Using Fully Homomorphic Hybrid Encryption to Minimize Non-interactive Zero-Knowledge Proofs*

Craig Gentry
IBM T.J. Watson Research Center, U.S.
cbgentry@us.ibm.com

Jens Groth
University College London, U.K.
j.groth@ucl.ac.uk

Yuval Ishai
Technion, Israel
yuvali@cs.technion.ac.il

Chris Peikert
Georgia Institute of Technology, U.S.
cpeikert@cc.gatech.edu

Amit Sahai
University of California Los Angeles, U.S.
sahai@cs.ucla.edu

Adam Smith
Pennsylvania State University, U.S.
asmith@psu.edu

Abstract

A non-interactive zero-knowledge (NIZK) proof can be used to demonstrate the truth of a statement without revealing anything else. It has been shown under standard cryptographic assumptions that NIZK proofs of membership exist for all languages in NP. While there is evidence that such proofs cannot be much shorter than the corresponding membership witnesses, all known NIZK proofs for NP languages are considerably longer than the witnesses.

Soon after Gentry's construction of fully homomorphic encryption, several groups independently contemplated the use of hybrid encryption to optimize the size of NIZK proofs and discussed this idea within the cryptographic community. This article formally explores this idea of using fully homomorphic hybrid encryption to optimize NIZK proofs and other related cryptographic primitives.

We investigate the question of minimizing the communication overhead of NIZK proofs for NP and show that if fully homomorphic encryption exists then it is possible to get proofs that are roughly of the same size as the witnesses.

Our technique consists in constructing a fully homomorphic hybrid encryption scheme with ciphertext size $|m|+\text{poly}(k)$, where m is the plaintext and k is the security parameter. Encrypting the witness for an NP-statement allows us to evaluate the NP-relation in a communication-efficient manner. We apply this technique 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 or to minimize the communication in secure computation protocols.

Keywords: Non-interactive zero-knowledge proofs, fully homomorphic encryption, hybrid encryption, secure function evaluation, minimizing communication.

*©IACR 2014. This article is a minor revision of the version published by Springer- Verlag available at http://dx.doi.org/10.1007/s00145-014-9184-y. The original publication is available at www.springerlink.com.

1 Introduction

Non-interactive zero-knowledge (NIZK) proof systems [BFM88] yield proofs that can convince others about the truth of a statement without revealing anything but this truth. We will consider statements of the form $x \in L$, where L can be an arbitrary language in NP. We require that the NIZK proof be *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 cannot convince the verifier that a false statement is true.

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.

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 encryption schemes [DDN00] to 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, GOS12].

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 [GOS12, 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 whose size depends only on the size of the witness and on the security parameter, but not on the size of the circuit used to verify the witness. However, the ratio between the proof size and the witness size in this construction grows polynomially with the security parameter.

Soon after Gentry's construction of fully homomorphic encryption, several groups independently contemplated the use of hybrid encryption to optimize the size of NIZK proofs and other related cryptographic primitives (e.g. [Ish09, PS09]). This article formally explores the idea of using fully homomorphic encryption to minimize communication.

1.2 Our contribution

We construct NIZK proofs for arbitrary NP-languages in which 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. This is essentially the best one could hope for given our current knowledge on the complexity of deciding membership of NP-languages. Indeed, any interactive proof system for NP in which the communication from the prover to the verifier is substantially smaller than the witness size would imply a breakthrough in complexity theory [GH98, GVW02].

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 [GOS12, Gro10, Gen09a]. All of these NIZK proofs are publicly verifiable given 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 [GOS12]	$\text{poly}(k)$	$\|C\| \cdot \text{poly}(k)$	Pairing-based
Groth [Gro10]	$\|C\| \cdot \text{polylog}(k) + \text{poly}(k)$	$\|C\| \cdot \text{polylog}(k) + \text{poly}(k)$	Naccache-Stern
Gentry [Gen09a]	$\text{poly}(k)$	$\|w\| \cdot \text{poly}(k)$	FHE and NIZK
This work	$\text{poly}(k)$	$\|w\| + \text{poly}(k)$	FHE and NIZK

Table 1: Comparison of NIZK proofs with 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] as well as 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 erase data from their systems after constructing their proofs. The universally composable NIZK proofs are also $|w| + \text{poly}(k)$ bits long.

1.3 Our technique: fully homomorphic hybrid encryption

Our technique is based on a hybrid form of fully homomorphic encryption. A fully homomorphic encryption scheme allows taking two ciphertexts and computing a new ciphertext containing the sum or the product of their plaintexts 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 their plaintexts. Following the breakthrough work of Gentry [Gen09b], several subsequent works improved the efficiency of fully homomorphic encryption and the assumptions on which it can be based [SV10, vDGHV10, SS10, BV11, BGV12, Bra12, BGH13].

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 wires, 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 encryption scheme it is 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 construct a fully homomorphic hybrid encryption scheme. We first encrypt the witness w using a symmetric key encryption scheme, for instance using a one-time pad with a pseudorandom string, and then use the fully homomorphic encryption scheme both to decrypt the symmetrically 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 using a symmetric encryption scheme and pk is a public key for the fully homomorphic encryption scheme and $\bar{s}$ is a fully homomorphic encryption of the secret key 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 key 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 $|w| + \text{poly}(k)$.

Fully homomorphic hybrid encryption is applicable in many situations. We apply it to non-interactive zero-knowledge proofs here, but one could for instance also use it in a similar way to get interactive zero-knowledge proofs in the plain model with a communication complexity of $|w| + \text{poly}(k)$. This compares favorably with the best previous communication-efficient interactive zero-knowledge proofs [KR08, IKOS09, GKR08], in which the communication complexity either grows linearly with the circuit size or is polynomial in the witness size and restricted to low-depth circuits. Following a preliminary version of our work, a similar use of fully homomorphic hybrid encryption has been used in the context of computationally private information retrieval [BV11, Lip11].

An additional application of fully homomorphic hybrid encryption is to minimizing communication in general secure computation. For simplicity, we address here the case of secure two-party computation in the semi-honest model (i.e., when the parties are honest-but-curious). Concretely, consider the case where Alice and Bob hold inputs x and y respectively and want to compute $f(x, y)$ for a given polynomial time computable function f. Using a fully homomorphic hybrid encryption scheme, there is a simple 2-party protocol to accomplish this task that is secure against static honest-but-curious adversaries: Alice encrypts x under her own key and Bob using the fully homomorphic property evaluates the function $f(\cdot, y)$ on the encrypted x, which Alice can then decrypt. The communication of this protocol is $|x| + \text{poly}(k) \cdot |f(x, y)|$ bits, which is optimal for functions $f(x, y)$ with a small output, such as the function corresponding to Yao's millionaires problem [Yao82]. Using specific fully homomorphic encryption schemes [BGV12, Lip11] the communication can be reduced further to be $|x| + |f(x, y)| \cdot (1 + o(1)) + \text{poly}(k)$, which is useful when the output is large. We note that this approach can also be applied in the multi-party case via the use of threshold fully homomorphic encryption [Gen09b], and that security against malicious parties can be obtained without increasing the asymptotic communication complexity by using sublinear-communication arguments (cf. [NN01]).

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 encryption scheme consists of four algorithms $(K_{\text{FHE}}, E, D, \text{Eval})$. The probabilistic polynomial time key generation algorithm K_{FHE} on input 1^k (and randomness $\rho \leftarrow \{0,1\}^{\ell_{K_{\text{FHE}}}(k)}$) 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 (and randomness $r \leftarrow \{0,1\}^{\ell_E(k)}$) 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 and t ciphertexts as input and returns a ciphertext. We require that the encryption scheme be *compact,* which means that there is a polynomial upper bound $\ell_{\text{Eval}}(k)$ on the size of the ciphertexts output by Eval, which is independent of the size of the circuit C.

We will often encrypt a 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 are correctness and indistinguishability under chosen plaintext attack as defined below.

Definition 1 (Correctness) $(K_{\text{FHE}}, E, D, \text{Eval})$ *is (perfectly) correct if for all all inputs* $m \in \{0,1\}^*$ *and all Boolean circuits* C *with* $|m|$ *input bits*

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

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

$$\Pr\Big[(pk, dk) \leftarrow K_{\text{FHE}}(1^k); b \leftarrow \{0,1\}; c \leftarrow E_{pk}(b) : \mathcal{A}(c) = b\Big] \approx \frac{1}{2}.$$

Please note that by a standard hybrid argument the above security definition implies IND-CPA security also when using E_{pk} for a bit-by-bit encryption of an arbitrary polynomial-size message m.

2.2 Pseudorandom generators

A length-flexible pseudorandom generator is a deterministic polynomial time algorithm G that on input (s, ℓ), where $s \in \{0,1\}^k$, 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*

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

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 be correct and *strongly* existentially unforgeable under a single chosen message attack, where "strongly" means that the adversary is not even allowed to obtain a different signature on the same message. We give the formal definition below.

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

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

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

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

We will use a strong one-time signature scheme that has fixed-length signatures, i.e., where there is a polynomial upper bound $\ell_{\mathrm{SIG}}(k)$ 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 [NY90] and Lamport signatures used in combination with Merkle trees [Rom90]). 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 polynomially bounded, 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 $L = \{\ x\ : \exists w\ (x, w) \in R\ \}$. We write $R(x, w) \in \{0,1\}$ (0 is no, 1 is yes) for the output of the polynomial time decision algorithm for R on input (x, w).

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 for all $x \in L$ of the same length, all witnesses have the same length which can be efficiently computed given $1^{|x|}$. 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 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.[1] P is the prover algorithm that takes as input the common reference string σ, a statement x and a witness w such that $(x, w) \in R$ 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 $\mathcal{A}$

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

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

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

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 $\mathcal{A}$

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

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.

As 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 pseudorandom 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 an NIZK proof for the key for

[1] Our constructions can be instantiated with a uniformly distributed σ, but other distributions are useful for broadening the class of intractability assumptions on which our results can be based.

the fully homomorphic encryption scheme 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 $|w| + \mathrm{poly}(k)$.

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 k input wires such that $C_{x,u}(\cdot) = R(x, u \oplus G(\cdot, |w|))$. Define also given a fully homomorphic encryption scheme $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ the relation

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

Let (K^F, P^F, V^F) be an NIZK proof system for R^F with zero-knowledge simulator (S_1^F, S_2^F). We can now give the detailed specification of the NIZK proof for R in Figure 1.

```
$\mathbf{K}(1^k)$
Return $\sigma \leftarrow K^F(1^k)$

$\mathbf{V}(\sigma, x, \Pi)$
Parse $\Pi = (pk, \bar{s}, u, \pi)$
$C_{x,u} = f(1^k, x, u)$
$v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$
Return $V^F(\sigma, (pk, \bar{s}, v), \pi)$
```

```
$\mathbf{P}(\sigma, x, w)$
$s \leftarrow \{0,1\}^k$
$u = w \oplus G(s, |w|)$
$C_{x,u} = f(1^k, x, u)$
$\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}}(k)}$
$(pk, dk) = K_{\mathrm{FHE}}(1^k; \rho)$
$\bar{r} \leftarrow (\{0,1\}^{\ell_E(k)})^k$
$\bar{s} = E_{pk}(s; \bar{r})$
$v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$
$\pi \leftarrow P^F(\sigma, (pk, \bar{s}, v), (\rho, s, \bar{r}))$
Return $\Pi = (pk, \bar{s}, u, \pi)$
```

```
$\mathbf{S}_1(1^k)$
Return $(\sigma, \tau) \leftarrow S_1^F(1^k)$

$\mathbf{S}_2(\tau, x)$
$u \leftarrow \{0,1\}^{|w|}$
$C_{x,u} = f(1^k, x, u)$
$(pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k)$
$\bar{s} \leftarrow E_{pk}(0^{|w|})$
$v = \mathrm{Eval}_{pk}(C_{x,u}, \bar{s})$
$\pi \leftarrow S_2^F(\tau, (pk, \bar{s}, v))$
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 encryption scheme. 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 encryption scheme 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 encryption scheme. 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 encryption scheme 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 instead of computing $\pi \leftarrow P^F(\sigma, (pk, \bar{s}, v), (\rho, s, \bar{r}))$ it simulates $\pi \leftarrow \mathcal{S}_2^F(\tau, (pk, \bar{s}, v))$. 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 $\bar{s} \leftarrow E_{pk}(0^k)$. By the IND-CPA security of $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ a hybrid argument gives us that 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

We will now give an NIZK proof system that is secure in the universal composability (UC) framework [Can01]. Universally composable NIZK proofs are secure even in an environment where arbitrary other protocols are running concurrently and automatically satisfy strong security notions such as non-malleability. The universally composable NIZK proofs we construct are communication-efficient consisting of $|w| + \mathrm{poly}(k)$ bits.

In the universal composability framework the secure execution of a protocol by a set of parties is modeled by an ideal functionality. We say a protocol is secure if it is equivalent to the parties handing all their inputs to an honest, trusted and incorruptible ideal functionality, which computes the corresponding protocol outputs and hands them to the parties. The parties send their protocol inputs and receive their protocol outputs through a secure private authenticated channel to the ideal functionality, although we allow for the adversary to schedule or block the arrival of outputs.

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

Verification: On input $(\textbf{verify}, sid, x, \pi)$ from a party P_j check whether (sid, x, π) is stored. If not send $(\textbf{verify}, P_j, sid, x, \pi)$ 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 (sid, x, π). If (sid, x, π) has been stored return $(\textbf{verification}, sid, x, \pi, 1)$ to P_j, else return $(\textbf{verification}, sid, x, \pi, 0)$ to P_j.

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

We are interested in securely realizing the ideal non-interactive zero-knowledge functionality $\mathcal{F}_{\text{NIZK}}^{R}$ described in Figure 2. The session ids *sid* are used to distinguish different invocations of the same functionality, which may for instance use different common reference strings in the underlying implementations. The functionality captures completeness by allowing 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. The ideal functionality 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. The ideal functionality also captures an ideal form of zero-knowledge, since it leaks no information about the witnesses used by honest provers.

Let us clarify what it means to securely realize $\mathcal{F}_{\text{NIZK}}^{R}$. We will construct a protocol ϕ_{NIZK} to be run by parties $P_1, \ldots, P_n$ that receive protocol inputs and make protocol 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_{NIZK} 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_{NIZK} simply forward their inputs to the ideal functionality $\mathcal{F}_{\text{NIZK}}^{R}$ and return the responses from $\mathcal{F}_{\text{NIZK}}^{R}$ 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 internals of the ideal execution taking place inside $\mathcal{F}_{\text{NIZK}}^{R}$. Formally, ϕ_{NIZK} 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_{NIZK} 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 $\mathcal{S}$.

We will present a non-interactive protocol ϕ_{NIZK} that securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ for an arbitrary NP-relation R. 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

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

We have generalized R^F slightly compared to the previous section by allowing statements to have an arbitrary string vk in the end. This will be used later in combination with one-time signatures to prevent proofs from being modified.

There are several examples of protocols securely realizing $\mathcal{F}^R_{\text{NIZK}}$ for Circuit Satisfiability in the common reference string model [DDO+02, Gro06, GOS12] and in the multi-string model [GO07] under standard cryptographic assumptions, and a related functionality has been securely realized in the registered public key model [BCNP04]. This implies that we already have many candidates for a secure realization of $\mathcal{F}^{R^F}_{\text{NIZK}}$. A useful feature of the UC framework is the universal composition theorem [Can01] 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 $\mathcal{F}$. Our result therefore says that any secure realization of $\mathcal{F}^{R^F}_{\text{NIZK}}$ implies a communication-efficient secure realization of $\mathcal{F}^R_{\text{NIZK}}$ if fully homomorphic encryption exists.

The construction of our universally composable NIZK proof is quite similar to the NIZK proof in Section 3 except the prover will make a strong one-time signature on each proof in order to prevent modifications of the proof and the protocol will call $\mathcal{F}^{R^F}_{\text{NIZK}}$ instead of using a standard NIZK proof system (K^F, P^F, V^F) in the construction. We therefore proceed directly to giving the details of the protocol in Figure 3.

Universally Composable NIZK Protocol ϕ_{NIZK}

```
$P_i$ on $(\textbf{prove}, sid, x, w)$

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

```
$P_j$ on $(\textbf{verify}, sid, x, \Pi)$

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

Figure 3: Universally composable NIZK proof for R.

Theorem 8 *The protocol ϕ_{NIZK} in Figure 3 securely realizes $\mathcal{F}^R_{\text{NIZK}}$ in the $\mathcal{F}^{R^F}_{\text{NIZK}}$-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_{NIZK} running with $P_1, \ldots, P_n$ and $\mathcal{A}$ and $\mathcal{F}^R_{\text{NIZK}}$ running with dummy parties $\tilde{P}_1, \ldots, \tilde{P}_n$ and $\mathcal{S}$. Our proof strategy is to start with ϕ_{NIZK} 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 $\mathcal{S}_{\text{REAL}}, \mathcal{S}_{\text{EXT}}, \mathcal{S}_{\text{SIM}}$ that are used in intermediate steps and have the ability to control $\mathcal{F}^R_{\text{NIZK}}$ in various ways. Informally, $\mathcal{S}_{\text{REAL}}$ running with the ideal functionality $\mathcal{F}^R_{\text{NIZK}}$ takes full control over $\mathcal{F}^R_{\text{NIZK}}$ and makes a perfect simulation of $\mathcal{A}$ running with ϕ_{NIZK}. $\mathcal{S}_{\text{EXT}}$ modifies the simulation $\mathcal{S}_{\text{REAL}}$ such that whenever an NIZK proof that has not been created by $\mathcal{F}^R_{\text{NIZK}}$ is verified as being valid it extracts the corresponding witness and inputs it to $\mathcal{F}^R_{\text{NIZK}}$. $\mathcal{S}_{\text{SIM}}$ and $\mathcal{S}$ complete the security proof by enabling the simulation of honest parties making NIZK proofs without knowledge of the witnesses.

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

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

$\mathcal{S}_{\text{REAL}}$ 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, $\mathcal{S}_{\text{REAL}}$ 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 $\mathcal{S}_{\text{REAL}}$ receives $(\textbf{prove}, P_i, sid, x)$ from $\mathcal{F}^R_{\text{NIZK}}$ it is because an honest $\tilde{P}_i$ has input $(\textbf{prove}, sid, x, w)$ with $(x, w) \in R$. Since $\mathcal{S}_{\text{REAL}}$ knows the inputs to $\mathcal{F}^R_{\text{NIZK}}$ it can simulate P_i running ϕ_{NIZK} in the $\mathcal{F}^{R^F}_{\text{NIZK}}$-hybrid model including $\mathcal{F}^{R^F}_{\text{NIZK}}$ sending $(\textbf{prove}, (pk, \bar{s}, v, vk))$ to $\mathcal{A}$ and on getting the answer $(\textbf{proof}, \pi)$ making the signature sig to complete the proof $\Pi = (pk, \bar{s}, u, vk, \pi, \text{sig})$. $\mathcal{S}_{\text{REAL}}$ answers $(\textbf{proof}, \Pi)$ to $\mathcal{F}^R_{\text{NIZK}}$.

On input $(\textbf{verify}, P_j, sid, x, \Pi)$ from $\mathcal{F}^R_{\text{NIZK}}$ the simulator $\mathcal{S}_{\text{REAL}}$ knows that the honest party $\tilde{P}_j$ has queried $(\textbf{verify}, sid, x, \Pi)$ to $\mathcal{F}^R_{\text{NIZK}}$, where (sid, 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 $(\textbf{verify}, sid, x, \Pi)$. The simulator forces $\mathcal{F}^R_{\text{NIZK}}$ to return the resulting answer $(\textbf{verification}, sid, x, \Pi, b)$ and stores (sid, x, Π) in $\mathcal{F}^R_{\text{NIZK}}$ if $b = 1$.

The simulation by $\mathcal{S}_{\text{REAL}}$ is exactly like running ϕ_{NIZK} in the $\mathcal{F}^{R^F}_{\text{NIZK}}$-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 $\mathcal{F}^R_{\text{NIZK}}$. However, if we look at a real execution of ϕ_{NIZK} we see that the correctness of the signature scheme, the correctness of the fully homomorphic encryption scheme and the properties of $\mathcal{F}^{R^F}_{\text{NIZK}}$ 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 ϕ_{NIZK} in the $\mathcal{F}^{R^F}_{\text{NIZK}}$-hybrid model with adversary $\mathcal{A}$ is perfectly indistinguishable from the simulation by $\mathcal{S}_{\text{REAL}}$ running with $\mathcal{F}^R_{\text{NIZK}}$.

$\mathcal{S}_{\text{EXT}}$: $\mathcal{S}_{\text{EXT}}$ 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 $(\textbf{verify}, sid, x, \Pi)$ in the execution of ϕ_{NIZK}, but if the answer is $(\textbf{verification}, sid, x, \Pi, 1)$ then it extracts a witness w such that $(x, w) \in R$ and aborts the simulation if the extraction fails.

More precisely, on input $(\textbf{verify}, P_j, sid, x, \Pi)$ from $\mathcal{F}^R_{\text{NIZK}}$ the simulator $\mathcal{S}_{\text{EXT}}$ simulates the honest P_j getting input $(\textbf{verify}, sid, x, \Pi)$ in ϕ_{NIZK}. If P_j outputs $(\textbf{verification}, sid, x, \Pi, 0)$ then $\mathcal{S}_{\text{EXT}}$ returns $(\textbf{witness}, \perp)$ to $\mathcal{F}^R_{\text{NIZK}}$ and forwards the resulting $(\textbf{verification}, sid, x, \Pi, 0)$ message to $\tilde{P}_j$. On the other hand, if P_j outputs $(\textbf{verification}, sid, x, \Pi, 1)$ then $\mathcal{S}_{\text{EXT}}$ will try to extract a witness w such that $(x, w) \in R$, return $(\textbf{witness}, w)$ to $\mathcal{F}^R_{\text{NIZK}}$ and send the resulting $(\textbf{verification}, sid, x, \Pi, 1)$ 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, when the signature sig on $(x, pk, \bar{s}, u, vk, \pi)$ is valid, because otherwise the protocol ϕ_{NIZK} will reject the proof.

Part of the verification protocol consists in querying $\mathcal{F}_{\text{NIZK}}^{R^F}$ on $(\mathbf{verify}, sid, (pk, \bar{s}, v, vk), \pi)$, where $v = \text{Eval}_{pk}(C_{x,u}, \bar{s})$. The simulated $\mathcal{F}_{\text{NIZK}}^{R^F}$ will only return $(\mathbf{verification}, sid, x, \pi, 1)$ 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 encryption scheme 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 modifications 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 $(\mathbf{prove}, sid, (pk, \bar{s}, v, vk)), \perp)$ 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 $(sid, (pk, \bar{s}, v, vk), \pi)$ as being a valid proof and return $(\mathbf{proof}, sid, (pk, \bar{s}, v, vk), \pi)$ to the requesting party P_i.

$\mathcal{S}_{\text{EXT}}$ can now change the way it constructs $\bar{s}$ to instead set $\bar{s} \leftarrow E_{pk}(0^k)$. 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 ϕ_{NIZK} running with $\mathcal{A}$ in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model. The protocol ϕ_{NIZK} therefore securely realizes $\mathcal{F}_{\text{NIZK}}^{R}$ in the $\mathcal{F}_{\text{NIZK}}^{R^F}$-hybrid model. □

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

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, the multi-string model and under any other setup assumption under which universally composable NIZK proofs exist.

5 Fully Homomorphic Hybrid Encryption

We have implicitly been using a hybrid encryption approach in the construction of the non-interactive zero-knowledge proofs. The underlying hybrid encryption scheme has ciphertexts that are both fully homomorphic and communication-efficient: the encryption of a plaintext m has size $|m| + \mathrm{poly}(k)$. Such a scheme is useful in itself, so we will now explicitly define fully homomorphic encryption for arbitrary length messages and show that the construction satisfies the definition.[2] We note that, unlike some definitions of fully homomorphic encryption, we allow the encrypted output to be encoded differently from the encrypted input.

5.1 Defining fully homomorphic encryption of arbitrary size messages

A fully homomorphic encryption scheme 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 plaintext $m \in \{0,1\}^*$ (and randomness $r \leftarrow \{0,1\}^{\ell_{E^*}(k)}$) outputs a ciphertext c. If we have a tuple of plaintexts $\vec{m} = (m_1, \ldots, m_n)$ to encrypt, we will for simplicity write $\vec{c} \leftarrow E^*_{pk}(\vec{m})$ when generating $\vec{c} = (c_1, \ldots, c_n)$ as $c_i \leftarrow E^*_{pk}(m_i)$ (using randomness $\vec{r} = (r_1, \ldots, r_n) \leftarrow (\{0,1\}^{\ell_{E^*}(k)})^n$). The deterministic polynomial time decryption algorithm D^* given a public key pk and a ciphertext returns a message m or an error symbol $\perp$. Finally, the (possibly probabilistic) polynomial time evaluation algorithm Eval^* takes a public key pk, a Boolean circuit C and n ciphertexts as input and returns a ciphertext. A well-formed request to Eval^* with a circuit that has n blocks of $t_1, \ldots, t_n$ input wires and t_{out} output wires includes input ciphertexts with plaintexts of lengths $t_1, \ldots, t_n$ and the output ciphertext then contains a plaintext of size t_{out}. We require that the encryption scheme be compact, which here means that there is a polynomial upper bound $\ell^*_{\mathrm{Eval}}(k, t_{\mathrm{out}})$ on the size of the ciphertexts output by Eval^*.

Definition 9 (Correctness) $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ *is (perfectly) correct if for all Boolean circuits C and all valid inputs $\vec{m} = (m_1, \ldots, m_n)$*

$$\Pr\Big[(pk, dk) \leftarrow K^*_{\mathrm{FHE}}(1^k); \vec{c} \leftarrow E^*_{pk}(\vec{m}); v \leftarrow \mathrm{Eval}^*_{pk}(C, \vec{c}) : D^*_{dk}(v) = C(m_1, \ldots, m_n)\Big] = 1.$$

Definition 10 (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 $(\mathcal{A}, \mathcal{D})$*

$$\Pr\Big[(pk, dk) \leftarrow K^*_{\mathrm{FHE}}(1^k); (m_0, m_1, \mathrm{St}) \leftarrow \mathcal{A}(pk); b \leftarrow \{0,1\}; c \leftarrow E^*_{pk}(m_b) : \mathcal{D}(\mathrm{St}, c) = b\Big] \approx \frac{1}{2},$$

where $\mathcal{A}$ outputs m_0, m_1 of the same bit-length.

Correctness and IND-CPA security were needed in the construction of non-interactive zero-knowledge proofs. However, in other scenarios such as secure function evaluation it is also necessary that the evaluation algorithm hide the circuit C used by the evaluation algorithm. Our definition of circuit privacy does not require the output of the evaluation algorithm to look identical to a

[2] In fact, our non-interactive zero-knowledge proofs require a bit more than just a fully homomorphic encryption scheme for arbitrary size messages. In the constructions, we used directly that one of the components u of the hybrid encryption scheme is of the same bit-length as the witness and therefore automatically a valid encryption of some $|u|$-bit witness w. This means we did not have to prove u was well-formed and kept down the size of the NIZK proof π. For this reason, we needed to work directly with the construction instead of just plugging in any fully homomorphic encryption scheme for witness-length messages.

fresh encryption of the output. Instead, it only requires that this encrypted output reveal to the decryption algorithm no more than the plaintext output $C(m)$. This corresponds to the notion of "1-hop" homomorphic encryption from [GHV10]. We require circuit privacy to hold even when the randomness used by the key generation and by the encryption algorithm are known; this is crucial for applications in which one party generates keys and encryptions and another evaluates C.

Definition 11 (Circuit privacy) $(K^*_{\text{FHE}}, E^*, D^*, \text{Eval}^*)$ *is computationally circuit private if for all non-uniform polynomial time* $(\mathcal{A}, \mathcal{D})$

$$\begin{aligned} &\Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K^*_{\text{FHE}}}(k)}; (pk, dk) \leftarrow K^*_{\text{FHE}}(1^k; \rho); (C, \vec{m}, \vec{r}, \text{St}) \leftarrow \mathcal{A}(\rho); \\ &\qquad \vec{c} = E^*_{pk}(\vec{m}; \vec{r}); v \leftarrow \text{Eval}^*_{pk}(C, \vec{c}) : \mathcal{D}(\text{St}, v) = 1\Big] \\ \approx\ &\Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K^*_{\text{FHE}}}(k)}; (pk, dk) \leftarrow K^*_{\text{FHE}}(1^k; \rho); (C, \vec{m}, \vec{r}, \text{St}) \leftarrow \mathcal{A}(\rho); \\ &\qquad v \leftarrow \text{Eval}^*_{pk}(\text{Id}, E^*_{pk}(C(\vec{m}))) : \mathcal{D}(\text{St}, v) = 1\Big], \end{aligned}$$

where $\mathcal{A}$ outputs consistent C and $\vec{m} = (m_1, \ldots, m_n)$ and $\vec{r}$, and where Id *is a circuit that simply returns its input (with t_{out} input and output wires).*

$(K^*_{\text{FHE}}, E^*, D^*, \text{Eval}^*)$ *is* statistically *circuit private if the above holds even for unbounded $(\mathcal{A}, \mathcal{D})$.*

Often statistical circuit privacy can be obtained by running a rerandomization algorithm on the output ciphertext after completing a deterministic fully homomorphic evaluation of the circuit. Such a division of labor gives us the best of two worlds. In our construction of non-interactive zero-knowledge proofs it was necessary to have a deterministic evaluation algorithm such that the computation could be replicated by the verifier who knew what the circuit was. In the secure function evaluation protocol in Section 6 on the other hand we will need circuit privacy but are happy to have a probabilistic circuit evaluation algorithm.

5.2 Length-optimal fully homomorphic hybrid encryption scheme construction

In our constructions of non-interactive zero-knowledge proofs we implicitly used a hybrid encryption approach where we first encrypt a symmetric encryption key and then use that to encrypt the plaintext. By using a pseudorandom one-time pad we made the ciphertext have size $|m| + \text{poly}(k)$, which is minimal except for an additive overhead. We recap the construction in Figure 4, which relies on a fully homomorphic bit-encryption scheme $(K_{\text{FHE}}, E, D, \text{Eval})$ and a pseudorandom generator G as defined in Section 2.

```
$\mathbf{K}^*_{\text{FHE}}(1^k) = K_{\text{FHE}}(1^k)$

$\mathbf{E}^*_{pk}(m)$
  $s \leftarrow \{0,1\}^k$
  $u = m \oplus G(s, |m|)$
  $\bar{s} \leftarrow E_{pk}(s)$
  Return $c = (\bar{s}, u)$

$\mathbf{D}^*_{dk}(v) = D_{dk}(v)$
```

```
$\mathbf{Eval}^*_{pk}(C, c_1, \ldots, c_n)$
  Parse $c_i = (\bar{s}_i, u_i)$
  $C_u \leftarrow f(1^k, C, u_1, \ldots, u_n)$
  Return $v \leftarrow \text{Eval}_{pk}(C_u, \bar{s}_1, \ldots, \bar{s}_n)$

  where $f(1^k, C, u_1, \ldots, u_n)$ returns a circuit $C_u$
  such that for all $s_1, \ldots, s_n \in \{0,1\}^k$
  $C_u(s_1, \ldots, s_n) = C(u_1 \oplus G(s_1, |u_1|), \ldots, u_n \oplus G(s_n, |u_n|))$
```

Figure 4: Fully homomorphic encryption for arbitrary size plaintexts.

Lemma 12 *If* $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ *is IND-CPA secure and* G *is a pseudorandom generator then* $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ *is IND-CPA secure.*

Proof. Consider the generation of $\bar{s} \leftarrow E_{pk}(s)$. By the IND-CPA security of the bit-encryption scheme this is computationally indistinguishable from $\bar{s} \leftarrow E_{pk}(0^k)$. Next, we pick $u \leftarrow \{0,1\}^{|m|}$ uniformly at random instead of generating it as a pseudorandom one-time pad. By the pseudorandomness of G this modification only changes the adversary's distinguishing advantage negligibly. Now everything but the length of the encrypted plaintext is perfectly hidden and we conclude $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ is IND-CPA secure. □

Lemma 13 *If* $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ *has computational (statistical) circuit privacy then* $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ *as constructed above has computational (statistical) circuit privacy.*

Proof. Given a circuit privacy adversary $(\mathcal{A}, \mathcal{D})$ for $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ we can construct a circuit privacy adversary $(\mathcal{B}, \mathcal{D})$ for $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$. The adversary $\mathcal{B}(\rho)$ runs $(C, m_1, \ldots, m_n, (s_1, \bar{r}_1), \ldots, (s_n, \bar{r}_n), \mathrm{St}) \leftarrow \mathcal{A}(\rho)$. It then computes $u_1 = m_1 \oplus G(s_1, |m_1|), \ldots, u_n = m_n \oplus G(s_n, |m_n|)$ and $C_u = f(1^k, C, u_1, \ldots, u_n)$. $\mathcal{B}$ returns $(C_u, s_1, \ldots, s_n, \bar{r}_1, \ldots, \bar{r}_n, \mathrm{St})$.

The computational (statistical) circuit privacy of $(K_{\mathrm{FHE}}, E, D, \mathrm{Eval})$ gives us

$$\begin{aligned}
& \Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K^*_{\mathrm{FHE}}(k)}}; (pk, dk) \leftarrow K^*_{\mathrm{FHE}}(1^k; \rho); (C, m_1, \ldots, m_n, (s_1, \bar{r}_1), \ldots, (s_n, \bar{r}_n), \mathrm{St}) \leftarrow \mathcal{A}(\rho); \\
& \qquad c_i = E^*_{pk}(m_i; (s_i, \bar{r}_i)); v \leftarrow \mathrm{Eval}^*_{pk}(C, c_1, \ldots, c_n) : \mathcal{D}(\mathrm{St}, v) = 1\Big] \\
= \; & \Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}(k)}}; (pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k; \rho); (C_u, s_1, \ldots, s_n, \bar{r}_1, \ldots, \bar{r}_n, \mathrm{St}) \leftarrow \mathcal{B}(\rho); \\
& \qquad \bar{s}_i = E_{pk}(s_i; \bar{r}_i); v \leftarrow \mathrm{Eval}_{pk}(C_u, \bar{s}_1, \ldots, \bar{s}_n) : \mathcal{D}(\mathrm{St}, v) = 1\Big] \\
\approx \; & \Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K_{\mathrm{FHE}}(k)}}; (pk, dk) \leftarrow K_{\mathrm{FHE}}(1^k; \rho); (C_u, s_1, \ldots, s_n, \bar{r}_1, \ldots, \bar{r}_n, \mathrm{St}) \leftarrow \mathcal{B}(\rho); \\
& \qquad v \leftarrow \mathrm{Eval}_{pk}(\mathrm{Id}, E_{pk}(C_u(s_1, \ldots, s_n))) : \mathcal{D}(\mathrm{St}, v) = 1\Big] \\
= \; & \Pr\Big[\rho \leftarrow \{0,1\}^{\ell_{K^*_{\mathrm{FHE}}(k)}}; (pk, dk) \leftarrow K^*_{\mathrm{FHE}}(1^k; \rho); (C, m_1, \ldots, m_n, (s_1, \bar{r}_1), \ldots, (s_n, \bar{r}_n), \mathrm{St}) \leftarrow \mathcal{A}(\rho); \\
& \qquad v \leftarrow \mathrm{Eval}^*_{pk}(\mathrm{Id}, E^*_{pk}(C(m_1, \ldots, m_n))) : \mathcal{D}(\mathrm{St}, v) = 1\Big],
\end{aligned}$$

which shows $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ has computational (statistical) circuit privacy. □

Let us call an encryption scheme length-optimal if the encryption procedure produces ciphertexts of size $|m| + \mathrm{poly}(k)$, where $|m|$ is the size of the plaintext and k is the security parameter. The following theorem then summarizes the results of this section.

Theorem 14 *If IND-CPA secure fully homomorphic public key bit-encryption schemes exist, then length-optimal IND-CPA secure fully homomorphic public key encryption schemes for arbitrary size plaintexts exist. If IND-CPA secure circuit private fully homomorphic public key bit-encryption schemes exist, then length-optimal IND-CPA secure circuit private fully homomorphic public key encryption schemes for arbitrary size plaintexts exist.*

Proof. The existence of bit-encryption implies the existence of pseudorandom generators. The construction of $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ in Figure 4 is perfectly correct. Since the encryption procedure outputs ciphertexts of size $|m| + \mathrm{poly}(k)$ the scheme is length-optimal. Lemma 12 shows the construction preserves IND-CPA security and Lemma 13 shows that the construction preserves circuit privacy. □

6 2-Party Secure Function Evaluation

As an illustration of the usefulness of fully homomorphic encryption for messages of arbitrary length, we will look at the case of 2-party secure function evaluation. Here Alice and Bob have inputs x and y respectively and Alice wants to learn $f(x, y)$ for a given polynomial time computable function f.

Given a fully homomorhic encryption scheme for arbitrary size plaintexts the construction of a 2-party secure function evaluation protocol is very simple. Alice encrypts her input x under her own public key and send the ciphertext to Bob. Bob then applies a suitable circuit to the ciphertext to compute the evaluation of $f(x, y)$ and sends it back to Alice. Alice can now decrypt and get the result. (A slight extension of the protocol would allow also Bob to learn the result $f(x, y)$ by having Alice sending the result encrypted under Bob's public key.) Intuitively, Alice's input will remain private because of the IND-CPA security of the encryption scheme while Bob's input will remain private because of the circuit privacy of the encryption scheme. The full construction can be found in Figure 6.

To formalize 2-party secure function evaluation, we use the universal composability framework [Can01], which we also used for non-interactive zero-knowledge proofs in Section 4. Here, however, we will only consider the case of static adversaries that corrupt a fixed set of parties from the start of the protocol and do not corrupt any more parties after that. We will also restrict ourselves to the case of honest-but-curious adversaries. This means corrupted parties controlled by the adversary follow the protocol honestly but the adversary will try to learn some extra information about the honest parties private inputs. We do not assume the parties have access to a common reference string but we do assume their communication is authenticated. If a party P_i receives a message from P_j it is therefore guaranteed this message did indeed originate from P_j although we can expect the adversary to have learned the contents of the message.

The ideal 2-party secure function evaluation functionality is given in Figure 5. Note that the size of the input x and the size of the output $z = f(x, y)$ are not secret but everything else is kept secret by the ideal functionality. There is one restriction we place on the polynomial time function f, which is that given $|x|$ and y it should be possible to determine uniquely the size of the output $f(x, y)$. This restriction, which is standard since UC protocols typically do not hide the lengths of inputs and outputs, is needed in our construction to enable the construction of a polynomial size circuit $C_{|x|,y}(\cdot)$ with a fixed number of output wires that corresponds to the computation of $f(\cdot, y)$.

Ideal 2-party secure function evaluation functionality $\mathcal{F}^f_{\mathrm{2SFE}}$

Parameterized with a polynomial time function $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ and running with parties $\tilde{P}_1, \ldots, \tilde{P}_n$ and adversary $\mathcal{S}$.

- On input $(\mathbf{input}, sid, P_i, P_j, x)$ from a party P_i store (sid, P_i, P_j, x), send $(sid, P_i, P_j, |x|)$ to $\mathcal{S}$ and ignore any future $(\mathbf{input}, sid, P_i, P_j, *)$ inputs from P_i.
- On input $(\mathbf{input}, sid, P_i, P_j, y)$ from a party P_j where (sid, P_i, P_j, x) has been stored, compute $z = f(x, y)$, send $(sid, P_i, P_j, |z|)$ to $\mathcal{S}$ and $(\mathbf{result}, sid, P_i, P_j, z)$ to P_i and ignore future inputs of the form $(\mathbf{input}, sid, P_i, P_j, *)$ from P_j.

Figure 5: Ideal 2-party secure function evaluation functionality $\mathcal{F}^f_{\mathrm{2SFE}}$.

Theorem 15 *ϕ_{2SFE} in Figure 6 securely realizes F^f_{2SFE} against static honest-but-curious adversaries if $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ is an IND-CPA secure fully homomorphic encryption scheme with*

Universally composable 2-party secure function evaluation protocol ϕ_{2SFE}

- A party P_i on input $(\textbf{input}, sid, P_i, P_j, x)$ from the environment generates $(pk, dk) \leftarrow K^*_{\text{FHE}}(1^k)$ and $c \leftarrow E^*_{pk}(x)$. It stores (sid, P_i, P_j, dk) and sends $(sid, P_i, P_j, pk, c, |x|)$ to P_j.

  P_i ignores future inputs of the form $(\textbf{input}, sid, P_i, P_j, *)$ from the environment and waits for a response of the form (sid, P_i, P_j, v) from P_j. When receiving such a response it computes $z = D^*_{dk}(v)$ and outputs $(\textbf{result}, sid, P_i, P_j, z)$ to the environment.

- A party P_j after receiving $(\textbf{input}, sid, P_i, P_j, y)$ from the environment and message $(sid, P_i, P_j, pk, c, |x|)$ from party P_i creates a circuit $C_{|x|,y}(\cdot)$ corresponding to $f(\cdot, y)$ on length $|x|$ inputs and computes $v \leftarrow \text{Eval}^*_{pk}(C_{|x|,y}, c)$. P_j then sends (sid, P_i, P_j, v) to P_i and ignores all future inputs of the form $(\textbf{input}, sid, P_i, P_j, *)$.

Figure 6: Universally composable 2-party secure function evaluation for f.

computational circuit privacy.

Proof. $\mathcal{S}$ in the ideal process runs a copy of $\mathcal{A}$ and will try to simulate an execution of the protocol such that $\mathcal{A}$ and the environment $\mathcal{Z}$ cannot detect that they are running in the ideal process. $\mathcal{S}$ forwards all communication between the simulated $\mathcal{A}$ and the environment $\mathcal{Z}$ and $\mathcal{S}$ simulates for each dummy party $\tilde{P}_i$ a corresponding real world party P_i. Since $\mathcal{A}$ is static the set of corrupted parties is fixed from the start and $\mathcal{S}$ has full control over the corresponding corrupted dummy parties. Whenever a corrupted dummy party $\tilde{P}_i$ receives an input $\mathcal{S}$ sends it to the ideal functionality, and whenever a simulated party P_i makes an output to the environment, $\mathcal{S}$ delivers the corresponding output from the ideal functionality to $\tilde{P}_i$ so it can output it to the environment. For each pair of parties (P_i, P_j) engaging in the protocol there are four options:

P_i and P_j are both corrupt: As specified above $\mathcal{S}$ learns the inputs x and y of the parties and submits them to the ideal functionality on behalf of $\tilde{P}_i$ and $\tilde{P}_j$. If $\mathcal{A}$ in the protocol outputs the result to $\mathcal{Z}$ on behalf of P_i, then $\mathcal{S}$ delivers the ideal functionality's result to $\tilde{P}_i$ and outputs it to the environment.

P_i is corrupt and P_j is honest: In this setting $\mathcal{S}$ has to simulate the message (sid, P_i, P_j, v) that P_j sends to P_i. Since $\tilde{P}_i$ is corrupted, $\mathcal{S}$ knows the ideal functionality's result $z = f(x, y)$. It can therefore compute $v \leftarrow \text{Eval}^*_{pk}(\text{Id}, E^*_{pk}(z))$ and use that in P_j's response to P_i.

P_i is honest and P_j is corrupt: In this setting $\mathcal{S}$ has to simulate the message $(sid, P_i, P_j, pk, c, |x|)$ that P_i sends to P_j. On input $(sid, P_i, P_j, |x|)$ from the ideal functionality $\mathcal{S}$ therefore picks $(pk, dk) \leftarrow K^*_{\text{FHE}}(1^k)$ and $c \leftarrow E^*_{pk}(0^{|x|})$ and instructs P_i to send $(sid, P_i, P_j, pk, c, |x|)$ to P_j.

P_i and P_j are both honest: Here $\mathcal{S}$ has to simulate the entire communication between P_i and P_j. On input $(sid, P_i, P_j, |x|)$ from the ideal functionality $\mathcal{S}$ therefore picks $(pk, dk) \leftarrow K^*_{\text{FHE}}(1^k)$ and $c \leftarrow E^*_{pk}(0^{|x|})$ and instructs P_i to send $(sid, P_i, P_j, pk, c, |x|)$ to P_j. On subsequent input $(sid, P_i, P_j, |z|)$ from the ideal functionality $\mathcal{S}$ then computes $v \leftarrow \text{Eval}^*_{pk}(\text{Id}, E^*_{pk}(0^{|z|}))$ and instructs P_j to send (sid, P_i, P_j, v) to P_i.

To see this is a good simulation, observe first that since $\mathcal{A}$ is honest-but-curious it always instructs corrupt parties P_i to act like an honest party would do, so it does for instance output the correct protocol output in the simulation when both parties are corrupt, generates a valid key and ciphertext when P_i is corrupt, and computes v correctly on the basis of y if P_j is corrupt. The IND-CPA security of the encryption scheme guarantees the ciphertext $c \leftarrow E^*_{pk}(0^{|x|})$ in the simulation when P_i is honest cannot be distinguished from the correct encryptions $c \leftarrow E^*_{pk}(x)$. The computational circuit privacy of the encryption scheme guarantees that $v \leftarrow \mathrm{Eval}^*_{pk}(\mathrm{Id}, E^*_{pk}(z))$ cannot be distinguished from the correct evaluation $v \leftarrow \mathrm{Eval}^*_{pk}(C_{|x|,y}, c)$ when P_j is honest, even when given the randomness used for generating c. When both P_i and P_j are honest the computational circuit privacy shows the real protocol running with $v \leftarrow \mathrm{Eval}^*_{pk}(C_{|x|,y}, c)$ cannot be distinguished from a simulated $v \leftarrow \mathrm{Eval}^*_{pk}(\mathrm{Id}, E^*_{pk}(f(x, y)))$. The IND-CPA security then says this cannot be distinguished from running $v \leftarrow \mathrm{Eval}^*_{pk}(\mathrm{Id}, E^*_{pk}(0^{|z|}))$ and that having $c \leftarrow E^*_{pk}(x)$ cannot be distinguished from $c \leftarrow E^*_{pk}(0^{|x|})$. This means the simulation in the ideal process cannot be distinguished by $\mathcal{Z}$ from the real execution of the protocol. □

Using the hybrid encryption scheme $(K^*_{\mathrm{FHE}}, E^*, D^*, \mathrm{Eval}^*)$ from Section 5 Alice's (P_i's) communication is $|x| + \mathrm{poly}(k)$ bits. For small output sizes $|z| = O(1)$ this gives an overall communication of $|x| + \mathrm{poly}(k)$ so there is only an additive overhead.

For large output sizes, the existence of fully homomorphic bit-encryption only guarantees a total communication of $|x| + \mathrm{poly}(k)|z|$ bits. However, using specific cryptographic assumptions it is possible to get a communication complexity of $|x| + |z| \cdot (1 + o(1)) + \mathrm{poly}(k)$. This follows from Lipmaa's [Lip11] method to batch many ciphertexts into one in the fully homomorphic encryption scheme by Brakerski, Gentry and Vaikuntanathan [BGV12].

Acknowledgments

Jens Groth was supported by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 307937 and the Engineering and Physical Sciences Research Council grant EP/G013829/1.

Yuval Ishai was supported by the European Union's Tenth Framework Programme (FP10/2010-2016) under grant agreement no. 259426 ERC-CaC, by ISF grant 1361/10, and by BSF grant 2012378.

Chris Peikert was supported by the National Science Foundation under CAREER Award CCF-1054495, by the Alfred P. Sloan Foundation, and by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under Contract No. FA8750-11-C-0098. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the National Science Foundation, the Sloan Foundation, DARPA or the U.S. Government.

Amit Sahai was supported in part from a DARPA/ONR PROCEED award, NSFgrants 1228984, 1136174, 1118096, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11- 1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

Adam Smith was supported by US National Science Foundation awards #0941553 and #0747294.

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.

[BGH13] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in lwe-based homomorphic encryption. In *Public Key Cryptography*, volume 7778 of *Lecture Notes in Computer Science*, pages 1–13, 2013.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, pages 309–325, 2012.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886, 2012.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS*, 2011.

[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 on 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 on 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.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-hop homomorphic encryption and rerandomizable Yao circuits. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172, 2010.

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

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC*, pages 113–122, 2008.

[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.

[GOS12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *Journal of the ACM*, 59(3):11, 2012.

[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.

[GVW02] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

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

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

[Ish09] Yuval Ishai. Efficiency vs. assumptions in secure computation. Presentation at Impagliazzo's Worlds Workshop, 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.

[Lip11] Helger Lipmaa. Efficient multi-query CPIR from ring-LWE. Cryptology ePrint Archive, Report 2011/595, 2011.

[NN01] Moni Naor and Kobbi Nissim. Communication preserving protocols for secure function evaluation. In *STOC*, pages 590–599, 2001.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437, 1990.

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

[PS09] Chris Peikert and Adam Smith. Concise, uninformative proofs. Rump Session Presentation at Asiacrypt, 2009.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[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.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS*, pages 160–164, 1982.