For pairs $(x, w) \in R$ we call x the statement and w the witness. Let L be the NP-language consisting of statements with witnesses in R.

We will construct NIZK proofs that have almost the same size as the witnesses. The proofs therefore leak the length of the witnesses, so we will assume that given $x \in L$ all witnesses have the same (efficiently computable given x) length. There is only little loss of generality here, since by definition of NP all witnesses have length polynomial in $|x|$ and an appropriate amount of padding could be used to ensure that all witnesses have the same length. We note that most popular NP-complete languages such as SAT, Circuit Satisfiability and Hamiltonicity for instance do indeed have statements that uniquely determine the length of potential witnesses.

An efficient-prover non-interactive zero-knowledge proof for the relation R consists of three probabilistic polynomial time algorithms (K, P, V). K is the common reference string generator that takes the security parameter written in unary 1^k and outputs a common reference string σ. P is the prover algorithm that takes as input the common reference string σ, a statement x and a witness w so $(x, w) \in R$ and outputs a proof π. V is the verifier algorithm that on a common reference string σ, a statement x and a proof π outputs 0 or 1. We interpret a verifier output of 0 as a rejection of the proof and a verifier output of 1 as an acceptance of the proof.

Definition 6 *(K, P, V) is a non-interactive zero-knowledge proof for R if it is complete, sound and zero-knowledge as described below.*

Perfect completeness. Completeness means that a prover with a witness can convince the verifier. For all adversaries $\mathcal{A}$ we have

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

Statistical soundness. Soundness means that it is impossible to convince the verifier of a false statement. For all adversaries $\mathcal{A}$ we have

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

If the probability is exactly 0, we say (K, P, V) is perfectly sound.

Computational zero-knowledge. (K, P, V) is zero-knowledge if it is possible to simulate the proof of a true statement without knowing the witness. Formally, we require the existence of a probabilistic polynomial time simulator $S = (S_1, S_2)$. S_1 outputs a simulated common reference string σ and a simulation trapdoor τ. S_2 takes the simulation trapdoor and a statement as input and produces a simulated proof π. We require for all non-uniform polynomial time adversaries $\mathcal{A}$ that

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

where $P(\cdot, \cdot)$ on input $(x, w) \in R$ returns $\pi \leftarrow P(\sigma, x, w)$ and $S(\cdot, \cdot)$ on input $(x, w) \in R$ returns $\pi \leftarrow S_2(\tau, x)$.

3 Minimal NIZK Proofs from Fully Homomorphic Encryption

We will now construct an NIZK proof system for an arbitrary NP-relation R. The common reference string has length $\text{poly}(k)$ and the proof for a statement x has size $|w| + \text{poly}(k)$, where $|w|$ is the size of witnesses for x. This is likely to be close to the optimal size for both the common reference string and the proofs [GH98].

As we explained in the introduction, the idea in the proof system is to use a pseudorandom one-time pad to encrypt the witness as $u = w \oplus G(s, |w|)$. This ciphertext has length $|w|$. Using a fully homomorphic encryption scheme the prover encrypts the seed s for the one-time pad. Both the prover and the verifier can use the evaluation algorithm to compute a fully homomorphic encryption of $R(x, u \oplus G(s, |w|))$. The prover gives a NIZK proof for the key for the fully homomorphic cryptosystem having been correctly generated, that a seed s has been correctly encrypted and that the resulting encryption of $R(x, u \oplus G(s, |w|))$ decrypts to 1. The fully homomorphic encryption part and the NIZK proof have size polynomial in k, independently of the sizes of $|w|$ or $|x|$. The total size of the proof is therefore $|w| + \text{poly}(k)$.

In order to make this more precise, define given a relation R and a pseudorandom generator G the deterministic polynomial time computable function f that takes as input the security parameter k, a statement x and a string u of length $|w|$ and outputs a Boolean circuit $C_{x,u}$ with $\ell_G(k)$ input wires such that $C_{x,u}(\cdot) = R(x, u \oplus G(\cdot, |w|))$. Define also given a fully homomorphic cryptosystem $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ the relation

$$R^F = \{((pk, \bar{s}, v), (\rho, s, \bar{r})) : \\ \rho \in \{0,1\}^{\ell_{K_{\mathrm{FHE}}}(k)} \wedge (pk, dk) = K_{\mathrm{FHE}}(1^k; \rho) \wedge \bar{r} \in (\{0,1\}^{\ell_E(k)})^{|s|} \wedge \bar{s} = E_{pk}(s; \bar{r}) \wedge D_{dk}(v) = 1\}.$$

Let (K^F, P^F, V^F) be an NIZK proof system for R^F. We can now give the detailed specification of the NIZK proof for R in Figure 1.

$\mathbf{K}(1^k)$ Return $\sigma \leftarrow K^F(1^k)$	$\mathbf{P}(\sigma, x, w)$ $s \leftarrow \{0,1\}^{\ell_G(k)}$ $u = w \oplus G(s, \lvert w\rvert)$ $C_{x,u} = f(k, x, u)$ $\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}}(k)}$ $(pk, dk) = K_{\mathrm{FHE}}(1^k; \rho)$ $\bar{r} \leftarrow (\{0,1\}^{\ell_E(k)})^{\ell_G(k)}$ $\bar{s} = E_{pk}(s; \bar{r})$ $v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$ $\pi \leftarrow P^F(\sigma, (pk, \bar{s}, v), (\rho, s, \bar{r}))$ Return $\Pi = (pk, \bar{s}, u, \pi)$	$\mathbf{S}_1(1^k)$ Return $(\sigma, \tau) \leftarrow S_1^F(1^k)$
$\mathbf{V}(\sigma, x, \Pi)$ Parse $\Pi = (pk, \bar{s}, u, \pi)$ $C_{x,u} = f(k, x, u)$ $v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$ Return $V^F(\sigma, (pk, \bar{s}, v), \pi)$		$\mathbf{S}_2(\tau, x)$ $u \leftarrow \{0,1\}^{\lvert w\rvert}$ $C_{x,u} = f(k, x, u)$ $(pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k)$ $\bar{s} \leftarrow E_{pk}(0^{\lvert w\rvert})$ $v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$ $\pi \leftarrow S_2^F(\tau, (pk, \bar{s}, v))$ Return $\Pi = (pk, \bar{s}, u, \pi)$

Figure 1: NIZK proof system for R.

Theorem 7 *(K, P, V) described in Figure 1 is an NIZK proof system for R.*

Proof. Perfect completeness follows from the perfect completeness of (K^F, P^F, V^F) and the perfect correctness of the fully homomorphic cryptosystem. The prover generates a valid key pair (pk, dk), makes valid encryptions of the bits in s and by the correctness of the fully homomorphic cryptosystem v decrypts to 1 provided $w = u \oplus G(s, |w|)$ is a witness for x. The statement and witness provided to P^F is therefore valid and the completeness of (K^F, P^F, V^F) implies that the resulting proof π is acceptable.

Statistical soundness follows from the statistical soundness of (K^F, P^F, V^F) and the correctness of the fully homomorphic cryptosystem. To see this, consider a proof $\Pi = (pk, \bar{s}, u, \pi)$ for a statement x. By the statistical soundness of π there exists ρ such that $(pk, dk) = K_{\mathrm{FHE}}(1^k; \rho)$ and $v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$ decrypts to 1 under dk. Furthermore, the statistical soundness also guarantees that $\bar{s} = E_{pk}(s; \bar{r})$ for some seed s and randomness $\bar{r}$. The perfect correctness of the fully homomorphic cryptosystem now guarantees that $w = u \oplus G(s, |w|)$ is a witness for $x \in L$.

Computational zero-knowledge follows from the computational zero-knowledge of (K^F, P^F, V^F), the pseudorandomness of G and the IND-CPA security of $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$. Figure 1 describes a zero-knowledge simulator (S_1, S_2) and we will now show that for every non-uniform polynomial time $\mathcal{A}$

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

where $P(\cdot, \cdot)$ on input $(x, w) \in R$ returns $P(\sigma, x, w)$ and $S(\cdot, \cdot)$ on input $(x, w) \in R$ returns $S_2(\tau, x)$.

Consider generating the common reference string using $(\sigma, \tau) \leftarrow S_1^F(1^k)$ instead of using $K = K^F$ and consider a modified oracle $P'(\cdot, \cdot)$ that on $(x, w) \in R$ returns a proof $\Pi = (pk, \bar{s}, u, \pi)$ generated as a normal prover $P(\sigma, x, w)$ would do except simulating $\pi \leftarrow S_2(\tau, x)$. By the zero-knowledge property of (K^F, P^F, V^F) we have for all non-uniform polynomial time $\mathcal{A}$

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

Let P'' be a modification of P' where the responses $\Pi = (pk, \bar{s}, u, \pi)$ are generated by computing $\bar{s} \leftarrow E_{pk}(0^{\ell_G(k)})$. By the IND-CPA security of $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ we have for all non-uniform polynomial time $\mathcal{A}$

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

Finally, since $S_1 = S_1^F$ we can view S as a modification of P'' where the responses $\Pi = (pk, \bar{s}, u, \pi)$ are generated such that $u \leftarrow \{0,1\}^{|w|}$. By the pseudorandomness of G we have for all non-uniform polynomial time $\mathcal{A}$

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

We conclude that (S_1, S_2) is a zero-knowledge simulator for (K, P, V). □

The transformation preserves many properties of the underlying NIZK proof (K^F, P^F, V^F). If K^F outputs uniformly random common reference strings, then so does K. If the underlying NIZK proof has perfect soundness, then so does (K, P, V). If the underlying NIZK proof is a proof of knowledge, i.e., given a secret extraction key ξ related to the common reference string it is possible to extract the witness, then so is the resulting witness-length NIZK proof.

4 Universally Composable NIZK Proofs from Fully Homomorphic Encryption

The universal composability (UC) framework [Can01] enables strong security statements of the form that a protocol ϕ securely realizes an ideal functionality $\mathcal{F}$, which implies that ϕ is secure even if it is running in an environment where arbitrary other protocols are running concurrently with ϕ. The ideal functionality specifies the ideal security properties of the protocol. The parties use a secure channel to forward their inputs to the ideal functionality that computes each party's outputs. The universal composition theorem that says if a protocol $\phi^{\mathcal{F}'}$ securely realizes an ideal functionality $\mathcal{F}$ in an $\mathcal{F}'$-hybrid model where it can make calls to an ideal functionality $\mathcal{F}'$, then for any protocol ψ securely realizing $\mathcal{F}'$ we have that ϕ^{ψ} securely realizes $\mathcal{F}$.

We are interested in securely realizing the ideal non-interactive zero-knowledge functionality $\mathcal{F}_{\mathrm{NIZK}}^R$ described in Figure 2. The session ids *sid* are used to distinguish different invocations of the same functionality, and the proof ids *pid* are used to distinguish different queries to the functionality. The functionality allows a prover to compute a proof π for a statement x if it has a witness w such that $(x, w) \in R$ and will always verify such proofs as being correct. However, the proof π is computed without any knowledge of the witness w and therefore the functionality captures an ideal notion of zero-knowledge. The ideal functionality also captures an ideal form of soundness, since the only way a proof π for a statement x can be accepted is if at some point a witness w such that $(x, w) \in R$ has been provided to the ideal functionality.

Ideal NIZK Proof Functionality $\mathcal{F}_{\text{NIZK}}^{R}$

Parameterized with NP-relation R and running with parties $\tilde{P}_1, \ldots, \tilde{P}_n$ and adversary $\mathcal{S}$.

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

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

Figure 2: Ideal NIZK proof functionality $\mathcal{F}_{\text{NIZK}}^{R}$.

Let us clarify what it means to securely realize $\mathcal{F}_{\text{NIZK}}^{R}$. We will construct a protocol ϕ to be run by parties $P_1, \ldots, P_n$ that receive protocol inputs from the environment and make outputs to the environment in which they are operating. We model the environment as a non-uniform polynomial time algorithm $\mathcal{Z}$. The execution of the protocol itself is attacked by a non-uniform polynomial time adversary $\mathcal{A}$ that may communicate with the environment and corrupt parties adaptively. When corrupting a party P_i the adversary learns the present state of the party and takes control over the actions of P_i.

We say the protocol ϕ securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ if there is a simulator $\mathcal{S}$ that can simulate the protocol execution on top of the ideal functionality $\mathcal{F}_{\text{NIZK}}^{R}$. The simulator $\mathcal{S}$ runs with dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$ that instead of running ϕ simply forward their inputs to the ideal functionality $\mathcal{F}_{\text{NIZK}}^{R}$ and return the responses from $\mathcal{F}_{\text{NIZK}}$ to the environment. The simulator $\mathcal{S}$ has the same ability as $\mathcal{A}$ to corrupt dummy parties and to communicate with the environment, but does not have access to the internal workings of the ideal execution taking place inside $\mathcal{F}_{\text{NIZK}}$.

Formally, it is said ϕ securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ if for any non-uniform polynomial time adversary $\mathcal{A}$ there is a non-uniform polynomial time simulator $\mathcal{S}$ such that no non-uniform polynomial time environment can distinguish between ϕ executed by real parties $P_1, \ldots, P_n$ under attack by $\mathcal{A}$ and $\mathcal{F}_{\text{NIZK}}^{R}$ being used by dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$ in the simulation by $\mathcal{S}$. There are known examples of protocols securely realizing $\mathcal{F}_{\text{NIZK}}$ for Circuit Satisfiability in the common reference string model [DDO+02, Gro06, GOS06b] and in the multi-string model [GO07]. Also a related functionality has been securely realized in the registered public key model [BCNP04].

We will now present a non-interactive protocol ϕ that securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ for an arbitrary NP-relation R with minimal communication. A proof for a statement x with witnesses of size $|w|$ consists of $|w| + \text{poly}(k)$ bits. We make two assumptions, namely that a fully homomorphic encryption scheme $(K_{\text{FHE}}, E, D, \text{Eval})$ exists and that $\mathcal{F}_{\text{NIZK}}^{R^F}$ can be securely realized for the relation

$$R^F = \{((pk, \bar{s}, v, vk), (\rho, s, \bar{r})) : \\ \rho \in \{0,1\}^{\ell_{K_{\text{FHE}}}(k)} \wedge (pk, dk) = K_{\text{FHE}}(1^k; \rho) \wedge \bar{r} \in (\{0,1\}^{\ell_E(k)})^{|s|} \wedge \bar{s} = E_{pk}(s; \bar{r}) \wedge D_{dk}(v) = 1\}.$$

Note that we have generalized R^F slightly compared to the previous section by allowing statements to have an arbitrary vk in the end.

The construction is quite similar to the one in the previous section except the prover will make a strong one-time signature on each proof in order to prevent modifications. We therefore proceed directly to giving the details of the protocol in Figure 3.

Universally Composable NIZK Protocol

P_i on $(\mathbf{prove}, sid, pid, x, w)$

Ignore if $(x, w) \notin R$
$s \leftarrow \{0,1\}^{\ell_G(k)}$
$u = w \oplus G(s, |w|)$
$C_{x,u} = f(k, x, u)$
$\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}}(k)}$
$(pk, dk) = K_{\mathrm{FHE}}(1^k; \rho)$
$\bar{r} \leftarrow (\{0,1\}^{\ell_E(k)})^{\ell_G(k)}$
$\bar{s} = E_{pk}(s; \bar{r})$
$v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$
$(vk, sk) \leftarrow K_{\mathrm{SIG}}(1^k)$
Run $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$ using input
  $(\mathbf{prove}, sid, pid, (pk, \bar{s}, v, vk), (\rho, s, \bar{r}))$
  and immediately deleting $dk, \rho, s, \bar{r}$
Wait for answer $(\mathbf{proof}, sid, pid, \pi)$
$\mathrm{sig} \leftarrow \mathrm{Sign}_{sk}(x, pk, \bar{s}, u, vk, \pi)$
Return $(\mathbf{proof}, sid, pid, (pk, \bar{s}, u, vk, \pi, \mathrm{sig}))$
  after deleting all other data

P_j on $(\mathbf{verify}, sid, pid, x, (pk, \bar{s}, u, vk, \pi, \mathrm{sig}))$

Check $\mathrm{Vfy}_{vk}(x, pk, \bar{s}, u, vk, \pi) = 1$
$C_{x,u} = f(k, x, u)$
$v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$
Run $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$ using input
  $(\mathbf{verify}, sid, pid, (pk, \bar{s}, v, vk), \pi)$
Wait for answer
  $(\mathbf{verification}, sid, pid, b)$
Check $b = 1$
If all checks pass return
  $(\mathbf{verification}, sid, pid, 1)$
Else return $(\mathbf{verification}, sid, pid, 0)$

Figure 3: Universally composable NIZK proof for R.

Theorem 8 *The protocol ϕ in Figure 3 securely realizes $\mathcal{F}_{\mathrm{NIZK}}^{R}$ in the $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$-hybrid model.*

Proof. We have to show that for any adversary $\mathcal{A}$ there is an ideal process adversary $\mathcal{S}$ such that no environment $\mathcal{Z}$ has more than negligible advantage in distinguishing between ϕ running with $P_1, \ldots, P_n$ and $\mathcal{A}$ and $\mathcal{F}_{\mathrm{NIZK}}^{R}$ running with dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$ and $\mathcal{S}$. Our proof strategy is to start with ϕ running with $\mathcal{A}$ and modifying the experiment in steps that the environment has negligible probability of distinguishing. For this purpose we define three additional simulators $\mathcal{S}_{\mathrm{REAL}}, \mathcal{S}_{\mathrm{EXT}}, \mathcal{S}_{\mathrm{SIM}}$ that are used in intermediate steps and have the ability to control $\mathcal{F}_{\mathrm{NIZK}}^{R}$ in various ways. Informally, $\mathcal{S}_{\mathrm{REAL}}$ running with the ideal functionality $\mathcal{F}_{\mathrm{NIZK}}^{R}$ takes full control over $\mathcal{F}_{\mathrm{NIZK}}^{R}$ and makes a perfect simulation of $\mathcal{A}$ running with ϕ. $\mathcal{S}_{\mathrm{EXT}}$ modifies the simulation such that whenever an NIZK proof that has not been created by $\mathcal{F}_{\mathrm{NIZK}}$ is verified as being valid it extracts the corresponding witness and inputs it to $\mathcal{F}_{\mathrm{NIZK}}^{R}$. $\mathcal{S}_{\mathrm{SIM}}$ and $\mathcal{S}$ complete the security proof by enabling the simulation of honest parties making NIZK proofs without knowledge of the corresponding witnesses.

We now give the details of the simulators and the security proof.

$\mathcal{S}_{\mathrm{REAL}}$: $\mathcal{S}_{\mathrm{REAL}}$ learns the inputs to $\mathcal{F}_{\mathrm{NIZK}}^{R}$ and controls the outputs. It can therefore run a perfect simulation of $P_1, \ldots, P_n$ and $\mathcal{A}$ running π_{NIZK} in the $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$-hybrid model.

$\mathcal{S}_{\mathrm{REAL}}$ simulates $\mathcal{A}$ and forwards all communication between the simulated $\mathcal{A}$ and the environment $\mathcal{Z}$. Whenever the simulated $\mathcal{A}$ corrupts a simulated P_i, $\mathcal{S}_{\mathrm{REAL}}$ corrupts $\tilde{P}_i$ and lets it interact with the environment as $\mathcal{A}$ instructs the simulated P_i to interact with the environment.

When $\mathcal{S}_{\mathrm{REAL}}$ receives $(\mathbf{prove}, P_i, sid, pid, x)$ from $\mathcal{F}_{\mathrm{NIZK}}^{R}$ it is because an honest $\tilde{P}_i$ has input $(\mathbf{prove}, sid, pid, x, w)$ with $(x, w) \in R$. Since $\mathcal{S}_{\mathrm{REAL}}$ knows the inputs to $\mathcal{F}_{\mathrm{NIZK}}^{R}$ it can simulate P_i running π_{NIZK} in the $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$-hybrid model including $\mathcal{F}_{\mathrm{NIZK}}^{R^F}$ sending $(\mathbf{prove}, (pk, \bar{s}, u, vk))$

to $\mathcal{A}$ and on getting the answer $(\mathbf{proof}, \pi)$ making the signature sig to complete the proof $\Pi = (pk, \bar{s}, u, vk, \pi, \text{sig})$. $\mathcal{S}_{\text{REAL}}$ answers $(\mathbf{proof}, \Pi)$ to $\mathcal{F}_{\text{NIZK}}^{R}$.

On input $(\mathbf{verify}, P_j, sid, pid, x, \Pi)$ from $\mathcal{F}_{\text{NIZK}}^{R}$ the simulator $\mathcal{S}_{\text{REAL}}$ knows that the honest party $\tilde{P}_j$ has queried $(\mathbf{verify}, sid, pid, x, \Pi)$ to $\mathcal{F}_{\text{NIZK}}^{R}$, where (x, Π) has not been stored before and hence not been created by an honest party. $\mathcal{S}_{\text{REAL}}$ simulates P_j running the verification protocol on input $(\mathbf{verify}, sid, pid, x, \Pi)$. The simulator forces $\mathcal{F}_{\text{NIZK}}^{R}$ to return the resulting answer $(\mathbf{verification}, sid, pid, b)$ and stores (x, Π) in $\mathcal{F}_{\text{NIZK}}^{R}$ if $b = 1$.

The simulation is exactly like running π_{NIZK} in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model, except for the fact that a proof Π for a statement x output by an honest party $\tilde{P}_i$ is guaranteed to be accepted in the verification phase and once a proof Π for a statement x is accepted, it will always be accepted by $\mathcal{F}_{\text{NIZK}}^{R}$. However, if we look at a real execution of π_{NIZK} we see that the correctness of the signature scheme, the correctness of the fully homomorphic cryptosystem and the properties of $\mathcal{F}_{\text{NIZK}}^{R^F}$ guarantees that proofs Π created by honest parties P_i are accepted and also that accepted proofs will always be accepted again. To the environment, a real execution of π_{NIZK} in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model with adversary $\mathcal{A}$ is perfectly indistinguishable from the simulation by $\mathcal{S}_{\text{REAL}}$ running with $\mathcal{F}_{\text{NIZK}}^{R}$.

$\mathcal{S}_{\text{EXT}}$**:** $\mathcal{S}_{\text{EXT}}$ runs like $\mathcal{S}_{\text{REAL}}$ when proofs are constructed, but changes the way proofs are verified. As $\mathcal{S}_{\text{REAL}}$ it simulates P_j getting input $(\mathbf{verify}, sid, pid, x, \Pi)$ in the execution of π_{NIZK}, but if the answer is $(\mathbf{verification}, sid, pid, 1)$ then it extracts a witness w such that $(x, w) \in R$ and aborts the simulation if the extraction fails.

More precisely, on input $(\mathbf{verify}, P_j, sid, pid, x, \Pi)$ from $\mathcal{F}_{\text{NIZK}}^{R}$ the simulator $\mathcal{S}_{\text{EXT}}$ simulates the honest P_j getting input $(\mathbf{verify}, sid, pid, x, \Pi)$ in π_{NIZK}. If P_j outputs $(\mathbf{verification}, sid, pid, 0)$ then $\mathcal{S}_{\text{EXT}}$ returns $(\mathbf{witness}, \perp)$ to $\mathcal{F}_{\text{NIZK}}^{R}$ and forwards the resulting $(\mathbf{verification}, sid, pid, 0)$ message to $\tilde{P}_j$. On the other hand, if P_j outputs $(\mathbf{verification}, sid, pid, 1)$ then $\mathcal{S}_{\text{EXT}}$ will try to extract a witness w such that $(x, w) \in R$, return $(\mathbf{witness}, w)$ to $\mathcal{F}_{\text{NIZK}}^{R}$ and send the resulting $(\mathbf{verification}, sid, pid, 1)$ message to $\tilde{P}_j$.

$\mathcal{S}_{\text{EXT}}$ parses $\Pi = (pk, \bar{s}, u, vk, \pi, \text{sig})$. We only need to extract a witness for Π, where the signature sig on $(x, pk, \bar{s}, u, vk, \pi)$ is valid, because otherwise the protocol π_{NIZK} will reject the proof. Part of the verification protocol also consists in querying $\mathcal{F}_{\text{NIZK}}^{R^F}$ on $(\mathbf{verify}, sid, pid, (pk, \bar{s}, v, vk), \pi)$, where $v = \text{Eval}_{pk}(C_{x,u}, \bar{s})$. The simulated $\mathcal{F}_{\text{NIZK}}^{R^F}$ will only return $(\mathbf{verification}, sid, pid, 1)$ if an honest party created the proof π on $(pk, \bar{s}, v, vk)$ or if $\mathcal{A}$ supplies a witness $(\rho, s, \bar{r})$ such that $(pk, dk) = K_{\text{FHE}}(1^k; \rho)$ and $\bar{s} = E_{pk}(s; \bar{r})$ and $D_{dk}(v) = 1$. In the latter case, this tells $\mathcal{S}_{\text{EXT}}$ what dk is and hence it can compute s and $w = u \oplus G(s, |w|)$. The correctness of the fully homomorphic cryptosystem shows that in this case, the witness w satisfies $(x, w) \in R$ and therefore $\mathcal{S}_{\text{EXT}}$ can submit $(\mathbf{witness}, w)$ to $\mathcal{F}_{\text{NIZK}}^{R}$. On the other hand, if an honest party created the proof π on $(pk, \bar{s}, v, vk)$ then the strong existential unforgeability of the one-time signature scheme implies that there is negligible probability of the adversary producing a different valid signature sig using vk. There is therefore only negligible risk of $\mathcal{S}_{\text{EXT}}$ not being able to extract a witness w.

$\mathcal{S}_{\text{SIM}}$**:** $\mathcal{S}_{\text{EXT}}$ runs the verification process of $\mathcal{F}_{\text{NIZK}}^{R}$ without interference, but in the proof process it uses knowledge of the inputs the honest parties provide to $\mathcal{F}_{\text{NIZK}}^{R}$. In the next couple of modification of the simulator, we will move towards simulating the proofs instead of using knowledge of the inputs to $\mathcal{F}_{\text{NIZK}}^{R}$.

Let $\mathcal{S}_{\text{SIM}}$ be a modification of $\mathcal{S}_{\text{EXT}}$ that instead of running a perfect simulation of $\mathcal{F}_{\text{NIZK}}^{R^F}$ allows simulated honest parties to submit $(\mathbf{prove}, sid, pid, (pk, \bar{s}, v, vk)), w)$ even if $(x, w) \notin R$.

This means, $\mathcal{F}_{\text{NIZK}}^{R^F}$ may ask $\mathcal{A}$ for a proof π for a false statement $(pk, \bar{s}, v, vk)$ and store $((pk, \bar{s}, v, vk), \pi)$ as being a valid proof and return $(\textbf{proof}, sid, pid, (pk, \bar{s}, v, vk), \pi)$ to the requesting party P_i.

$\mathcal{S}_{\text{EXT}}$ can now change the way it constructs $\bar{s}$ to instead set $\bar{s} \leftarrow E_{pk}(0^{\ell_G(k)})$. Due to the IND-CPA security of the fully homomorphic encryption scheme this tuple of ciphertexts $\bar{s}$ is indistinguishable from a bit-wise encryption of s. Running $\mathcal{F}_{\text{NIZK}}^{R}$ with $\mathcal{S}_{\text{SIM}}$ is therefore computationally indistinguishable from running $\mathcal{F}_{\text{NIZK}}^{R}$ with $\mathcal{S}_{\text{EXT}}$.

$\mathcal{S}$: We will now make a modification of $\mathcal{S}_{\text{SIM}}$ to get a simulator that does not have access to the internals of $\mathcal{F}_{\text{NIZK}}^{R}$. $\mathcal{S}$ is a modification of $\mathcal{S}_{\text{SIM}}$ that simulates the proof created by an honest party P_i by setting $u \leftarrow \{0,1\}^{|w|}$ instead of using $u = w \oplus G(s, |w|)$. Since G is a pseudorandom generator, it is not possible for the environment to distinguish whether $\mathcal{S}_{\text{SIM}}$ or $\mathcal{S}$ is making the simulation with $\mathcal{F}_{\text{NIZK}}^{R}$.

Since $\mathcal{S}$ does not need to know the witness when simulating a proof for an honest party P_i, it runs entirely without access to or control over the workings of $\mathcal{F}_{\text{NIZK}}^{R}$. As we have shown $\mathcal{S}$ running with $\mathcal{F}_{\text{NIZK}}^{R}$ is indistinguishable from the protocol π_{NIZK} running with $\mathcal{A}$ in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model. The protocol π_{NIZK} therefore securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model. □

There may be many ways to securely realize $\mathcal{F}_{\text{NIZK}}^{R^F}$ and by the universal composition theorem [Can01] our result shows that all of them imply the existence of witness-length universally composable NIZK proofs if fully homomorphic encryption exists. In particular, we get witness-length universally composable NIZK proofs in the common reference string model [BFM88], the multi-string model [GO07] or under any other setup assumption under which universally composable NIZK proofs exist.

We have assumed a dynamic corruption model in the construction. However, our construction also shows that if $\mathcal{F}_{\text{NIZK}}^{R^F}$ can be securely realized against static adversaries, then we get a witness-length universally composable NIZK proof for any NP-relation R that is secure against static adversaries.

References

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *FOCS*, pages 186–195, 2004.

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

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444, 2006.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP*, volume 4596 of *Lecture Notes in Computer Science*, pages 423–434, 2007.

[Dam92] Ivan Damgård. Non-interactive circuit based proofs and non-interactive perfect zero-knowledge with preprocessing. In *EUROCRYPT*, volume 658 of *Lecture Notes in Computer Science*, 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 *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598, 2002.

[DDP02] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Randomness-optimal characterization of two NP proof systems. In *RANDOM*, volume 2483 of *Lecture Notes in Computer Science*, pages 179–193, 2002.

[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.

[Gen09a] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GH98] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Inf. Process. Lett.*, 67(4):205–214, 1998.

[GK96] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal of Computing*, 25(1):169–192, 1996.

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

[GO07] Jens Groth and Rafail Ostrovsky. Cryptography in the multi-string model. In *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 323–341, 2007.

[GOS06a] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 97–111, 2006.

[GOS06b] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero-knowledge for NP. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358, 2006.

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT*, volume 4248 of *Lecture Notes in Computer Science*, pages 444–459, 2006.

[Gro10] Jens Groth. Short non-interactive zero-knowledge proofs. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 341–358, 2010.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432, 2008.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal of Computating*, 28(4):1364–1396, 1999.

[IKOS09] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM Journal of Computing*, 39(3):1121–1152, 2009.

[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.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive pcp. In *ICALP*, volume 5126 of *Lecture Notes in Computer Science*, pages 536–547, 2008.

[Ore87] Yair Oren. On the cunning power of cheating verifiers: Some observations about zero knowledge proofs. In *FOCS*, pages 462–471, 1987.

[Sah01] Amit Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 2001.

[SS10] Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 377–394, 2010.

[SV10] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443, 2010.

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43, 2010.