Black-Box Proof of Knowledge of Plaintext and Multiparty Computation with Low Communication Overhead

Steven Myers[1,⋆], Mona Sergi[2], and abhi shelat[2]

[1] Indiana University, Bloomington, IN, USA
[2] University of Virginia, Charlottesville, VA, USA

Abstract. We present a 2-round protocol to prove knowledge of a plaintext corresponding to a given ciphertext. Our protocol is black-box in the underlying cryptographic primitives and it can be instantiated with almost any fully homomorphic encryption scheme.

Since our protocol is only 2 rounds it cannot be zero-knowledge [GO94]; instead, we prove that our protocol ensures the semantic security of the underlying ciphertext.

To illustrate the merit of this relaxed proof of knowledge property, we use our result to construct a secure multi-party computation protocol for evaluating a function f in the standard model using only *black-box access* to a threshold fully homomorphic encryption scheme. This protocol requires communication that is *independent of* $|f|$; while Gentry [Gen09a] has previously shown how to construct secure multi-party protocols with similar communication rates, the use of our novel primitive (along with other new techniques) avoids the use of complicated generic white-box techniques (cf. PCP encodings [Gen09a] and generic zero-knowledge proofs [AJLA$^+$12, LATV11].)

In this sense, our work demonstrates in principle that *practical* TFHE can lead to reasonably practical secure computation.

Keywords: Fully Homomorphic Encryption, Threshold Encryption, Secure Multi-Party Computation, Communication and Round Complexity, Proof Of Knowledge.

1 Introduction

The main technical contribution of this paper is a novel proof of knowledge of a plaintext protocol and its demonstrated use in the construction of a fully black-box multi-party computation protocol with low communication overhead. We briefly describe the motivation behind our work.

[⋆] This work, and the authors are sponsored by NSF Grant 0939718, and DARPA and Air Force Research Laboratory under Grant FA8750-11-2-0211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US government.

A. Sahai (Ed.): TCC 2013, LNCS 7785, pp. 397–417, 2013.

Communication. Secure computation with an honest majority can be accomplished without any cryptographic assumptions, but the best such protocol requires the parties to communicate $|f| \log |f| + d^2 \cdot poly(n, \log |f|)$ bits [DIK10] and at least d rounds. Here $|f|$ is the size of the function being computed and d is the circuit depth of f, and thus the communication of the protocol is super-linearly related to the number of gates in f. Until recently, even the use of cryptographic assumptions for secure computation required $polylog(\lambda)$ communication overhead per gate [DIK10] where λ is a security parameter.

Gentry [Gen09a] circumvents per-gate overhead as follows: the honest-but-curious parities use secure multi-party computation to generate an FHE key, each party encrypts its input, and sends the resulting ciphertext and proof to other parties. Once all parties have encryptions of everyone's inputs, they compute the function of interest locally using the evaluation procedure of the FHE. Finally, to use the resulting ciphertexts as inputs to a secure multi-party computation which computes the decryption of the majority input. In order to be secure against malicious adversaries, the Naor and Nissim compiler [NN01], which makes use of the PCP theorem, can be applied. The use of the PCP theorem in the SMC steps makes the approach impractical, even when presented with a practical FHE scheme.

The motivation behind our work is to remove any use white-box techniques, such as the PCP theorem or generic ZK or NIZK, from the above framework for constructing communication-efficient secure protocols. These techniques have historically been inefficient. In other words, we seek a black-box transformation from TFHE to secure computation.

First Contribution. The main technical hurdle in devising a black-box transformation from TFHE to secure computation is to implement the requirement for each player to prove that they "know the plaintext" corresponding to the encrypted input that they have broadcast. This step is essential because it prevents one player from copying (or mauling via the homomorphism) the input of a player who has acted earlier. To handle this step, we show how to construct a two-round black-box proof of knowledge of an encrypted bit for any circuit private FHE scheme using only the encryption scheme. Since our protocol is only two rounds, it is not zero-knowledge (cf. [GO94]), but can provably keep the encrypted bit hidden. Our POK requires that the public-key contain a labeled encryption of 0 and 1, which given all known FHE schemes seems to be a natural modification. [1] For traditional FHE schemes, the POK can be used completely black-box, without even the need for the modification.

The basic idea of our proof of knowledge protocol is to first modify the encryption scheme so that the message is encoded using an error-correcting code (ECC) based verifiable secret sharing (VSS) scheme. To encrypt a message we

[1] Since all current schemes contain bit-wise encryptions of their own secret-keys which are random bit strings, and a natural extension of any protocol that provides encryptions of one's own secret-key can be used to derive a labeled encryption of 0 and 1 which we describe.

first generate its secret shares, and encrypt them independently using fresh randomness. A verifier now requests the Prover to reveal the randomness used to encrypt a sub-threshold number of the shares. The verifier then does a consistency check, based on the ECC underlying the scheme, to ensure that the shares were encoded properly. In particular, the error-correcting code we choose offers a property that allows one to check whether local parts of the codeword are error-free. The verifier accepts if everything appears to be properly coded. Since the number of shares revealed is less than the threshold, it does not leak any information about the original message. To show a proof of knowledge property, we argue that an extractor can rewind the Prover and ask for another set of shares to be opened. With high probability, this second transcript provides enough new shares to run the VSS recover algorithm, and recover the original message. The one issue with this approach is that the Prover must reveal the randomness used to encrypt some of the shares. The semantic security of an encryption scheme does not guarantee any security when these random bits are revealed—in particular, the security of the rest of the unopened encryptions are not guaranteed. Instead, we require the encryption scheme to be secure against a selective opening attack (SOA). Fortunately, a result of Hemenway et al. [HLOV11] can be generalized to show that any circuit private homomorphic encryption scheme can be made into an SOA-secure one.

We point out that our proof of knowledge requires the encryption scheme to be homomorphic and circuit-private. Recently, Damgård et al. [DPSZ12] demonstrates a three-round Σ-protocol for knowledge of plaintext, but their protocol *requires* the underlying encryption scheme to also be homomorphic on the random coins used to encrypt. Although many FHE schemes support this property on their random coins, it is certainly not specified in the definition of FHE. In contrast, circuit privacy has been independently defined and seems to be a naturally weaker property.[2] Moreover, their scheme requires the message space for the FHE to be over $\mathbb{Z}_N$ for N related to the security parameter. While in general, single-bit FHE implies many-bit FHE, we are not aware of any such transformation that *also* preserves the homomorphism over the random coins as required by their protocol. Thus, the requirement for large message space *and* homomorphism over the random coins seem to be extra assumption which our work can avoid (our protocol also works on single-bit FHE). Finally, the Σ-protocol from [DPSZ12] must be compiled into a full zero-knowledge protocol using standard techniques which add round complexity and/or setup assumptions; we show that our two-round protocol with its hidden-bit property suffices for our secure computation protocol.

Second Contribution. By combining our result with almost any TFHE scheme, we construct a secure multi-party protocol that avoids *both* per-gate communi-

[2] Even though current schemes achieve circuit privacy via randomness homomorphisms, it is certainly plausible for future constructions to achieve circuit privacy in other ways. Moreover, there do not seem to be any natural ways to transform a circuit private scheme to one with a randomness homomorphism, and thus we feel it is a weaker notion.

cation complexity *and* white-box techniques such as the PCP theorem or Zero-Knowledge. The communication complexity of our protocol is $O(\lambda^c \cdot n^2)$ where λ is a security parameter and c is a small constant for the TFHE scheme and is thus independent of $|f|$. Our black-box transformation is particularly important because if practical FHE (and TFHE) can be constructed, our transformations will result in practical SFE. Our work is in the standard model and does not require trust assumptions such as the common reference string, a random oracle or public-key setup.

Final Contribution. For completeness, we also construct a *threshold* fully homomorphic public-key encryption scheme (TFHE) based on the Approximate GCD problem and the fully homomorphic encryption scheme presented by van Dijk et al. [vDGHV10], and our result was the first to demonstrate the feasibility of directly achieving this threshold primitive for FHE. Since our original eprint submission, [AJLA$^+$12] and [LATV11] present more efficient TFHE constructions based on LWE-style assumptions. The point of this construction is to demonstrate feasibility of TFHE under different complexity assumptions.[3]

We present our protocols in the information-theoretic model over secure point-to-point channels, and thus our protocols are secure in the presence of an honest majority. Thus, when used with our transformation, the resulting protocol is also only secure with an honest majority. By using another TFHE that tolerates a dishonest majority, our transformation results in an secure computation protocol that also tolerates the same.

The TFHE scheme provides a constant-round protocol for n players to generate a public-key and distribute private shares of the corresponding secret-key of a fully homomorphic encryption scheme. This step itself is non-trivial since the generation of the public-key for an FHE scheme (that is based on bootstrapping) requires encryption of the secret-key. Later, a majority of players can cooperatively decrypt a ciphertext by running a constant-round protocol on their private shares and a public ciphertext. We also provide methods for distributed encryption and for proving knowledge of an encrypted value.

We note that both our TFHE key generation and decryption protocols are more efficient than generically applying secure function evaluation techniques to the key generation or decryption algorithms of an FHE scheme. For example, with the right set of the parameters, our decryption protocol requires only a constant number of share multiplications, whereas generic techniques would require $O(poly(\lambda))$ such multiplications. We heavily exploit the linear nature of the operations involved in key generation, encryption and decryption for the particular FHE scheme of van Dijk et al. For key generation and decryption, we develop specific multiparty computation protocols that evaluates an arithmetic circuit using verifiable secret sharing techniques, that is more efficient than the application of generic techniques.

[3] We note that historically, threshold encryption has been presented where the key-generation algorithm and decryption algorithms are single algorithms, or they are multi-party protocols. We present multi-party protocols.

Comparison with Other FHE-Based Secure Computation Protocols. Gentry's [Gen09a] secure computation protocol was the first to achieve communication complexity that is independent of $|f|$ by using the PCP theorem in several steps.

Asharov, Jain and Wichs [AJLA$^+$12] and López-Alt, Tromer, and Vaikuntanathan [LATV11] have constructed more efficient TFHE schemes based on LWE and the closely related RLWE assumption, which can be reduced to varying degrees to worst-case lattice problems. Their approaches rely on the ability to construct an FHE that also has a homomorphism on the secret-keys, and can also be used to achieve secure computation with communication that is independent of $|f|$. Together, our results demonstrate that the TFHE primitive can be developed from reductions to different classes of hardness assumptions, and therefore TFHE is not simply a consequence of a specific hardness property.

To achieve security against malicious adversaries, López-Alt et al. rely on a common reference string setup so that players can use a NIZK to prove to each other that their keys and their input ciphertexts are well-formed. The use of such NIZK also requires additional hardness assumptions, since (T)FHE is not known to imply NIZK. They can also instantiate their ideas in the standard model by replacing these NIZK proofs with traditional interactive ZK proofs; but in either case, the generic (NI)ZK techniques used require non-blackbox use of the underlying TFHE scheme.[4] By choosing the CRS model, the authors observed that by using a more expensive simulation-sound NIZK, their protocols can also achieve UC-security. Our protocols only claim standard security, but it has bee pointed out to us that it is likely that we can state some of ours results as UC in a TFHE-hybrid model.

Asharov et al. use efficient Σ-Protocol constructions to prove well-formedness; these make heavy use of the underlying mathematical structure of the LWE assumption. In order to have efficient NIZK proofs, they must rely on the use of the Random Oracle model, and the use of the Fiat-Shamir heuristic to transform the Σ-protocols into NIZK proofs. In any case, due to the black-box nature of our SMC construction, with simple modifications to the public-key to include labeled ciphertexts representing encryptions of 0 and 1, either of the López-Alt et al. or Asharov et al. TFHE schemes can be plugged in to our construction to achieve security against an arbitrary number of malicious adversaries, with abort. In contrast, with our scheme we are guaranteed output delivery, but need an honest majority of players.

The protocols of Damgård et al. [DPSZ12] and Bendlin et al. [BDOZ11] use a different approach to constructing secure computation protocols from traditional homomorphic encryption. Their schemes rely on the idea from Beaver [Bea91] for circuit randomization. First, they use an offline phase in which the parties use a somewhat homomorphic encryption primitive to create shares of triples (a, b, c) such that $a \cdot b = c$. One triple is required for each multiplication gate in f that is to be evaluated and requires approximately $O(n/s)$ "heavy" cryptographic

[4] In other words, the encryption algorithm of the TFHE will need to be expressed in terms of a graph-coloring instance (or Hamiltonicity, circuit-sat ,etc...). As far as we know, this transformation requires a high-order polynomial overhead.

operations to generate. Next, after such triples have been created, the parties use only information-theoretic methods to evaluate the circuit. This approach results in admirable communication parameters for small circuits (as they have also run practical examples); nonetheless, the approach requires linear communication for each gate in $|f|$, and thus does not achieve our main aim of eliminating this relationship.

Finally, these prior results are all in a model in which n parties are computing, and the protocols can tolerate up to $n-1$ malicious parties. In contrast, our protocols require an honest majority. The relative incomparability of these models is well understood. In particular, in the model that tolerates up to $n-1$ malicious adversaries, if any one party deviates form the protocol or fails, then all parties output $\perp$. Alternately, with an honest majority, all parties can output an effective output, as supported by our protocol. For a discussion of the relative merits of the two models, and the impossibility of having protocols that achieve the best of both worlds for general functionalities, see the work of Ishai et al. [IKK+11].

In summary, all of these recent works have advantages and disadvantages of their own; our major contribution is the black-box transformation and the independent hardness assumption.

Related work. Cramer, Damgård and Nielson [CDN01], along with Jakobbsson and Juels [JJ00] show how to use threshold cryptography to construct secure multiparty computation protocols. In more detail, we use many ideas from [CDN01] which shows how a homomorphic threshold cryptosystem can be used to achieve general multiparty computation protocols. The notion of using secret-sharing to encode encryptions, as we will do, was first seen in [CDSMW08] and has recently been extended in [GLOV12], although these works use the technique to ensure consistency, and not a proof-of-knowledge, as pursued here.

2 Preliminaries and Notation

A 4-tuple of protocols and algorithms (**Gen**, **Enc**, **Dec**, **Eval**) is a (t, n)-threshold fully homomorphic encryption scheme if the following hold:

Key Generation. An n-party protocol **Gen** that at each invocation returns a new public-key PK and the secret-key $(\mathrm{SK}_1, \ldots, \mathrm{SK}_n)$, where SK_i is the share of the secret-key for Player_i.

Encryption. A PPT algorithm $\mathbf{Enc}_{\mathrm{PK}}(m, r)$ that returns the encryption of the plaintext m under the public-key PK with random coins r.

Decryption. There exists a PPT n-party protocol $\mathbf{Dec}(c, \mathrm{SK}_1, \ldots, \mathrm{SK}_n)$, which returns the plaintext m using the shares SK_i held by honest party Player_i, where $c = \mathbf{Enc}(m, r)$ for some random r.

$f_{\mathbf{PK}}$-homomorphic. There exists a PPT algorithm **Eval** which given a polynomial f, ciphertexts $c_1 \in \mathbf{Enc}_{\mathrm{PK}}(m_1), \ldots, c_k \in \mathbf{Enc}_{\mathrm{PK}}(m_k)$ for some k and a public-key PK, outputs $c \in \mathbf{Enc}(f(m_1, \ldots, m_k))$.

The natural notion of chosen plaintext attack indistinguishability needs to be modified in the venue of threshold cryptography to take into account the fact that the adversary has access to shares of the secret-key. The appropriate corresponding and natural definition is given in [CDN01], and full version of our paper [MSas11]. Standard security notions for secure multi-party computation protocols can be used to define the security for the protocols **Gen** and **Dec** in any given instantiation of a TFHE (e.g., we can consider security in the real/ideal standalone paradigm, the UC framework, etc..)

Next, we present the notion of bootstrapping a ciphertext. Gentry developed the notion of Bootstrapping to reduce noise in a somewhat fully homomorphic encryption scheme, in order to achieve a fully homomorphic scheme. In contrast, we assume the existence of an FHE and simply use it to reduce noise produced in ciphertexts generated in our selective opening attack secure scheme that we introduce later.

Definition 1. *(Bootstrapping a Ciphertext) For a FHE scheme $\Pi = (G, E, D, \textbf{\textit{Eval}})$ and the security parameter k, let D_Π be Π's decryption circuit, which takes a secret-key and s ciphertext as input. Given a ciphertext C encrypted with respect to a public-key PK and secret-key $SK = (SK_1, .., SK_\ell)$ we require that PK contains a bit-wise encryption of SK, denoted $s_1, ..., s_\ell$ where $s_i = E(PK, SK_i)$. Let $(C_1, .., C_n)$ denote the bits of C, and generate $c_i = E(PK, C_i)$. We say that the value $C^\dagger = \textbf{\textit{Eval}}(PK, D_\pi, s_1, ..., s_\ell, c_1, .., c_n)$ (which homomorphically evaluates $D(SK, C)$) is the result of bootstrapping C.*

2.1 Selective Opening Security

In our construction, we will need to refer to encryption schemes where messages that are encrypted remain secure, even after the randomness used to encrypt related messages is revealed. This notion of security is called Selective Opening Security.

Definition 2 (IND-SO-SEC Encryption Security). *A public-key encryption scheme $\Pi = (G, E, D)$ is Indistinguishable Selective Opening secure if, for any message sampler M that supports efficient conditional resampling, and any ppt adversary $A = (A_1, A_2)$ there exists a negligible function μ such that for all sufficiently large k:*

$$\left|\Pr[A_\Pi^{\mathsf{Ind\text{-}SO\text{-}Real}}(1^k) = 1] - \Pr[A_\Pi^{\mathsf{Ind\text{-}SO\text{-}Ideal}}(1^k) = 1]\right| \leq \mu(k).$$

A message sampler M is a PPT algorithm that outputs a vector $\boldsymbol{m}$ of n messages from a given distribution. It is an efficient conditional resampler if, when given two auxiliary inputs, a set of indices $I \subseteq [n]$, and a vector of messages $\boldsymbol{m} = (m_1, ..., m_n)$, M samples another vector $\boldsymbol{m'} = (m'_1, \cdots, m'_n)$ conditioned on $m_i = m'_i$ for each $i \in I$. We define the experiments Ind-SO-Real *and* Ind-SO-Ideal *as follows.*

```
$\mathsf{Ind\text{-}SO\text{-}Real}(1^k, A)$
    $(PK, SK) \leftarrow G(1^k)$
    $\boldsymbol{m} = (m_1, \ldots, m_n) \leftarrow M$
    $r_1, \ldots, r_n \leftarrow R$
    $(I, \sigma) \leftarrow A_1(PK, E_{PK}(m_1, r_1), \ldots, E_{PK}(m_n, r_n))$
    $\textit{Output } A_2(\sigma, (m_i, r_i)_{i \in I}, \boldsymbol{m})$

$\mathsf{Ind\text{-}SO\text{-}Ideal}(1^k, A)$
    $(PK, SK) \leftarrow G(1^k)$
    $\boldsymbol{m} = (m_1, \ldots, m_n) \leftarrow M$
    $(I, \sigma) \leftarrow A_1(PK, E_{PK}(m_1, r_1), \ldots, E_{PK}(m_n, r_n))$
    $\boldsymbol{m'} = (m'_1, \ldots, m'_n) \leftarrow M_{|I, m[I]}.$
    $\textit{Output } A_2(\sigma, (m_i, r_i)_{i \in I}, \boldsymbol{m'})$
```

2.2 Circuit Privacy

Definition 3. *((Statistical) Circuit Private Homomorphic Encryption). A homomorphic encryption scheme $\varepsilon = (\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec})$ is circuit-private for circuits in a set C_ε if, for any key pair (PK, SK) output by $\boldsymbol{Gen}(\lambda)$, any circuit $C \in C_\varepsilon$, and any fixed ciphertext $\psi = \langle \psi_1, \ldots, \psi_t \rangle$ that are in the image of **Enc** for plaintexts $\pi_1, \ldots, \pi_t$, the following distributions (over the random coins in **Enc**, **Eval**) are (statistically) indistinguishable:*

$$\boldsymbol{Enc}_{PK}(C(\pi_1, \ldots, \pi_t)) \approx \boldsymbol{Eval}_{PK}(C, \psi)$$

In the original schemes first presented by both Dijk et al. [vDGHV10] and Gentry [Gen09a], the initial evaluation functions are deterministic and not circuit-private. In order to overcome this problem, both works introduce a method for adding random noise to encryptions, whether they are output from **Eval** or **Enc**, and thus in some sense rerandomizing them. This is done by adding an 'encryption' of 0 to the ciphertext in question, but where the 'encryption' has significantly more noise than would be generated by either the legitimate encryption or evaluation process. Specifically, they introduce ppt algorithms labeled $\mathsf{CircuitPrivacy} : \mathcal{C}_b \to \mathcal{C}'_b$, where C consists of all the ciphertexts that are output from $\mathbf{Enc}_{\mathrm{PK}}(b)$ or a call to **Eval** with an encrypted output bit of b. It is the case that for any b and any $c_{b,0}$,$c_{b,1} \in \mathcal{C}_b$.

$$\mathsf{CircuitPrivacy}(c_{b,0}) \approx_s \mathsf{CircuitPrivacy}(c_{b,1}).$$

3 Proof of Knowledge of an Encryption

As noted in the Introduction, the method of Cramer, Damgård, and Nielsen [CDN01] requires an honest-verifier zero-knowledge proof of knowledge of encrypted values for the threshold schemes that they employ. We provide a weaker 2-round solution to that requirement, which alas, is not zero-knowledge, but also does not release any information about the bit being discussed (we formalize this below). Moreover, our construction is black-box in the underlying circuit-private FHE scheme.

We construct this proof through a two-step process. At a high-level, instead of encrypting a bit b, we use a specific $(n, n/2+2)$ verifiable secret sharing scheme to generate n shares of b and encrypt those shares.[5] In order to give a proof of knowledge of the encryption of b, we allow a verifier to select $n/2+1$ of the encryptions of shares of b, and then direct the Prover to reveal the randomness used to encrypt those shares. To extract the bit, our extractor rewinds the proof and selects an alternate $n/2+1$ shares, so that with high probability, it can use $n/2+2$ shares to reconstruct b, and only b due to the verifiability of the secret sharing scheme. The problem with this approach is that revealing the randomness for an encryption raises selective decommitment issues. We use techniques from Hemenway et al. [HLOV11] to construct a bit-wise Indistinguishable Selective-Opening Secure encryption scheme from our threshold fully-homomorphic scheme. We can then use it to bitwise encrypt the VSS shares.

We note that the encryptions of the shares under the bit-wise Indistinguishable Selective-Opening Secure scheme, is not itself a homomorphic encryption scheme. For example, we cannot multiply directly two sets of shares encoding b_0 and b_1 and expect the result to encode $b_0 \cdot b_1$. However, the individual encrypted bits are still properly encoded ciphertexts under the FHE scheme that have a circuit-privacy evaluation function applied to them. Intuitively, therefore, we can homomorphically evaluate the reveal function of the secret sharing scheme to get a single encryption representing the reconstituted bit. This encryption can then be used to homomorphically evaluate the function as in Cramer et al. [CDN01]. There is however a snag: in principle, once the circuit-privacy function has been applied to a ciphertext, it may no longer be able to have homomorphic operations applied to it, as this is not guaranteed by the definition.[6] However, this problem is easily surmounted by applying Gentry's bootstrapping technique (cf. Defn 1) to re-encode the selective-opening secure schemes into ciphertexts which can have homomorphic operations applied to them, and thus the VSS's reveal algorithm can be applied to the individual bits of the shares, resulting in ciphertext of the encoded bit, which is in the ciphertext space of the TFHE scheme.

Using FHE to construct a Selective Opening Encryption Scheme. Hemenway et al. [HLOV11] show how any re-randomizable encryption scheme can be used to construct a natural lossy encryption scheme and thus, by the result of Bellare et al. [BHY09], is secure against indistinguishable selective opening attacks.

Since the Hemenway and Ostrovsky construction relies on re-randomization, they suggest that the distribution of a "fresh" encryption of a message should be

[5] We use a verifiable secret sharing scheme with a $n/2+2$ threshold to simplify the proof of the VSS, thus $|T| = n/2+1$ is chosen to be right under the threshold of the VSS, as one might expect.

[6] Further, in practice, with known schemes, these ciphertexts have too much noise in them to allow further homomorphic operations without sacrificing decryption correctness.

statistically close to a rerandomization of a fixed message. They point out that all homomorphic encryption schemes up to that point achieved this property by adding an encryption of 0 to the current message. While this property was true of all schemes at the time, it is not actually true of the known fully homomorphic encryption schemes, because each time we add an encrypted message to another we increase the amount of noise that is embedded in the ciphertexts, and thus fresh encryptions have less noise than encryptions that have had operations (such as addition) applied to them. Fortunately, the property they state is overly strong, and a simple observation shows that for their construction to go through they only require that the distributions

$$\{r \leftarrow R : E_{pk}(,0,r) \boxplus E_{pk}(m,r_0)\} \approx_s \{r \leftarrow R : E_{pk}(0,r) \boxplus E_{pk}(m,r_1)\},$$

for all public-keys pk, messages m and random strings r_0 and r_1 where $\boxplus$ is the homomorphic addition operation. However, it is simple to see that even these two distributions are not statistically close for the fully homomorphic encryption schemes that have been proposed. Fortunately, both schemes under consideration have rerandomization functions built to ensure *Circuit-Privacy*, as is defined in [Gen09b] and Def. 3.

Construction of a SOA from Lossy. We generate a public-key for the Lossy scheme by generating a traditional public-key and secret-key for the TFHE, and then we augment the public-key with two labeled ciphertexts c_0 and c_1, representing encryptions of 0 and 1. Now, to encrypt a bit b, we take c_b, and rerandomize it using the circuit-privacy function (In comparison, Hemenway and Ostrovsky add an encryption of the bit 0). Decryption works as it does in the FHE scheme. The lossy key generator simply has c_1 represent an encryption of 0 instead of 1. By the IND-CPA security of the TFHE scheme, the keys are indistinguishable. The scheme is formally described below.

Key Generation $G'(1^k, b), b \in \{\mathsf{INJ}, \mathsf{LOSSY}\}$**:** Let $(\mathrm{PK}, \mathrm{SK}) \leftarrow G(1^k), c_0 \leftarrow E(\mathrm{PK}, 0)$, $c_1 \leftarrow E(\mathrm{PK}, 1)$ and $c_1' \leftarrow E(\mathrm{PK}, 0)$. If $b = \mathsf{INJ}$ Output $\mathrm{PK}' = (pk, c_0, c_1)$ and $\mathrm{SK}' = \mathrm{SK}$, else when $b = \mathsf{LOSSY}$ output $\mathrm{PK}' = (\mathrm{PK}, c_0, c_1')$ and $\mathrm{SK}' = \mathrm{SK}$.

Encryption $E'(\mathbf{PK}' = (\mathbf{PK}, c_0, c_1), b)$**:** Output $\mathsf{ReRand}(c_b)$.

Decryption $D'(\mathbf{SK}, c)$**:** Output $D(\mathrm{SK}, c)$.

Theorem 1. *If (G, E, D) is a circuit-private FHE, then the blackbox construction (G', E', D') described above is an IND-SO-SEC secure encryption scheme.*

Proof. Follows from [HLOV11] and [BHY09].

Modifying the SOA-secure Encryption Scheme to Support POKs. Again, in order to be able to provide a proof of knowledge that a party has knowledge of the value encrypted, we need to provide a POK. We will show a 2-round public-coin proof of knowledge of the encrypted bit based on any selective opening secure scheme. The protocol is neither zero-knowledge nor witness indistinguishable, but does maintain secrecy of the encrypted bit. First, we encrypt bits using the following protocol. Let $\Pi' = (G', E', D')$ be the selective-opening attack secure scheme described in Thm. 1. We construct a new encryption scheme $\hat{\Pi} = (\hat{G}, \hat{E}, \hat{D})$ to encode bits as follows. We define $\hat{G} = G'$, and present the algorithms for $\hat{E}$ and $\hat{D}$ below. Refer to a full version of our work [MSas11] for the standard definitions of the Verifiable Secret Sharing algorithms.

```
$\hat{E}(\mathrm{PK}, b, r)$
    $(s_1, ..., s_n) \leftarrow \mathit{VSShare}_{(n,n/2+2)}(b)$
    Let $M$ be the $n \times n$ matrix
    representation of shares $(s_1, \ldots, s_n)$
    $c_{i,j} = E'(\mathrm{PK}, M_{i,j}, r_{i,j})$
    Output $\mathbf{C} = \{c_{i,j}\}_{i,j \in n}$
```

```
$\hat{D}(\mathrm{SK}, \mathbf{C})$
    $M = \{M_{i,j}\}_{i,j\in[n]} \leftarrow D'(\mathrm{SK}, \mathbf{C})$
    Let $(s_1, \ldots, s_n)$ be the shares
    corresponding to matrix $M$.
    $T' = \{t | 1 \leq t \leq n$ share $s_t$
            is $n/2 + 2$ -consistent$\}$
    If $|T'| < n/2 + 2$ output $\perp$.
    Let $T \subseteq T'$ s.t. $|T| = n/2 + 2$.
    Output $\mathit{VSReveal}_{(n,n/2+2)}(\{s_{t_i}\})_{t_i \in T}$
```

Hidden Bit POK. Given a ciphertext $\mathbf{C} = \{c_{i,j}\}_{i,j\in n}$ output by encryption algorithm $\hat{E}$ and the random coins $\boldsymbol{r}$ used to generate it, we show how to perform a two-round proof of knowledge of the encrypted bit $\hat{D}(\mathrm{SK}, \mathbf{C})$. P will prove that it has knowledge of the underlying shares of the verifiable secret-sharing scheme that have been encrypted. In order to do this, the verifier sends a random challenge of indices $T \subset [n]$, where $|T| = n/2+1$. The encryptor then decommits to these encryptions by providing the random-bits used to encrypt each share of the bit. If each bit decommits successfully, and the result is $n/2 + 1$ valid shares to the VSS, then the verifier accepts.

```
Prover$(\mathrm{PK}, \mathbf{C} = \{c_{i,j}\}_{i,j\in[n]}$                              Verifier$(\mathrm{PK}, \mathbf{C} = \{\mathbf{c_{i,j}}\}_{\mathbf{i,j\in[n]}})$
  $= \hat{E}(\mathrm{PK}, b, r), M, r)$
Let $c_{i,j} = E'(\mathrm{PK}, M_{i,j}, r_{i,j})$     $\xleftarrow{T}$     $T \leftarrow \{S | S \subset [n] \wedge |S| = \frac{n}{2} + 1\}$
        $\xrightarrow{\{M_{i,x}, r_{i,x}, M_{x,i}, r_{x,i}\}_{i\in T, x\in[n]}}$     if $\exists i, j$: $c_{ij} \neq E'(\mathrm{PK}, M_{i,j}, r_{i,j})$,
                                                          output $\perp$.
                                                     Output 1.
```

```
$\mathit{Extractor}(\mathbf{C}, \mathrm{PK}, U_1 = \{M_{i,x}, r_{i,x}, M_{x,i}, r_{x,i}\}_{i\in T_1, x\in[n]}, U_2 = \{M_{i,x}, r_{i,x}, M_{x,i}, r_{x,i}\}_{i\in T_2, x\in[n]})$
Let $T = T_1 \cup T_2,\ U = U_1 \cup U_2$
If $|T| < n/2$ output $\perp$.
If $\exists i \in T, x \in [n]$ s.t. $E'(\mathrm{PK}, M_{i,x}, r_{i,x}) \neq c_{i,x}$ or $E'(\mathrm{PK}, M_{x,i}, r_{x,i}) \neq c_{x,i}$ output $\perp$.
For each $i \in T$ reconstruct its corresponding share $s_i$.
Output $\mathit{VSReveal}_{(n,n/2+2)}(s_{r_1}, .., s_{r_{\frac{n}{2}}})$, where $r_1, .., r_{\frac{n}{2}}$ are the smallest indices in $T$.
```

Completeness. Follows by inspection.
Extractability (Soundness). Soundness follows from an extractor.

Theorem 2. *For all sufficiently large n, for all $d > 0$, for all $(SK, PK) \leftarrow \hat{G}$, for all 'ciphertext' inputs C, and provers P', if $(P', V)(C = \{c_{i,j}\}_{i,j \in [n]}, PK)$ accepts with probability $1/n^d$, then there exists a probabilistic polynomial time extractor that, with all but negligible probability, outputs a set of decommitments to all ciphertexts for a given set of indices $L = \{\ell_1, \cdots, \ell_{n/2+2}\} \subseteq [n]$ that constitute shares $S = \{s_{\ell_1}, ..., s_{\ell_{n/2+2}}\}$ such that $VSReveal_{(n,n/2+2)}(s_{\ell_1}, ..., s_{\ell_{n/2+2}}) = \hat{D}(SK, C)$.*

Definition 4. *We say an $n \times n$ matrix representation of shares has t-consistent indices if there is a set S of size t such that for each $i \in S$, each row i and column i is $n/2 + 2$ consistent.*

Proof. Given the ability to rewind the prover-verifier protocol, we can extract the encrypted bit by recovering enough shares of the VSS scheme. We continue to execute the prover/verifier protocol until we get two distinct separate accepting proofs. It is a simple observation that except with exponentially small probability, we will succeed in $O(n^{d+1})$ rewinds. Let (T_1, U_1) and (T_2, U_2) be the flows in the first and second accepting proofs, respectively. By the security of the commitment scheme (here we are using our encryption scheme as a simple commitment scheme), the probability that there is a ciphertext $c_{i,j}$ that is ever decommitted to in two distinct fashions is negligble.

We feed these inputs in to *Extractor*. If there is not a valid encryption of a bit (fewer than $n/2 + 2$ committed and consistent shares), then by Lemma 1, the probability that the verifier outputs anything other than $\perp$ is less than $\frac{1}{\binom{n}{n/2+2}}$ which grows exponentially small.

Given the decommitments of the shares $\{s_i\}_{i \in T_i}$ for different randomly chosen set of indices T_1 and T_2, note these sets are not the same by selection, and therefore there is no chance that $\perp$ is output by the extractor. Next the extractor executes a $VSReveal_{(n,n/2+2)}$ command. However, this is not necessarily over the same shares as would be revealed in a legitimate decryption. We need to ensure that no matter which of the rewound and newly played legitimate traces we receive, we are going to reveal the same encrypted bit, with all but negligible probability. That is, we need to ensure that $VSReveal_{(n,n/2+2)}(s_{r_1}, ..., s_{r_{n/2}}) = VSReveal_{(n,n/2+2)}(s_1, ..., s_{n/2})$. This is the case, as shown in Lemma 2 because of the verifiable properties of the secret sharing scheme ensures that even in the case of a corrupted dealer (improper ciphertext encoding of shares) then all honest players will reveal the same value, with all but negligible probability. Therefore, with all but negligible probability we have that the extractor outputs the same value as $D(\mathrm{SK}, \boldsymbol{c})$.

Lemma 1. *Let M be an $n \times n$ matrix with at most $n/2 + 1$ consistent indices. The probability that any $n/2+1$ randomly selected indices (without replacement) choose a set of $n/2 + 1$ consistent indices is no more than*

$$1/\binom{n}{n/2+1}.$$

Proof. There can be at most 1 set of size $(n/2+1)$ that is $(n/2+1)$ consistent in an $n \times n$ matrix. The lemma follows by computing the probability of choosing this one set from a set of n objects.

Lemma 2. *Let M be $n \times n$ matrix representation of shares. Let $S, T \subseteq [n]$, $|S| = |T| = n/2+2$, $S \neq T$, and the rows $R_S = \{r_i\}_{i \in S}$, $R_T = \{r_i\}_{i \in T}$ and columns $C_S = \{c_i\}_{i \in S}$, $C_T = \{c_i\}_{i \in T}$ are all $n/2+2$-consistent. Let $s = (s_1, ..., s_{n/2+2})$ and $t = (t_1, ..., t_{n/2+2})$ be the shares drawn from M corresponding to the sets of indices S and T respectively. Then*

$$VSReveal_{(n,n/2+2)}(s_1, ..., s_{n/2+1}) = VSReveal_{(n,n/2+2)}(t_1, ..., t_{n/2+1})$$

Proof. Note that $VSReveal_{(n,n/2+2)}$ will never output $\perp$ under our conditions, so all that we need do is show that f will interpolate to the same value in both cases.

We know that the rows $R_T = \{r_i\}_{i \in T}$ and columns $C_R = \{c_i\}_{i \in T}$ are all $(n/2+2)$-consistent. Choose any $j \in S \setminus T$. Let $T = \{t_1, \ldots, t_{n/2+2}\}$. Consider $c_j = (c_{1,j}, c_{2,j}, \ldots, c_{n,j})^T$. Since c_j is $n/2+2$-consistent, the points $(c_{t_1,j}, t_1), \ldots, (c_{t_{n/2+1},j}, t_{n/2+2})$, interpolate to a unique univariate degree $n/2+1$ polynomial (i.e. $f(x,j)$). This defines $(c_{1,j}, c_{2,j}, \ldots, c_{n,j})^T$, so the column j must be consistent with T. Since the jth column was an arbitrary column in S different from those in T, all such columns must be consistent with the rows defined be T. A symmetric argument shows that rows selected by S must be consistent with the columns selected by T. Therefore, both sets are consistent in that they define the same polynomials. Therefore, interpolation in $VSReveal_{(n,n/2+2)}$ will result in the same output.

Hidden Bit. We show that no efficient cheating verifier can predict the bit b, when given $C = \hat{E}(\mathrm{PK}, b, r)$ as a theorem for which we are engaging in a POK.

Theorem 3. *For every P.P.T. adversary $A = (A_1, A_2)$, there exists a negligible function μ such that $\Pr[HB_A(1^k) = 1] \leq 1/2 + \mu(k)$, where HB_A is defined below:*

```
$HB_A(1^k)$
    $(PK, SK) \leftarrow \hat{G}(1^k)$
    $b \in \{0,1\}$
    $C = \{c_{i,j}\}_{i,j \in [n]} = \hat{E}(PK, b)$  where  $c_{i,j} = E'(PK, M_{i,j}, r_{i,j})$ are SOA-sec.
    $(T, \sigma) \leftarrow A_1(PK, C)$ where $T \subset [n]$, $|T| = n/2+1$.
    $b' \leftarrow A_2(\sigma, (M_{i,j}, r_{i,j})_{i,j \in T})$
    Output $1$ iff $b = b'$
```

Proof. This follows directly from the IND-SO-SEC security of $\Pi' = (G', E', D')$. Suppose an adversary $A = (A_1, A_2)$ breaks the hidden bit security of the protocol. That is for some $d > 0$ and infinitely many k: $\Pr[HB_A(1^k) = 1] \geq 1/2 + 1/k^d$. We use it to build an adversary $B = (B_1, B_2)$ and message selector M that breaks the IND-SO-SEC security (cf. Defn. in [BHY09] or [MSas11]) of $\Pi' = (G', E', D')$. The message selector M chooses a random bit b, let

$(s_1, ..., s_n) \leftarrow \mathit{VSShare}_{(n,n/2+2)}(b)$, and let $\mathbf{M}$ be the $n \times n$ matrix that represents the shares $(s_1, \ldots, s_n)$ according to the ECC representation of the VSS. Output $\mathbf{M}$.

The adversary $B_1\left(\mathrm{PK}, (E(\mathrm{PK}, \mathbf{M_{i,j}}, \mathbf{r_{i,j}}))_{i,j \in n}\right)$ for the IND-SO-SEC experiment simulates $(T, \sigma) \leftarrow A_1(\mathrm{PK}, C = (E(\mathrm{PK}, \mathbf{M_{i,j}}, \mathbf{r_{i,j}}))$, and outputs $I = \{(i,j) | i,j \in n, i \in T \text{ or } j \in T\}$ and $\sigma' = (T, \sigma)$. Recall by the definition of A_1, $|T| = n/2 + 1$.

The conditional message selector $M_{I,\boldsymbol{m}[I]}$ from the SOA security definition finds a random bi-variate polynomial of degree $n/2 + 1$ in each variable over the field F such that $f(0,0) \in \{0,1\}$ and for each $(i,j) \in I$, it holds that $f(i,j) = \mathbf{M}_{i,j}$. Since $|T| = n/2 + 1$, and thus we have effectively release $n/2 + 1$ shares for a VSS scheme that requires $n/2 + 2$ for reconstruction, the information secrecy property of the VSS guarantees there are exactly the same number of such selections for the case $f(0,0) = 0$ and $f(0,0) = 1$. $M_{I,\boldsymbol{m}[I]}$ outputs $\{f(i,j)\}_{1 \leq, i,j \leq n}$.

The adversary $B_2(\sigma, (M_{i,j}, r_{i,j})_{(i,j) \in I}, \mathbf{M}^*)$ computes the shares $(s_1^*, ..., s_n^*)$ that correspond to $\mathbf{M}^*$, and runs $\mathit{VSReveal}_{(n,n/2+2)}(s_1^*, .., s_n^*) = b'$, it then executes $b \leftarrow A_2(\sigma, (m_{i,j}, r_{i,j})_{(i,j) \in I})$ and outputs 1 iff $b = b'$.

Now consider $\Pr[B_{\Pi}^{\mathsf{Ind\text{-}SO\text{-}Real}}(1^k) = 1]$, this is a perfect simulation of $HB_A(1^k)$, and therefore by the assumption that A breaks the hidden-bit security, the term must exceed $1/2 + \epsilon$, where $\epsilon \geq 1/k^c$. In contrast, consider $\Pr[B_{\Pi}^{\mathsf{Ind\text{-}SO\text{-}Ideal}}(1^k) = 1]$. In the case that $\mathit{VSReveal}_{(n,n/2+2)}(s_1^*, \ldots, s_n^*) = \mathit{VSReveal}_{(n,n/2+2)}(s_1, \ldots, s_n)$, which occurs with probability exactly 1/2, it is again a perfect simulation of $HB_A(1^k)$, and so the experiment outputs 1 with probability $1/2 + \epsilon$. In contrast, when $\mathit{VSReveal}_{(n,n/2+2)}(s^*_, 1..., s_n^*)) \neq \mathit{VSReveal}_{(n,n/2+2)}(s_1, ..., s_n)$, then we know that A_2 outputs $\mathit{VSReveal}_{(n,n/2+2)}(s_1, ..., s_n)$ with probability $1/2 + \epsilon$, and so B_2 outputs 1 with probability $1 - (1/2 + \epsilon) = 1/2 - \epsilon$. Therefore, $\Pr[B_{\Pi}^{\mathsf{Ind\text{-}SO\text{-}Ideal}}(1^k) = 1] = (1/2)(1/2 + \epsilon + 1/2 - \epsilon) = 1/2$. Therefore, $\Pr[B_{\Pi}^{\mathsf{Ind\text{-}SO\text{-}Real}}(1^k) = 1] - \Pr[B_{\Pi}^{\mathsf{Ind\text{-}SO\text{-}Ideal}}(1^k) = 1] = 1/2 + \epsilon - 1/2 \geq 1/k^c$, breaking IND-SO-SEC security.

Using the SOA Ciphertexts in a Secure Multiparty Computation Protocol. In our SMC construction, we encode all users' inputs using the POK scheme above. The encrypted inputs are sent to the other parties. After each party's input has been confirmed with a proof of knowledge, the parties homomorphically evaluate the different ciphertexts to get an appropriate encrypted output. However, as explained before, the POK encryptions are not themselves homomorphic. To solve this problem we use Gentry's bootstrapping technique. Bootstrapping lets us take a ciphertext in an FHE scheme with any amount of noise that still allows for proper decryption (specially, this is potentially more noise than is permissible to perform any extra homomorphic operations without destroying the correctness of the ciphertext), and output a new ciphertext in the FHE scheme, of the same value, but with a small enough amount of noise that it can be properly computed on through the use of the FHE's evaluation function. Given a ciphertext $C = \{c_{i,j}\}_{i,j \in [n]}$ in the POK scheme, each $c_{i,j}$ is a ciphertext from a lossy encryption scheme. To convert C into a corresponding encryption $c^\dagger$ in the TFHE scheme we do the following: We bootstrap each $c_{i,j}$ which is simply a TFHE ciphertext that has had the circuit-privacy function applied to it—thus containing potentially too much noise to apply further homomorphic operations to, but not so much that it decrypts improperly— to receive the corresponding lower-noise TFHE ciphertext $c'_{i,j}$. The c' ciphertexts can now be evaluated in the THFE eval function, and in particular we can use the TFHE eval function, to evaluate $\mathit{VSReveal}_{(n,n/2+2)}$. The result of this evaluation is the ciphertext $C^\dagger$ corresponding to the output.

Protocols vs. Algorithms. We note that there is one technical issue that needs to be resolved, which is that in this section we have described the key generation and decryption algorithms as stand-alone algorithms, rather than protocols. For our purposes, we need a joint protocol for key generation and decryption. For this reason, we need to modify our key generation algorithm in the TFHE scheme to include an encryption of the bits 0 and 1 in the public-key. These values allow the parties to encrypt under the SOA secure encryption scheme $\hat{\Pi}$. The SOA secure scheme does not modify the decryption algorithm, so there is no need for modification to the decryption protocol.

4 Secure Multiparty Computation

We follow the Cramer et al. [CDN01] approach for constructing a multi-party computation protocol based on threshold cryptography. Our biggest changes are that we do not need a protocol for multiplication, we use a different approach for proving knowledge of encryption, and we explicitly describe a key generation phase whereas it is assumed as an external setup in [CDN01]. Since our solution requires less interaction among the parties, our simulation argument is simpler than the argument from [CDN01].

We use the standard simulation-based definition of stand-alone secure multiparty computation. We assume the existence of a standard n-party CoinFlipping protocol which guarantees soundness in the presence of $< n/2$ adversaries: namely, for any minority set of adversaries, the protocol guarantees that the distribution is still statistically close to uniform. Such a protocol can be easily constructed based on the existence of hiding commitments. (Unlike [CDN01], we do not need this coin flipping protocol to be simulatable.). See our full version [MSas11] for a definition of the real/ideal paradigm for secure multi-party computation from [CDN01] and [IKK$^+$11]. In this section the TFHE scheme used is denoted $\tilde{\Pi} = (\tilde{G}, \tilde{E}, \tilde{D}, \mathbf{Eval})$.

We assume that the players can communicate via an authenticated broadcast channel and via point-to-point private and authenticated channels (which may in turn be implemented using signatures, public-key encryption, etc.)

Protocol 1. Each party holds private input x_i; the parties jointly compute $f(x_1, \ldots, x_n)$.

1: Party P_i receives as input $(1^k, n, x_i)$. (We assume the adversary receives as input $1^k, n$, a set of corrupted parties C and the inputs $\{x_c\}_{c \in X}$ for the corrupted parties, and auxiliary information.)

2: Players run the TFHE key generation subprotocol $\tilde{G}(\eta, \tau, \rho, \theta, \Theta, \kappa)$ to generate a public-key $\tilde{\mathrm{PK}}$ and shares of the secret for the threshold scheme $\tilde{\Pi}$. At the end of this step, player p_i holds share SK_i of the secret-key SK. If the sub-protocol halts prematurely, then players halt and output $\perp$.

3: The players take sequential turns sharing their input using the encryption scheme $\hat{\Pi}$ that is constructed from Π (see §3). More specifically, for $i \in [n]$, player P_i broadcasts $c_{i,j} \leftarrow \hat{E}(\tilde{\mathrm{PK}}, x_{i,j})$. Then all of the players run a standard CoinFlipping protocol to generate a random string r_i. Player P_i now interprets r_i as n strings $r_{i,1}, \ldots, r_{i,n}$ and uses coins $r_{i,j}$ as the random coins to run Verifier(PK,$c_{i,j}$) (see §3) of the Hidden Bit POK protocol on input $c_{i,j}$ for each bit $j \in [n]$ of input x_i. Player P_i runs the corresponding Prover algorithm on $c_{i,j}$ using the random coins used to generate $c_{i,j}$ as the witness, and broadcasts the Prover message. The remaining players also execute the Verifier algorithm using the same random coins and verify that the first message is consistent and the second message is accepted. If player P_i fails the POK protocol, then P_i is excluded from the rest of the protocol, and the remaining players that have not been excluded use a canonical encryption of 0 as the input for P_i (e.g., they use $\tilde{E}(\tilde{\mathrm{PK}}, 0; 0)$ as each input bit).

4: The players that have not been excluded locally run $\mathbf{Eval}(\tilde{\mathrm{PK}}, c_{1,1}, \ldots, c_{n,n}, \tilde{f})$ where the function $\tilde{f}$ first transforms the input ciphertexts encrypted under $\hat{\Pi}$ into ones for scheme $\tilde{\Pi}$. This is done by homomorphically evaluating the decryption procedure described in §3 (i.e. bootstrapping, see Defn. 1).(Note: All of the ciphertexts in $c_{i,j}$ have a large degree of noise in them due to the circuit-privacy call that was used to rerandomize the ciphertexts. Therefore, the first thing that is done is that the ciphertexts are re-encoded with less noise using the same procedure as FHE bootstrapping.) Next, compute ciphertext z_i of the result $f(x_1, \ldots, x_n)$. Note that each player can complete this step using only local information (since the public-key for the FHE includes all the information needed for evaluation).

5: Each player P_i that has not been excluded broadcasts the ciphertext z_i computed in the previous step. Each player then locally computes the majority of the broadcasts as ciphertext z'. A majority is guaranteed to exist since the malicious players form a minority and **Eval** is deterministic. Any player whose broadcast differs from the majority is excluded from the remaining portion of the protocol.

6: Players p_i that have not been excluded run the distributed subprotocol $\tilde{D}(z', \mathrm{SK}_1, \ldots, \mathrm{SK}_n)$ using input z' and their local share SK_i. The output of the protocol is taken as the output.

Theorem 4. *Let π be Protocol 8 for a function f, and fix $s \in \{1, \ldots, n/2\}$. If $\mathit{\Pi}$ is a circuit-private TFHE encryption scheme, then for any ppt adversary A, there exists a ppt adversary A' such that for every polynomial-size circuit family $Z = Z_k$ corrupting a minority of parties the following is negligible:*

$$\left|\Pr\left[\mathrm{REAL}_{\pi,A,Z}(k) = 1\right] - \Pr\left[\mathrm{IDEAL}_{f,A',Z}(k) = 1\right]\right|.$$

See full version for details.

5 Threshold FHE for the Integers

In this section we briefly highlight the construction of a TFHE scheme $\tilde{\mathit{\Pi}} = (\tilde{G}, \tilde{E}, \tilde{D}, \tilde{\mathbf{Eval}})$ from the FHE scheme $\mathit{\Pi} = (G, E, D, \mathbf{Eval})$ based on the Approximate-GCD problem described by [vDGHV10]. The details are presented in our full version online. We point out that in any such transformation $\tilde{E} = E$ and $\tilde{\mathbf{Eval}} = \mathbf{Eval}$, and thus we only need to describe protocols for computing $\tilde{G}$ and $\tilde{D}$.

Sharing the Public and Secret-Key. Recall the secret-key p for the "somewhat homomorphic encryption scheme" is an odd η-bit integer. To sample p in a distributed fashion, we notice that the bits p_0 and $p_{\eta-1}$ should be 1 whereas the rest of the bits $p_1, \ldots, p_{\eta-2}$ should be randomly shared. At the end, each player holds a share of p. We then extend techniques from [KLML05] to allow multiple parties who hold shares of p to compute shares of $1/p$ and $x_p = \lfloor 2^\kappa/p \rceil$.

Recall that the secret-key for $\mathit{\Pi}$ consists of a Θ-bit vector $\boldsymbol{s}$ with Hamming weight θ. Our first modification to $\mathit{\Pi}$ is to note that instead of θ, it suffices to select a vector with Hamming weight in the interval $\theta \pm \theta/4$. To verify this, note that the sparse subset-sum problem is assumed to be hard for $\theta = \Theta^\epsilon$ for $0 < \epsilon < 1$; our change does not violate this condition. Also, our new range of settings for θ does not increase the total degree of the decryption circuit by more than a factor of 2 and thus the condition that the decryption protocol is admissible is maintained (and thus the scheme is bootstrappable. See the computation on p.18 [vDGHV10].) Our approach for producing $\boldsymbol{s}$ is to securely generate a random number r_i in the range $[0, \Theta]$ for each s_i and setting $s_i = 1$ if $r_i \leq \theta$ and 0 otherwise.

The public-key consists of the vectors $\boldsymbol{x}$ and $\boldsymbol{u}$. Using $\boldsymbol{s}$ and x_p, we compute the vector $\boldsymbol{u}$ using the formula $\boldsymbol{u} = \sum_i s_i \cdot u_i \mod 2^{\kappa+1}$. These shares can be used to compute the vector $\boldsymbol{y}$.

Using bits of $1/p$ computed in previous steps, we generate the x_i's. Recall from the original public-key generation algorithm that we need to sample $x_i \leftarrow D_{\gamma,\rho}(p)$ for $i = 0, \ldots, \tau$. Intuitively, these x_i represent random encryptions of 0 that get added to our base encryption in the homomorphic scheme. Further, recall that

$$D_{\gamma,\rho}(p) = \{\text{choose } q \leftarrow Z \cap [0, 2^\gamma/p), r \leftarrow Z \cap (-2^\rho, 2^\rho) : \text{ output } x \leftarrow pq + r\}.$$

After sampling, the list should be relabeled so that x_0 is the largest. The key-generation process requires that the process is restarted if either x_0 is even or $x_0 - \lfloor x_0/p \rceil \cdot p$ is odd. Since $x_0 = pq + r$ is generated as directed for some random q and r and since p is an odd number, the requirement that x_0 is odd can be checked by inspecting the least significant bits of the q and r: If $q_0 + r_0 = 1$, then x_0 satisfies the first condition. To check the second condition, that $x_0 - \lfloor x_0/p \rceil \cdot p$ is an odd number, we observe that because of the constraints $-2^{\rho} < r < 2^{\rho}$ and $2^{\eta-1} \leq p < 2^{\eta}$, it follows that $-2^{\rho-\eta+1} < r/p < 2^{\rho-\eta+1}$.

Since $\rho = \lambda$ and $\eta = \tilde{O}(\lambda^2)$, therefore for all sufficiently large λ (if $\eta = \lambda^2$, then for $\lambda > 2$), $\lfloor r/p \rceil = 0$ and as a result r can be ignored. That is $\lfloor x_0/q \rceil = \lfloor pq + r/q \rceil = q + \lfloor r/q \rceil = q$. So $x_0 - \lfloor x_0/p \rceil \cdot p = x_0 - q \cdot p$. Because x_0 and p are both odd, q must be odd to make the term $x_0 - \lfloor x_0/p \rceil \cdot p$ even. These constraints imply that for x_0 to be odd and $x_0 - \lfloor x_0/p \rceil \cdot p$ to be even, then q must be even and r must be odd.

Computing encryptions of $\boldsymbol{s}$. One step in Gentry's paradigm for FHE construction requires the public-key to contain an encryption of the secret-key. We assume circular security of the underlying encryption scheme, as do van Dijk et al. [vDGHV10] and Gentry [Gen09b]. Towards this goal, we design a protocol that enables players who hold private shares of the secret-key (as well as the entire public-key) to compute an encryption of the secret-key under the public-key. Note this *cannot* be done trivially with homomorphic evaluation because the encrypted secret-key is in fact necessary to homomorphically evaluate circuits of an arbitrary depth, resulting in a circular requirement.

Recall that in Dijk et al. [vDGHV10], the encryption of m under the public-key $\langle x_0, \ldots, x_\tau \rangle$ computes as $[m + 2r + 2\sum_{i \in S} x_i]_{x_0}$, where $r \in (-2^{\rho'}, 2^{\rho'})$ and $S \subseteq \{1, \ldots, \tau\}$ is a random subset. Since both the x_i's and r can take negative values (as integers) whereas the computation is in a finite field, we need to somehow make sure the computation in the finite field result in the same integer value of the encryption of m. To resolve this issue, we compute the value *min* which is a unique value that satisfies the following two properties: 1) $\mathit{min} = 0 \bmod x_0$, and 2) for an arbitrary S and for our set of x_i's and any value of r, it would make the summation $m + 2r + 2\sum_{i \in S} x_i$ positive. Because the range of values that r can take is public, all users can compute *min* locally and agree on respective shares. Next, to encrypt the secret-key, all users generate shares for a set S and the shares for a value r. All users then add their shares of r, use shares in S to add in appropriate x_i's, and add *min*.See the full version for details.

Computing encryptions of 0 *and* 1 *for PK.* The same techniques from the previous step can be used to produce encryptions of random bits. These encryptions can then be collaboratively decrypted until both an encryption of 0 and an encryption of 1 are identified. These two ciphertexts can then be adjoined to the public-key—they are guaranteed to be well-formed and have the right amount of noise.

References

[AJLA+12] Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (2012)

[BDOZ11] Bendlin, R., Damgård, I., Orlandi, C., Zakarias, S.: Semi-homomorphic Encryption and Multiparty Computation. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 169–188. Springer, Heidelberg (2011)

[Bea91] Beaver, D.: Efficient Multiparty Protocols Using Circuit Randomization. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 420–432. Springer, Heidelberg (1992)

[BHY09] Bellare, M., Hofheinz, D., Yilek, S.: Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (2009)

[CDD+99] Cramer, R., Damgård, I., Dziembowski, S., Hirt, M., Rabin, T.: Efficient Multiparty Computations Secure against an Adaptive Adversary. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 311–326. Springer, Heidelberg (1999)

[CDN01] Cramer, R., Damgård, I.B., Nielsen, J.B.: Multiparty Computation from Threshold Homomorphic Encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)

[CDSMW08] Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Black-Box Construction of a Non-malleable Encryption Scheme from Any Semantically Secure One. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 427–444. Springer, Heidelberg (2008)

[DIK10] Damgård, I., Ishai, Y., Krøigaard, M.: Perfectly Secure Multiparty Computation and the Computational Overhead of Cryptography. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 445–465. Springer, Heidelberg (2010)

[Gen09a] Craig Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), `http://crypto.stanford.edu/craig`

[DPSZ12] Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (2012)

[Gen09b] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

[GLOV12] Goyal, V., Lee, C.-K., Ostrovsky, R., Visconti, I.: Constructing non-malleable commitments: A black-box approach. In: FOCS, pp. 51–60 (2012)

[HLOV11] Hemenway, B., Libert, B., Ostrovsky, R., Vergnaud, D.: Lossy Encryption: Constructions from General Assumptions and Efficient Selective Opening Chosen Ciphertext Security. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 70–88. Springer, Heidelberg (2011)

[GO94] Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. J. Cryptology 7(1), 1–32 (1994)

[IKK+11] Ishai, Y., Katz, J., Kushilevitz, E., Lindell, Y., Petrank, E.: On achieving the "best of both worlds" in secure multiparty computation. SIAM J. Comput. 40(1), 122–141 (2011)

[JJ00] Jakobsson, M., Juels, A.: Mix and Match: Secure Function Evaluation via Ciphertexts. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 162–177. Springer, Heidelberg (2000)

[KLML05] Kiltz, E., Leander, G., Malone-Lee, J.: Secure Computation of the Mean and Related Statistics. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 283–302. Springer, Heidelberg (2005)

[LATV11] López-Alt, A., Tromer, E., Vaikuntanathan, V.: Cloud-assisted multiparty computation from fully homomorphic encryption. IACR Cryptology ePrint Archive, 2011:663 (2011)

[MSas11] Myers, S., Sergi, M., Shelat, A.: Threshold fully homomorphic encryption and secure computation. Cryptology ePrint Archive, Report 2011/454 (2011), http://eprint.iacr.org/

[NN01] Naor, M., Nissim, K.: Communication preserving protocols for secure function evaluation. In: STOC, pp. 590–599 (2001)

[vDGHV10] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully Homomorphic Encryption over the Integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)

A Verifiable Secret-Sharing Scheme

A $\binom{n}{n/2+2}$ Verifiable Secret-Sharing scheme consists of a sharing algorithm which takes as input a secret s and produces n-shares $s_1, ..., s_n$. These shares have the property that for any $T \subset \{1, \ldots n\}$, $|T| < n/2 + 2$ it is the case that $\{s_i\}_{i \in T}$ is information theoretically independent from s. However, for any $S \subseteq \{1, \ldots n\}$, $|S| \geq n/2 + 2$, it is the case that the reveal algorithm, when given $\{s_i\}_{i \in S}$, can reconstruct s. In a traditional interactive setting we require that all non-cheating parties agree on the reconstructed secret. We use a modification of the Cramer et al. [CDD+99] verifiable secret sharing scheme; we do not need to deal with interactive adversaries, nor players, so the scheme is significantly simplified. We present the sharing and revealing algorithms in our full version.

Definition 5. *A vector $(e_1, ..., e_n) \in F_n$ is $n/2 + 2-$*consistent *if there exists a polynomial w of degree at most $n/2 + 1$ such that $w(i) = e_i$ for $0 \leq i < n$.*

Definition 6. *Given two shares $s_i = (i, \boldsymbol{a}_i = (a_{i1}, \ldots, a_{in}), \boldsymbol{b}_i = (b_{1i}, \ldots, b_{ni}))$ and $s_j = (j, \boldsymbol{a}_j(a_{j1}, \ldots, a_{jn}), \boldsymbol{b}_j = (b_{1j}, \ldots, b_{nj}))$, we say that they are* pairwise consistent *if $a_{ij} = b_{ij}$ and $a_{ji} = b_{ji}$.*

Definition 7. *For our purposes it is useful to note that given the $n \times n$ matrix*

$$\begin{bmatrix} f(1,1) & f(1,2) & \dots & f(1,n) \\ f(2,1) & f(2,2) & \dots & f(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ f(n,1) & f(n,2) & \dots & f(n,n) \end{bmatrix},$$

that a share s_i simply corresponds to the i^{th} row and column of the matrix. We will call this the matrix representation of the shares. Notice that when given in the matrix representation, any two shares are necessarily pairwise consistent. Given a set of n pairwise consistent shares $\boldsymbol{s} = (s_1, ..., s_n)$, we define $M_{\boldsymbol{s}}$ as the $n \times n$ matrix representation of the shares.