Spooky Encryption and Its Applications

Yevgeniy Dodis[1(✉)], Shai Halevi[2(✉)], Ron D. Rothblum[3(✉)], and Daniel Wichs[4(✉)]

[1] NYU, New York, NY, USA
dodis@cs.nyu.edu
[2] IBM Research, Yorktown Heights, NY, USA
shaih@alum.mit.edu
[3] MIT, Cambridge, MA, USA
rothblum@gmail.com
[4] Northeastern University, Boston, MA, USA
danwichs@gmail.com

Abstract. Consider encrypting n inputs under n independent public keys. Given the ciphertexts $\{c_i = \mathsf{Enc}_{\mathsf{pk}_i}(x_i)\}_i$, Alice outputs ciphertexts $c'_1, \ldots, c'_n$ that decrypt to $y_1, \ldots, y_n$ respectively. What relationships between the x_i's and y_i's can Alice induce?

Motivated by applications to delegating computations, Dwork et al. [11] showed that a semantically secure scheme disallows *signaling* in this setting, meaning that y_i cannot depend on x_j for $j \neq i$. On the other hand if the scheme is homomorphic then any *local* (component-wise) relationship is achievable, meaning that each y_i can be an arbitrary function of x_i. However, there are also relationships which are neither signaling nor local. Dwork et al. asked if it is possible to have encryption schemes that support such "spooky" relationships. Answering this question is the focus of our work.

Our first result shows that, under the LWE assumption, there exist encryption schemes supporting a large class of "spooky" relationships, which we call *additive function sharing* (AFS) spooky. In particular, for any polynomial-time function f, Alice can ensure that $y_1, \ldots, y_n$ are random subject to $\sum_{i=1}^{n} y_i = f(x_1, \ldots, x_n)$. For this result, the public keys all depend on common public randomness. Our second result shows that, assuming sub-exponentially hard indistinguishability obfuscation (iO) (and additional more standard assumptions), we can remove the common randomness and choose the public keys completely independently. Furthermore, in the case of $n = 2$ inputs, we get a scheme that supports an even larger class of spooky relationships.

We discuss several implications of AFS-spooky encryption. Firstly, it gives a strong counter-example to a method proposed by Aiello et al. [1] for building arguments for NP from homomorphic encryption. Secondly, it gives a simple 2-round multi-party computation protocol where, at the end of the first round, the parties can locally compute an additive secret sharing of the output. Lastly, it immediately yields a function secret sharing (FSS) scheme for all functions.

We also define a notion of *spooky-free* encryption, which ensures that no spooky relationship is achievable. We show that any non-malleable encryption scheme is spooky-free. Furthermore, we can construct spooky-free *homomorphic* encryption schemes from SNARKs, and it remains an open problem whether it is possible to do so from falsifiable assumptions.

1 Introduction

Imagine Alice and Bob, standing on different planets light years apart. They are "simultaneously" given some input bits x_1 and x_2 respectively, and must answer by outputting bits y_1 and y_2 respectively. Classical physics allows them to implement *local* (component-wise) strategies where y_1 is an arbitrary function of x_1 and y_2 is a function of x_2. On the other hand, the impossibility of faster-than-light communication disallows *signaling* strategies, meaning that the distribution of y_1 cannot depend on the value of x_2 and vice versa.

However, there are strategies that are neither local nor signaling. For example, perhaps Alice and Bob want to ensure that y_1, y_2 are random bits subject to $y_1 \oplus y_2 = x_1 \wedge x_2$. In this case, the distribution of y_1 does not depend on x_2 (and vice versa) so the strategy is not signaling, but it's also not local. Surprisingly some such strategies which are neither signaling nor local are achievable using quantum mechanics, if Alice and Bob share an entangled quantum state. Einstein referred to this phenomenon as "spooky action at a distance".

In this work, we consider an analogous scenario, first considered by Dwork et al. [11], where the separation between x_1, x_2 is enforced not via physical distance but by encrypting these bits under two independent public keys.[1] Here Alice gets the two ciphertexts $c_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}_1}(x_1), c_2 \leftarrow \mathsf{Enc}_{\mathsf{pk}_2}(x_2)$, and outputs two other ciphertexts c'_1, c'_2 which are decrypted as $y_i \leftarrow \mathsf{Dec}_{\mathsf{sk}_i}(c'_i)$, $i = 1, 2$. As in the physical analogy, here too we can rule out signaling strategies (if the encryption is semantically secure), and can implement local strategies (if the encryption is homomorphic). But can we replace the entangled state from above by a special "spooky encryption scheme" that would allow Alice to implement spooky strategies? Answering this question is the focus of this work, and we obtain the following results:

- Assuming the hardness of learning with errors (LWE), there exists a secure encryption scheme in which Alice can implement a wide class of spooky strategies that we call *additive function sharing* (AFS) spooky. Namely, for any two-argument function $f : (\{0,1\}^*)^2 \to \{0,1\}$, Alice can convert encryption of inputs $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(x_i)$ to encryption of outputs $y_i \leftarrow \mathsf{Dec}_{\mathsf{sk}_i}(c'_i)$, ensuring that $y_1 \oplus y_2 = f(x_1, x_2)$, except for a small error probability.
  This construction, described in Sect. 3, uses the LWE-based multi-key FHE schemes from [7,22,26], and it inherits from these multi-key scheme their dependence on a common random string.
- In Sect. 4 we describe a spooky scheme that supports arbitrary two-input spooky relations on short inputs, as well as a very wide class of two-input spooky relations on long inputs. This construction uses probabilistic indistinguishability obfuscation (piO), which is an extension of iO to probabilistic

[1] Dwork *et al.* considered PIR rather than encryption, but the translation is immediate.

circuits recently introduced by Canetti *et al.* [6], in conjunction with lossy encryption schemes which are homomorphic and ensure circuit privacy against malicious adversaries. This construction works in the plain model without common-random string and has no error, and it can be realized based on exponentially strong iO, exponentially strong PRFs, and DDH.

- In Sect. 5 we describe a transformation from a scheme that supports only two-input spooky relations on one-bit inputs to one that supports AFS spooky relations on arbitrary number of inputs (of arbitrarily length each). This transformation can be applied to both our LWE-based and piO-based constructions from above.
- We show several implications of (AFS-)spooky encryption. On a negative, it gives a strong counter-example to a method proposed by Aiello et al. [1] for building succinct arguments for NP from homomorphic encryption[2], resolving a question posted by [11]. On a positive, it immediately yields a function secret sharing (FSS) scheme for all functions [4,15], and also gives a simple 2-round multi-party computation protocol where, at the end of the first round, the parties can locally compute an additive secret sharing of the output. These application are discussed in Sect. 6.
- We also study in Sect. 7 the concept of *spooky free* encryption, i.e., an encryption scheme where we can prove that no spooky strategy is feasible. We show that any non-malleable encryption scheme is spooky-free, and also build spooky-free *homomorphic* encryption schemes from SNARKs. It remains an open problem to construct spooky-free homomorphic encryption under more standard assumptions. Spooky-free homomorphic encryption can be used to instantiate the approach of Aiello et al. to get succinct arguments for NP.

1.1 Technical Overview

LWE-Based Construction. Our LWE-based construction builds on the multi-key FHE schemes from [7,22,26]. In these schemes (after some syntactic massaging) secret keys and single-key ciphertexts are vectors in $\mathbb{Z}_q^n$, and decryption consists of computing $w = \langle \boldsymbol{s}, \boldsymbol{c} \rangle \bmod q$, then rounding to the nearest multiple of $q/2$, outputting zero if w is closer to 0 or one if w is closer to $q/2$.

These schemes, however, also support homomorphic computation across ciphertexts relative to different keys. Roughly, they feature a "lifting procedure" where a dimension-n ciphertext vector relative to one key $\boldsymbol{s}_i$ is "lifted" to a dimension ℓn vector $\boldsymbol{c'} = (\boldsymbol{c'}_1, \ldots, \boldsymbol{c'}_\ell)$ relative to the concatenated key $\boldsymbol{s'} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\ell)$ of dimension ℓn. These lifted ciphertexts can still be computed on, and the decryption procedure proceeds just as before, except using the higher-dimension vectors. Namely, to decrypt $\boldsymbol{c'}$ using $\boldsymbol{s'}$, one first computes the inner product $w' = \langle \boldsymbol{s'}, \boldsymbol{c'} \rangle$ modulo q, then rounds to the nearest multiple of $q/2$. In other words, we compute the individual inner products $w_i = \langle \boldsymbol{s}_i, \boldsymbol{c'}_i \rangle$, then add them all up and round to the nearest multiple of $q/2$.

[2] Although included in the ICALP conference proceedings, the article [1] was withdrawn before the conference and was not presented there.

We observe (cf. Lemma 1) that for the special case of two keys, $\ell = 2$, instead of adding the w_i's and then rounding, we can first round each w_i to the nearest multiple of $q/2$ and then add, and this yields the same result with high probability. Specifically, the error probability is proportional to the rounding error for the overall sum w'. This observation immediately yields additive function sharing (AFS) spooky encryption for two-argument functions: We use one of the schemes from [7,22,26] to encrypt the two arguments x_1, x_2 under two keys, then use the multi-key evaluation procedure to compute a multi-key ciphertext $\boldsymbol{c'} = (\boldsymbol{c'}_1, \boldsymbol{c'}_2)$ encrypting the value $f(x_1, x_2)$. Viewing each $\boldsymbol{c'}_i$ as a single-key ciphertext, we apply the usual decryption procedure to each of them, and the resulting two bits are an additive secret sharing of $f(x_1, x_2)$, except with a small error probability. The error probability can be made negligible by relying on LWE with a super-polynomial approximation factor.

piO-Based Construction. In Sect. 4 we show that using iO we can construct an AFS encryption scheme without CRS and without errors, and moreover we can support *arbitrary spooky relations* on two bits, not just additive sharing. For this overview, let us focus on the simpler task of constructing AFS spooky scheme for the multiplication function $\mathsf{MULT}(b_1, b_2) = b_1 \cdot b_2$.

The starting point of the construction takes a homomorphic encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ and adds to the public key an obfuscation of the randomized functionality that decrypts, computes the functions f, and re-encrypts secret-sharing of the result. Specifically, let us denote for any $x_1, y_1 \in \{0,1\}$ the function $f_{x_1,y_1}(x_2) = x_1 \cdot x_2 \oplus y_1$, and consider the following randomized program:

```
Program $P_{\mathsf{sk}_1,\mathsf{pk}_1}(c_1, \mathsf{pk}_2, c_2)$
1. $y_1 \leftarrow \{0,1\}$.                      4. $c'_2 = \mathsf{Eval}(\mathsf{pk}_2, f_{x_1,y_1}, c_2)$.
2. $c'_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}_1}(y_1)$.     5. Output $(c'_1, c'_2)$.
3. $x_1 = \mathsf{Dec}_{\mathsf{sk}_1}(c_1)$.
```

Given the two pairs $(\mathsf{pk}_1, \mathsf{Enc}_{\mathsf{pk}_1}(x_1))$, $(\mathsf{pk}_2, \mathsf{Enc}_{\mathsf{pk}_2}(x_2))$, and access to the program $P_{\mathsf{sk}_1,\mathsf{pk}_1}$, we can run $P_{\mathsf{sk}_1,\mathsf{pk}_1}(c_1, \mathsf{pk}_2, c_2)$ to get two ciphertexts c'_1 and c'_2, encrypting y_1, y_2, respectively, such that $y_1 \oplus y_2 = x_1 \cdot x_2$. We would like, therefore, to add an obfuscation of $P_{\mathsf{sk}_1,\mathsf{pk}_1}$ to the public key, thereby obtaining AFS spooky multiplication.

As described, however, this construction is not even secure when $P_{\mathsf{sk}_1,\mathsf{pk}_1}(c_1, \mathsf{pk}_2, c_2)$ is only accessed by a perfect black box. The reason is that if the underlying homomorphic encryption is not *circuit private*, then the evaluated ciphertext c'_2 could leak information about x_1. To fix this issue, we require the use of circuit-private homomorphic encryption in this construction. In fact, since the adversary could run the program $P_{\mathsf{sk}_1,\mathsf{pk}_1}(c_1, \mathsf{pk}_2, c_2)$ on arbitrary inputs of its choice, we need a stronger notion of *circuit privacy against malicious adversaries* [24], that guarantees privacy even if the public-key and ciphertext given to the evaluation algorithm are generated adversarially.

Using a malicious circuit private homomorphic encryption scheme, the construction above would be secure if the program $P_{\mathsf{sk}_1,\mathsf{pk}_1}(c_1, \mathsf{pk}_2, c_2)$ is accessed as a perfect black box (e.g., using VBB obfuscation). However, we would like to rely on the weaker notion of indistinguishability obfuscation (iO), or rather probabilistic iO [6] (since we are dealing with a randomized program). We need to somehow argue that the secret key sk_1 that is encoded within the program $P_{\mathsf{sk}_1,\mathsf{pk}_1}$ is hidden by the weaker obfuscation, and we do it using a technique from the work of Canetti *et al.* [6], employing a *lossy* encryption scheme.

We note that the construction above only uses homomorphic computations for single-bit functions (in addition to probabilistic iO), and there are only four such function (identity, negation, constant 0 and constant 1). A secure and malicious-circuit-private encryption scheme that supports these operations was constructed by Naor and Pinkas [23] based on the DDH assumption.

From 2-Spooky to $\boldsymbol{n}$-Spooky. Both the LWE and piO based constructions above only support two-argument spooky relations. Specifically the LWE-based scheme only supports AFS-spooky relations for two-argument functions, and the piO-based scheme supports a large class of spooky relations but again, only on two inputs. We extend the supported spooky relations by showing how to transform a scheme that supports (multiple hops of) AFS-spooky two-input multiplication and single-key additive homomorphism, into a leveled AFS spooky scheme for any number of inputs of any length.

The transformation is inspired by the Goldreich-Micali-Wigderson MPC protocol [16]: Suppose that we are given n public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, bit-by-bit encryptions of the input values $\mathsf{Enc}_{\mathsf{pk}_i}(x_i)$, and an arithmetic circuit $C : (\{0,1\}^*)^n \to \{0,1\}$ that we want to evaluate (i.e., to produce encrypted shares of $C(x_1, \ldots, x_n)$). We process the circuit gate by gate, while maintaining the invariant that for every wire w we produce ciphertexts $\mathsf{Enc}_{\mathsf{pk}_1}(w_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(w_n)$ such that $\oplus_{i\in[n]} w_i$ is equal to the wire w's value. The wires are processed inductively:

1. For an *input* wire holding a bit b, which is part of the j'th input x_j, we take the ciphertext c that encrypts b relative to pk_j, and append to it the ciphertexts $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(0)$ for all $i \neq j$. Clearly the ciphertexts $(c_1, \ldots, c_{j-1}, c, c_{j+1}, \ldots, c_n)$ are encryptions of an additive sharing of the wire's value b.
2. For an addition gate with input wires u, v and output wire w, by induction we already have $\mathsf{Enc}_{\mathsf{pk}_1}(u_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(u_n)$ and $\mathsf{Enc}_{\mathsf{pk}_1}(v_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(v_n)$. Using just an additive homomorphism on each key individually, we can produce $\mathsf{Enc}_{\mathsf{pk}_1}(u_1 \oplus v_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(u_n \oplus v_n)$ which is the desired secret sharing.
3. For a multiplication gate with input wires u, v and output wire w, again by induction we already have $\mathsf{Enc}_{\mathsf{pk}_1}(u_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(u_n)$ and $\mathsf{Enc}_{\mathsf{pk}_1}(v_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_n}(v_n)$. Using the AFS spooky multiplication we compute an encrypted *tensor product* of the $\boldsymbol{u}$ and $\boldsymbol{v}$ vectors. Namely, for every i, j we use spooky multiplication to compute

$$\big(\mathsf{Enc}_{\mathsf{pk}_i}(x_{i,j}), \mathsf{Enc}_{\mathsf{pk}_j}(y_{i,j})\big) \leftarrow \mathsf{SpookyMult}\big(\mathsf{Enc}_{\mathsf{pk}_i}(u_i), \mathsf{Enc}_{\mathsf{pk}_j}(u_j)\big),$$

such that $x_{i,j} \oplus y_{i,j} = u_i \cdot v_j$. Then we collapse this tensor product back into an n-vector using the additive homomorphism relative to each key separately.

That is, for every $i \in [n]$ we can compute a ciphertext $\mathsf{Enc}_{\mathsf{pk}_i}(w_i)$ such that $w_i = \bigoplus_{j\in[n]} x_{i,j} \oplus \bigoplus_{j\in[n]} y_{j,i}$. We observe that these ciphertexts form a secret sharing of $u \cdot v$. Indeed, adding up the plaintexts we get:

$$\bigoplus_{i\in[n]} \left(\bigoplus_{j\in[n]} x_{i,j} \oplus \bigoplus_{j\in[n]} z_{j,i} \right) = \bigoplus_{i,j\in[n]} (x_{i,j} \oplus y_{i,j}) = \bigoplus_{i,j\in[n]} u_i \cdot v_j = (\bigoplus_i u_i) \cdot (\bigoplus_j v_j) \tag{1}$$

Thus, if the scheme can support $2d$ interleaved hops of (two-key) spooky multiplication and (single-key) additive homomorphism then it is an AFS-spooky scheme for the class of all depth d arithmetic circuits. We note that the resulting scheme does not depend on the number of inputs or their length, and it only depends on the complexity of C inasmuch as the underlying scheme depends on the depth of the evaluated circuit.

Applications of Spooky Encryption. In Sect. 6 we describe both positive and negative applications of spooky encryption. On the positive, it immediately yields a function secret sharing (FSS) scheme for all functions [4,15]. Previously such a general function secret sharing scheme was only known to follow from sub-exponentially hard indistinguishability obfuscation [4] whereas we can base it on LWE (using our LWE based spooky encryption).

Spooky encryption also gives a simple 2-round multi-party computation protocol. Roughly, AFS-spooky encryption lets each party broadcast an encryption of its input under its own key, then everyone individually performs the AFS-spooky evaluation locally, each party can locally decrypt and recover a share of the output, and the output is recover using another round of communication. There are some technicalities that should be addressed for this idea to work, and perhaps the easiest way of addressing them is to use AFS-spooky encryption to construct *multi-key FHE with threshold decryption (TMFHE)*, which can then be used to get a two-round protocol as done in [22]. Using our obfuscation based construction (which does not require a CRS), this gives the first 2-round semi-honest secure MPC protocol in the plain model.[3]

On the negative side, AFS-spooky encryption yields a counter-example for the transformation of Aiello et al. [1] from multi-prover (MIP) to single-prover protocols. Their idea was to send all of the MIP queries to a single prover, but encrypted under independents keys of a homomorphic encryption scheme. The single prover can homomorphically implement the actions of the MIP provers on the individual encrypted queries, and hopefully the fact that the queries are encrypted under independent keys means that no cross-influence is possible. It is easy to see that spooky encryption violates this hope (by its very nature). Moreover, we show that this transformation can lead to a total break of soundness - in Sect. 6.1 we show how using AFS-spooky encryption can lead to an

[3] In contrast, [12] and [22] construct 2-round protocols in the *CRS* model. As for security against a *malicious* adversary, [20] show that 5 rounds are necessary in the plain model (with respect to black-box proofs of security).

unsound single-prover protocol, when the transformation is applied to a simple two-prover protocol for graph 3-colorability.

Spooky-Free Encryption. Finally, in Sect. 7 we discuss the notion of spooky-free (SF) encryption, which provably ensures that any correlation that an attacker can induce between the original messages $(m_1, \ldots, m_n)$ and "tampered messages" $(m'_1, \ldots, m'_n)$, can be simulated by a "local simulator" that produces m'_i only as a function of m_i (and some shared randomness), see Definition 6. To validate this definition, we show that a spooky-free FHE suffices to prove the security of the natural approach of Aiello *et al.* [1], which was discussed above, of converting a succinct MIP into a succinct one-round argument discussed above. Indeed, spooky-freeness ensures that the attacker cannot cause more damage from seeing all n ciphertexts than what it could have done by seeing each plaintext independently.

We then turn to the systematic study of spooky-free encryption. First, we show that spooky-freeness implies semantic security. On the other hand, a very weak form of non-malleability (called 1-non-malleability here, or 1-bounded CCA security in [8]) implies spooky-freeness. However, since the scheme is non-malleable, it is inherently not homomorphic and so we cannot use it to obtain a delegation scheme via the foregoing approach.

Indeed, to instantiate the approach of Aiello *et al.* constructing succinct arguments for NP, we need a *homomorphic* encryption scheme which is spooky free. As a proof of concept, in the full paper [9] we show how to built such a homomorphic spooky-free encryption using succinct non-interactive arguments of knowledge (SNARKs [3,14]), true-simulation-extractable NIZKs [10] and regular FHE. While the use of SNARKs makes this construction uninteresting in the application to succinct arguments, the clean definition of SF-encryption, coupled with our "proof of concept" implementation, might open the door for more useful future constructions.

1.2 Related Work

The starting point for this line of work is the natural approach, suggested by Aiello *et al.* [1], for constructing a secure delegation scheme by combining a multi-prover interactive proof-system (MIP) with a homomorphic encryption scheme as described above. This intuition was questioned by Dwork *et al.* [11] and our work confirms that the approach of [1] is not always secure.

An approach to overcoming this barrier was taken by Kalai *et al.* [18,19]. They designed a specific MIP (for $\mathcal{P}$) that is sound even against arbitrary no-signaling adversaries. Since semantic-security rules out signaling strategies, they obtain a secure delegation protocol for any language in $\mathcal{P}$.

Spooky Free vs. Homomorphism Extraction. Bitansky and Chiesa defined in [2] a security notion called *homomorphism extraction*, that they show can be used to securely instantiate the construction of Aiello *et al.* and get succinct arguments for NP. Intuitively, this notion says that to produce a valid encryption of m'

from an encryption of m, you must know a function f such that $m' = f(m)$. Compared to our notion of spooky-free (which is also sufficient for the Aiello *et al.* transformation), the main difference is that of "extraction vs. soundness", so homomorphism extraction seems a stronger requirement. For example, homomorphism extraction implies some form of "plaintext awareness" and therefore is non-trivial even for schemes that aren't homomorphic, whereas we show that any non-malleable encryption scheme is spooky-free.

Multi-key FHE. A notion that is related to spooky-encryption, introduced by López-Alt *et al.* [21] is that of multi-key FHE. In a multi-key FHE, similarly to a spooky encryption scheme, the homomorphic evaluation procedure gets as input n ciphertexts encrypted under different keys. The difference is that the output of the evaluation in a multikey FHE is a single ciphertext that can only be decrypted by combining all the n keys. In contrast, in a spooky encryption scheme the result of the spooky evaluation is n ciphertexts, $c_1, \ldots, c_n$ where each c_i is encrypted under the i^{th} original. Thus, spooky encryption can be thought of as a specific type of multi-key FHE.

2 Definitions

2.1 Local, No-Signaling, and Spooky Relations

We say that two distributions D_1, D_2 over a (finite) universe $\mathcal{U}$ are ε-close if their statistical distance $\frac{1}{2}||D_1 - D_2||_1$ is at most ε, and denote it by $D_1 \overset{\varepsilon}{\approx} D_2$. We write $D_1 \equiv D_2$ to denote that the distributions are identical. We say that D_1, D_2 are δ-far if their statistical distance is *at least* δ.

Definition 1. *Let $f : \{0,1\}^{\ell_1} \times \cdots \{0,1\}^{\ell_n} \to \{0,1\}^{\ell'_1} \times \cdots \{0,1\}^{\ell'_n}$ be a* randomized *mapping from n input to n outputs. For input $\boldsymbol{x} = (x_1, \ldots, x_n)$ to f, we denote the i'th component of the output by $f(\boldsymbol{x})_i$, and more generally for a subset $I \subset [n]$ we denote the projected input by $\boldsymbol{x}_I = (x_i : i \in I)$ and the projected output by $f(\boldsymbol{x})_I = (f(\boldsymbol{x})_i : i \in I)$.*

- *f is* local *if there exist n randomized "component mappings" $f_i : \{0,1\}^{\ell_i} \to \{0,1\}^{\ell'_i}$ such that for all $(x_1, \ldots, x_n) \in \{0,1\}^{\ell_1} \times \cdots \{0,1\}^{\ell_n}$, the distribution $f(x_1, \ldots, x_n)$ is a product distribution $f(x_1, \ldots, x_n) \equiv f_1(x_1) \times \cdots \times f_n(x_n)$.*
- *f is* no-signaling *if for every subset $I \in [n]$ and every two inputs $\boldsymbol{x}, \boldsymbol{x}'$ with the same I projection, $\boldsymbol{x}_I = \boldsymbol{x}'_I$, the corresponding projected distributions are equal, $f(\boldsymbol{x})_I \equiv f(\boldsymbol{x}')_I$.*
- *We say that f is ε-*spooky *for some $\varepsilon > 0$ if it is no-signaling, but for every local f' there exists some input $\boldsymbol{x}$ such that $f(\boldsymbol{x})$ and $f'(\boldsymbol{x})$ are at least ε-far.*

These definitions extends to an ensemble of mappings $F = \{f_k : k \in \mathbb{N}\}$, with the mapping parameters n, ℓ_i, ℓ'_i and the distance bound ε possibly depending on the ensemble parameter k. In this case we say that F is spooky *if the f_k's are ε-spooky for a non-negligible $\varepsilon = \varepsilon(k)$.*

As an example, consider the randomized function $f(x_1, x_2) = (y_1, y_2)$ where y_1, y_2 are uniformly random subject to $y_1 \oplus y_2 = x_1 \wedge x_2$. This function is no-signaling since the distributions $f(x)_1$ and $f(x)_2$ are individually uniform, no matter what x is. However, it's easy to show that for any local function $f' = (f'_1, f'_2)$ there is an input $x = (x_1, x_2)$ such that $\Pr[f'_1(x_1) \oplus f'_2(x_2) = x_1 \wedge x_2] \leq 1/2$. Therefore the function f is ε-spooky for $\varepsilon = 1/2$.

2.2 Spooky Encryption

A public-key encryption scheme consists of a tuple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ of polynomial-time algorithms. The key-generation algorithm Gen gets as input a security parameter $\kappa \in \mathbb{N}$ and outputs a pair of public/private keys $(\mathsf{pk}, \mathsf{sk})$. The encryption algorithm Enc gets as input the public-key pk and a bit $m \in \{0,1\}^{\mathsf{poly}(\kappa)}$ and outputs a ciphertext c, whereas the decryption algorithm Dec gets as input the private-key sk and the ciphertext c and outputs the plaintext bit m. The basic correctness guarantee is that $\Pr[\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m)) = m] > 1 - \mathsf{negl}(k)$, where the probability is over the randomness of all these algorithms. The security requirement is that for every pair of polynomial-sized adversaries (A_1, A_2) it holds that

$$\Pr_{\substack{(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\kappa) \\ b \leftarrow \{0,1\}}} \left[\begin{array}{c} (m_0, m_1) \leftarrow A_1(\mathsf{pk}) \text{ s.t. } |m_0| = |m_1| \\ A_2\left(\mathsf{pk}, \mathsf{Enc}_{\mathsf{pk}}(m_b)\right) = b \end{array} \right] \leq \frac{1}{2} + \mathrm{negl}(\kappa).$$

If the message space consists of just a single bit then we say that the scheme is a bit encryption scheme.

Definition 2 (Spooky Encryption). *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a public-key bit-encryption scheme and* $\mathsf{Spooky\text{-}Eval}$ *be a polynomial-time algorithm that takes as input a (possibly randomized) circuit with* $n = n(\kappa)$ *inputs and* n *outputs,* $C : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$*, and also* n *pairs of (public-key, ciphertext), and outputs* n *ciphertexts.*

Let $\mathcal{C}$ *be a class of such circuits, we say that* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Spooky\text{-}Eval})$ *is a* $\mathcal{C}$-spooky encryption scheme *if for every security parameter* κ*, every randomized circuit* $C \in \mathcal{C}$*, and every input* $\boldsymbol{x} = (x_1, \ldots, x_n)$ *for* C*, the distributions*

$$SPOOK[C, x_1, \ldots, x_n] \stackrel{\text{def}}{=} \left\{ (\mathsf{Dec}(\mathsf{sk}_1, c'_1), \ldots, \mathsf{Dec}(\mathsf{sk}_n, c'_n)) : \begin{array}{r} \forall i \in [n] \ (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(1^\kappa), \\ \boldsymbol{c}_i \leftarrow \mathsf{Enc}(\mathsf{pk}_i, x_i), \\ (c'_1, \ldots, c'_n) \leftarrow \mathsf{Spooky\text{-}Eval}(C, (\mathsf{pk}_i, \boldsymbol{c}_i)_i) \end{array} \right\}$$

and $C(x_1, \ldots, x_n)$ *are close upto a negligible distance in* κ*.*

We note that the name *spooky encryption* stems from the application of Definition 2 to circuits C that compute spooky mappings. Indeed, as shown by Dwork *et al.* [11], the semantic security of $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ implies that only (almost) no-signaling C's can be realized, and every homomorphic scheme can realize C's that compute product mappings.

Spooky Encryption with CRS. We say that $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Spooky\text{-}Eval})$ is a $\mathcal{C}$-spooky encryption scheme with CRS if Definition 2 is satisfied except that we allow all algorithms (and the adversary) to get as input also a public uniformly distributed common random string.

2.3 Additive-Function-Sharing Spooky Encryption

An important special case of spooky encryption allow us to take encryptions $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(x_i)$ under n independent keys of inputs $x_1, \ldots, x_n$ to an n-argument function f, and produce new ciphertexts under the same n keys that decrypt to additive secret-shares of $y = f(x_1, \ldots, x_n)$. An encryption scheme that supports such "non-interactive sharing" is called *additive-function-sharing spooky encryption* (or AFS-spooky). Several variants of this concept are defined below:

- We can either insist on getting a *random* secret sharing of y, or contend ourselves with *any* secret sharing. Below we call the latter variant *weak* AFS-spooky, and the former is strong AFS-spooky (or just AFS-spooky).
- Similarly to homomorphic encryption schemes, we can have either a *leveled* variant where key-generation receives an additional depth parameter d and the result supports only circuits of depth upto d, or a *fully* AFS-spooky scheme that supports any circuit with a fixed parameter setting.
- We can either allow non-negligible error probability (i.e., the probability that the computation fails to produce a secret-sharing of the right output y), or insist on a negligible error probability. Below we denote by ε-AFS-spooky the variant where the error probability is bounded by some ε (that need not be negligible), and the variant with negligible error probability is just AFS-spooky.
- Sometimes we want to consider only two-argument functions $f(x_1, x_2)$, a scheme that only supports two-argument functions is called AFS-2-spooky.

The formal definition itself is provided in the full version [9], where we also show that the weak and strong variants are essentially equivalent.

3 LWE-Based Spooky Encryption

3.1 Learning with Errors (LWE) and Multi-key FHE

The LWE assumption roughly says that adding just a little noise to a set of linear equations makes them hard to solve. In our context, we consider equations modulo some integer q and the noise consists of numbers whose magnitude is much smaller than q, as expressed via a noise distribution χ that yields such "small numbers" with high probability. Below we identify $\mathbb{Z}_q$ with the symmetric interval $[-q/2, q/2)$ and let $[x]_q$ denote the reduction of x modulo q into this interval.

Definition 3 (Learning with Errors [28]**).** *Let $n = n(\kappa), q = q(\kappa) \in \mathbb{Z}$ be functions of the security parameter κ and $\chi = \{\chi(\kappa)\}_\kappa$ be a distribution ensemble over $\mathbb{Z}$. The decision-LWE assumption with parameters (n, q, χ) says that for any polynomial $m = m(\kappa) \in \mathbb{Z}$, the following two distribution ensembles are computationally indistinguishable*

$$\mathcal{LWE}[n, m, q, \chi] \stackrel{\text{def}}{=} \big\{(A, \boldsymbol{b}) : A \leftarrow \mathbb{Z}_q^{n\times m},\ \boldsymbol{s} \leftarrow \mathbb{Z}_q^n,\ \boldsymbol{e} \leftarrow \chi^m,\ b := [\boldsymbol{s}A + \boldsymbol{e}]_q\big\},$$

$$\textit{and}\ \ \mathcal{U}[n, m, q] \stackrel{\text{def}}{=} \big\{(A, \boldsymbol{b}) : A \leftarrow \mathbb{Z}_q^{n\times m},\ \boldsymbol{b} \leftarrow \mathbb{Z}_q^m\big\}\ (i.e., uniform\, over\ \mathbb{Z}_q^{(n+1)\times m}).$$

*For $\alpha = \alpha(\kappa) \in (0, 1)$, the α-*DLWE *assumption asserts the existence of parameters n, q, χ as above with n polynomial in κ, such that $e \leftarrow \chi$ yields $|e| < \alpha q$ with overwhelming probability.*

Note that the α-DLWE assumption becomes stronger as α gets smaller, and it is known to be false in the extreme case where $\alpha = 2^{-\Omega(n)}$ using lattice-reduction techniques. On the other hand, we have ample evidence to belive the α-DLWE assumption with $\alpha = 1/\mathsf{poly}(n)$ [5,25,28], and it is commonly belived to hold also for super-polynomially (and perhaps even sub-exponentially) small α's.

We show that assuming hardness of the learning-with-errors problem, there exists a function-secret sharing (in the common-random-string model) for any n-argument function f. Our construction can be built on the multi-key fully homomorphic encryption construction of Mukherjee and Wichs [22] or the one of Peikert and Shiehian [26], which are variations of the Clear-McGoldrick scheme from [7]. We summarize the relevant properties of these constructions:

Theorem 1 [7,22,26]**.** *Assuming the hardness of α-*DLWE *(for some $\alpha(\kappa)$), there exists a multi-key homomorphic encryption with the following properties:*

– *The construction works in the common-random-string model. For parameters $n, m, q = \mathsf{poly}(\kappa)$, all instances have access to a uniformly random matrix $A \in \mathbb{Z}_q^{(n-1)\times m}$.*
– *For any depth parameter d, the scheme supports multi-key evaluation of depth-d circuits using public keys of size $d \cdot \mathsf{poly}(\kappa)$, while secret keys are vectors $\boldsymbol{s} \in \mathbb{Z}_q^n$, regardless of the depth parameter.*
  Specifically, there is an efficient procedure Eval *that is given as input:*
  - *Parameters $d, \ell \in \mathbb{N}$;*
  - *A depth-d circuit computing an ℓ-argument function $f : (\{0, 1\}^*)^\ell \to \{0, 1\}$;*
  - *Public keys $(\mathsf{pk}_1, \dots, \mathsf{pk}_n)$ and* fresh encryptions *(bit-by-bit) of each argument $x_i \in \{0, 1\}^*$ under key pk_i, denoted $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(x_i)$.*

On such input, the Eval *procedure outputs a dimension $n\ell$-vector, $\boldsymbol{c}' = (\boldsymbol{c}'_1 \dots \boldsymbol{c}'_\ell)$ (with each $\boldsymbol{c}'_i \in \mathbb{Z}_q^n$),*[4] *such that for the secret keys $\boldsymbol{s}_i$ corresponding to pk_i it holds that*

$$\sum_{i=1}^{\ell} \langle \boldsymbol{s}_i, \boldsymbol{c}'_i \rangle = \lfloor q/2 \rfloor \cdot f(x_1, \dots, x_n) + e \pmod q$$

for some error $e \in \mathbb{Z}_q$ with $|e| < \alpha q \cdot \mathsf{poly}(\kappa)$.

[4] Referring to [22, Sect. 5.4], the vector $\boldsymbol{c}'_i$ is the result of the product $\hat{C}^{(i)} \times \hat{G}^{-1}(\hat{\boldsymbol{w}}^T)$, without the added noise term e_i^{sm}.

By further making a circular-security assumption, there exists a scheme that supports evaluation of circuits of any depth without growing the public keys.

3.2 LWE-Based AFS Spooky Encryption

Below we show that under the decision-LWE assumption we can construct AFS-spooky encryption schemes (in the common-random-string model). Namely, for every n-argument function $f(x_1, \dots, x_n)$, given encryption of the arguments under n independent public keys, we can compute an encryption of shares under the same keys of an additive secret-sharing of the output $y = f(x_1, \dots, x_n)$.

Theorem 2. *Assuming the hardness of α-DLWE, there exists a leveled ε-AFS-2-Spooky encryption scheme for $\varepsilon = \alpha \cdot \mathsf{poly}(\kappa)$. Further making a circular-security assumption, we get a (non-leveled) ε-AFS-2-spooky encryption scheme.*

Proof. We show that the encryption scheme from Theorem 1 is already essentially a leveled weak AFS-2-spooky encryption scheme. Specifically, Theorem 1 tells us that given the description of a depth-d circuit C, computing a 2-argument function $f : (\{0,1\}^*)^2 \to \{0,1\}$, together with two public-key and corresponding bit-by-bit encryptions, $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(x_i)$, the Eval procedure yields $(\boldsymbol{c}'_1, \boldsymbol{c}'_2) \leftarrow \mathsf{Eval}(C, (\mathsf{pk}_1, c_1), (\mathsf{pk}_2, x_2))$ such that $\langle \mathsf{sk}_1, \boldsymbol{c}'_1 \rangle + \langle \mathsf{sk}_2, \boldsymbol{c}'_2 \rangle = y \cdot q/2 + e \pmod q$, where the sk_i's are the secret keys corresponding to the pk_i's, $y = f(x_1, x_2)$, and $|e| < \alpha q \cdot \mathsf{poly}(\kappa) = \varepsilon q$.

Denote $v_i = [\langle \mathsf{sk}_i, \boldsymbol{c}'_i \rangle]_q$ for $i = 1, 2$ and $v = [v_1 + v_2]_q$. Lemma 1 below says that instead of first adding the v_i's and then rounding to the nearest multiple of $q/2$, we can first round and then add, and this will yield the same result except with error probability of at most 2ε. The only catch is that Lemma 1 assumes that v_1, v_2 are chosen at random subject to their sum modulo q being v, whereas in our case we do not have this guarantee. To account for this, we modify our $\mathsf{Spooky\text{-}Eval}$ procedure, letting it choose a random shift amount $\delta \in \mathbb{Z}_q$ and adding/subtracting it from v_1, v_2, respectively. More detail is provided in the full version [9].

Lemma 1. *Fix some modulus $q \in \mathbb{Z}$, bit $b \in \{0,1\}$, and a value $v \in \mathbb{Z}_q$ such that $v = b \cdot q/2 + e \pmod q$ for some bounded error $|e| < q/4$. Consider choosing v_1, v_2 uniformly at random in $\mathbb{Z}_q$ subject to $v_1 + v_2 = v \pmod q$, and denote $v_i = b_i \cdot q/2 + e_i \pmod q$ with $b_i = [\lceil v_i \cdot 2/q \rfloor]_2 \in \{0,1\}$ and $|e_i| \le q/4$. Then $\Pr_{v_1,v_2}[b_1 \oplus b_2 = b] > 1 - 2(|e|+1)/q$.*

Proof. We break the proof into four cases, namely $b = 0$ vs. $b = 1$ and $e \ge 0$ vs. $e < 0$. Below we prove only the first the case $b = 0$ and $v = e \ge 0$, the other three cases are similar. For the first case consider choosing at random $v_1 \in \mathbb{Z}_q$ and setting $v_2 = [v - v_1]_q = [e - v_1]_q$. It is straightforward (but tedious) to check that the condition $b_1 \oplus b_2 = b = 0$ is satisfied whenever we have

$$\text{either } v_1, v_2 \in \big(\frac{-q}{4} + e, \frac{q}{4}\big), \text{ or } v_1, v_2 \in \big[\frac{-q}{2}, \frac{-q}{4}\big) \cup \big(\frac{q}{4} + e, \frac{q}{2}\big).$$

For example when $v_1 \in \left(\frac{q}{4}+e, \frac{q}{2}\right)$ then we have $b_1 = 1$ and

$$v_2 = e - v_1 \in \left(e - \frac{q}{2},\ e - (\frac{q}{4}+e)\right) \subseteq (-\frac{q}{2},\ -\frac{q}{4}),$$

so we get also $b_2 = 1$ and therefore $b_1 \oplus b_2 = 0 = b$.

The only error regions are $v_1, v_2 \in (\frac{-q}{4}, \frac{-q}{4}+e)$, $v_1, v_2 \in (\frac{q}{4}, \frac{q}{4}+v)$, and (depending on rounding) also upto two of the four points $v_1 \in \{\frac{\pm q}{4},\ \frac{\pm q}{4}+e\} \cap \mathbb{Z}$.

3.3 Beyond AFS-2-Spooky Encryption

The construction from Theorem 2 does not directly extend to functions with more than two arguments, since Lemma 1 no longer holds for more than two v_i's (even for the no-error case of $e = 0$). Instead, we can use the GMW-like transformation that was sketched in the introduction and is described in detail in Sect. 5 to get a general AFS-spooky scheme.

To support this transformation, we need an AFS-2-spooky scheme which is *multi-hop* (in the sense of [13]), i.e. we need to apply the spooky evaluation procedure not just to fresh ciphertexts, but also to evaluated ciphertexts that resulted from previous applications of spooky evaluation. The AFS-2-spooky scheme in Theorem 2 may or may not have this property, depending on the underlying multi-key FHE scheme. In particular the Peikert-Shiehian scheme in [26] is “natively multi-hop,” so we can base our construction on that scheme and get directly a multi-hop AFS-2-spooky scheme which is suitable for our transformation.

On the other hand, the schemes from [7,22] support only one hop, since only fresh cipehrtexts can be processed in a multi-key fashion. We can stil use them for our purposes by applying the same bootstrapping-based transformation as in [13, Theorem 4], which transforms any compact fully-homomorphic scheme to a multi-hop one:[5] More details are provided in the full version [9].

Theorem 3. *Assuming the hardness of α-DLWE, there exists a leveled FHE scheme that supports d interleaved levels of AFS-2-spooky multiplications and single-key addition, with total error probability $\varepsilon = \alpha \cdot d \cdot \mathsf{poly}(\kappa)$.*

Corollary 1. *Assuming the hardness of α-DLWE, there exists a leveled ε-AFS-spooky encryption scheme for $\varepsilon = \alpha \cdot d \cdot \mathsf{poly}(\kappa)$. Further making a circular-security assumption, we get a (non-leveled) ε-AFS-spooky encryption scheme.* $\square$

4 piO Based Spooky Encryption

In this section we show a construction based on probabilistic iO, in conjunction with lossy homomorphic encryption, that can support many 2-key spooky relations, even beyond AFS-spooky. Compared to our LWE-based construction from

[5] The transformation in [13] is described for single-key FHE schemes, but it applies also to multi-key schemes.

Sect. 3, the construction here does not need a CRS and has zero error probability, and it supports more spooky distributions. On the other hand, we are making a much stronger assumption here, and also we need a different scheme for different spooky relations.[6]

The construction in this section supports in particular the functionality that we need for our generic transformation from Sect. 5, that turns an AFS-2-spooky scheme to an AFS-n-spooky one. The resulting AFS-n-spooky also does not use a CRS and has no error probability. Moreover, applying this transformation yields a single scheme supporting all AFS-spooky relations.

Organization of this Section. In Sect. 4.1 we introduce our tools, defining probabilistic indistinguishability obfuscation (using a slightly weaker variant of the definition of Canetti *et al.* [6]) and lossy homomorphic encryption with malicious circuit privacy. In Sect. 4.2 we describe and prove our construction for 2-input spooky encryption scheme, and finally in Sect. 4.3 we show how to obtain a multi-input AFS-spooky encryption.

4.1 Tools

Probabilistic Indistinguishability Obfuscation. Our construction uses probabilistic iO, a notion that was recently introduced by Canetti *et al.* [6]. Loosely speaking, this is an obfuscator for probabilistic circuits with the guarantee that the obfuscations of any two "equivalent" circuits are computationally indistinguishable.

Canetti *et al.* define several variants of piO, where the main distinction is the precise formulation of what it means for circuits to be equivalent. Our definition corresponds to a (weakened variant) of their X-Ind piO (which can be realized assuming sub-exponentially secure iO and sub-exponentially secure OWF, see Theorem 4 below). Roughly, our variant only considers pairs of circuits with the property that for *every* input, their output distributions are *identical*, while the definition in [6] allows a small statistical gap.

To formally define piO, we consider a (possibly randomized) PPT sampling algorithm S that given as input a security parameter 1^κ, outputs a triple (C_0, C_1, z), where C_0 and C_1 are randomized circuits (to be obfuscated) and z is some auxiliary input. We say that a sampler S is an *equivalent-circuit-sampler* if with probability 1 it outputs circuits C_0 and C_1 such that for every x the circuits $C_0(x)$ and $C_1(x)$ generate identical distributions.

Definition 4 (Probabilistic Indistinguishable Obfuscation (piO), [6]). *A* probabilistic indistinguishability obfuscator *is a probabilistic polynomial-time algorithm* piO *that, given as input a security parameter* 1^κ *and a* probabilistic *circuit* C*, outputs a circuit* $C' = \mathsf{piO}(1^\kappa, C)$ *(which may be deterministic) of size at most* $|C'| = \mathsf{poly}(\kappa, |C|)$ *such that the following two properties hold:*

[6] We can extend the construction so that a single scheme can handle an entire class of spooky relations, as long as we can describe relations in that class and verify that a given relation is no-signaling.

1. *For every individual input x, the distribution $C(x)$ and $\big(\mathsf{piO}(1^\kappa, C)\big)(x)$ are identical.*[7]
2. *For every* equivalent-circuit-sampler *S, drawing $(C_0, C_1, z) \leftarrow S(1^\kappa)$ we get computationally indistinguishable distributions:*

$$\{(C_0, C_1, z, \mathsf{piO}(1^\kappa, C_0))\} \overset{c}{=} \{(C_0, C_1, z, \mathsf{piO}(1^\kappa, C_1))\}$$

We note that our correctness guarantee is incomparable to that given by [6]. Indeed, motivated by their PRF based construction, the definition in [6] basically requires that no PPT adversary can distinguish between oracle access to C and to $\mathsf{piO}(1^\kappa, C)$ (so long as the adversary is not allowed to repeat its queries). On the one hand our definition is weaker in that it only considers each input individually, but on the other hand it is stronger in that it requires that for each such individual input the distributions are *identical.* Our correctness guarantee can be easily obtained from the construction in [6], by using an underlying PRF $\{f_s\}_s$ with the property that $f_s(x)$ is individually uniformly random for every x. The latter can be easily obtained by taking any PRF and xor-ing its output with a fixed random string.

Theorem 4 [6]**.** *Assuming the existence of a sub-exponentially indistinguishable indistinguishability obfuscator for circuits and a sub-exponentially secure puncturable PRF, there exists a* probabilistic indistinguishability obfuscator*.*

Lossy Encryption. Loosely speaking, a lossy encryption scheme has a procedure $\widetilde{\mathsf{Gen}}$ for generating "lossy public keys." These keys are indistinguishable from normal public keys, but have the property that ciphertexts generated using such lossy keys contain no information about their plaintext. We defer the formal definition to the full version [9].

Malicious Circuit-Private Encryption. A public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, with message space $\{0,1\}^\ell$, is a homomorphic encryption scheme for a class of Boolean circuits $\mathcal{C}$ on ℓ-bit inputs if there exists a PPT algorithm Eval, such that for every key-pair $(\mathsf{pk}, \mathsf{sk})$, circuit $C \in \mathcal{C}$ and ciphertext $c = \mathsf{Enc}_{\mathsf{pk}}(x)$, where $x \in \{0,1\}^\ell$, on input (C, c) the algorithm $\mathsf{Eval}_{\mathsf{pk}}$ outputs c^* such that $\mathsf{Dec}_{\mathsf{sk}}(c^*) = C(x)$. If the length of c^* does not depend on C then we say that the scheme is compact.

As noted in the introduction, our construction requires a homomorphic encryption scheme that has *malicious circuit privacy*, which means that the ciphertext c^* does not reveal any non-trivial information about the circuit C which was used to generate it, even for an adversarially chosen public-key pk and ciphertext c. We defer the formal definition to the full version [9].

Malicious circuit privacy for evaluating NC_1 circuits can be achieved by a "folklore" combination of an information theoretic variant of Yao's garbled circuit [17] with an oblivious transfer protocol that has perfect security against a

[7] The latter distribution is defined also over the randomnees of piO. Note that this does not imply that the joint distribution for multiple inputs will be the same in the two cases.

malicious receiver. The latter can be constructed based on DDH [23]. Moreover, these schemes can be made lossy using standard techniques.

Moreover, we can apply the techniques of Ostrovsky *et al.* [24] to bootstrap this result to any poly-circuit, assuming the existence of (leveled) fully homomorphic encryption with NC_1 decryption. The latter scheme can be instantiated based on LWE, see more details in the full version [9]. Hence we obtain:

Theorem 5. *Assuming the hardness of* LWE *and* DDH*, there exists a lossy leveled fully-homomorphic encryption scheme with malicious circuit privacy.*

4.2 Two-Key Spooky Encryption from piO

Our construction relies on a property of two-input relations that we call *re-sampleability.* Roughly, it should be possible to sample efficiently from the distribution of the second coordinate conditioned on a particular fixed value for the first coordinate.

Definition 5 (Efficiently Re-sampleable). *A randomized polynomial-size circuit* $C : \{0,1\}^{\ell_1} \times \{0,1\}^{\ell_2} \to \{0,1\}^{\ell'_1} \times \{0,1\}^{\ell'_2}$ *is* efficiently re-sampleable *if there exists a polynomial-size randomized "resampling circuit"* RS_C*, such that for any input* (x_1, x_2) *to* C*, the distribution* $C(x_1, x_2)$ *is identical to the "resampled distribution"* $\{(y_1, y'_2) : (y_1, y_2) \leftarrow C(x_1, x_2),\ y'_2 \leftarrow RS_C(x_1, x_2, y_1)\}$.

We construct a 2-key spooky scheme that supports any 2 input/output circuit that is both *efficiently re-sampleable* and *no-signaling.*

Theorem 6 (2-Key Spooky Encryption from piO). *Let* $C : \{0,1\}^{\ell_1} \times \{0,1\}^{\ell_2} \to \{0,1\}^{\ell'_1} \times \{0,1\}^{\ell'_2}$ *be an efficiently re-sampleable no-signaling circuit, with re-sampling circuit* RS_C*. If there exist (1)* piO*, and (2) a perfectly-lossy homomorphic encryption scheme that can evaluate* C *and* RS_C*, and is perfectly malicious circuit private, then there exists a* C*-spooky encryption scheme, which is also perfectly lossy (and hence semantically secure).*

We stress that the encryption scheme that we need for Theorem 6 must be able to evaluate C and RS_C and be perfectly malicious circuit private, but *it need not be compact.* In the full paper we describe such a scheme for NC_1 circuits based on DDH. Hence, under DDH and piO, we get a C-spooky scheme for every re-sampleable and no-signaling C in NC_1. Moreover, we can use the techniques of Ostrovsky *et al.* [24] to supports any poly-size circuit, assuming both DDH and FHE. Since [6] show that full-fledged FHE can be built based on piO, we get a construction under DDH and piO for C-spooky scheme for every re-sampleable and no-signaling polynomial-size circuit C.

Remark 1 (Almost No-Signaling). A natural relaxation of no-signaling circuits, considered in previous works (e.g., [11,18,19]), allows the distributions $C(x, y)_1$ and $C(x, y')_1$ to be indistinguishable (rather than identical). Such circuit is called *almost no-signaling.*

It is clear that for a secure C-spooky encryption scheme to exist, C must be (at least) almost no-signaling (cf. [11]). However our construction does not extend to the "almost" case, Theorem 6 requires that C to be perfectly no-signaling, i.e. $C(x,y)_1$ and $C(x,y')_1$ must be identically distributed for all x, y, y'. Supporting almost no-signaling circuits is left to future work.

Proof of Theorem 6. Let piO be a probabilistic indistinguishability obfuscator and let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be the encryption scheme from the theorem statement, with $\widetilde{\mathsf{Gen}}$ the corresponding lossy key generation algorithm and Eval the homomorphic evaluation algorithm with malicious circuit privacy.

Each instance of our construction uses two public/secret keys pairs, where only the first pair is used for "normal encryption and decryption," and the other pair is only used for spooky evaluation. In addition to the two pairs, the public key also contains an obfuscated program that implements spooky evaluation using the secret key. That obfuscated program has a secret key hard-wired, and given two ciphertexts c_1, c_2 it decrypt the first one, then evaluates the re-sampling circuit RS_C homomorphically on the other. A complete description of the resulting scheme is found in Fig. 1.

We first show that the scheme supports spooky evaluation of C and then show that it is a lossy encryption scheme (and in particular is semantically secure).

Lemma 2. *The scheme* $(\mathsf{Gen\text{-}Spooky}, \mathsf{Enc\text{-}Spooky}, \mathsf{Dec\text{-}Spooky}, \mathsf{Spooky\text{-}Eval})$ *is* C*-spooky.*

Proof. The spooky evaluation procedure gets as input two public-keys $\mathsf{pk\text{-}spooky}_1 = \big(\mathsf{pk1}_1, \mathsf{pk2}_1, \tilde{P}_1\big)$, $\mathsf{pk\text{-}spooky}_2 = \big(\mathsf{pk1}_2, \mathsf{pk2}_2, \tilde{P}_2\big)$, and matching ciphertexts $c_1 = \mathsf{Enc\text{-}Spooky}(\mathsf{pk\text{-}spooky}_1, x_1)$ and $c_2 = \mathsf{Enc\text{-}Spooky}(\mathsf{pk\text{-}spooky}_2, x_2)$ (for some inputs x_1, x_2 to C). It simply runs the obfuscated program $\tilde{P}_1 = \mathsf{piO}(1^\kappa, P[\mathsf{sk1}_1, \mathsf{pk2}_1])$ on input $(c_1, \mathsf{pk1}_2, c_2)$ and returns its output.

By construction and using the correctness of piO, this procedure outputs c_1' and c_2' such that $c_1' \leftarrow \mathsf{Enc}(\mathsf{pk2}_1, y_1)$, where $y_1 \leftarrow \big(C(x_1, 0^{\ell_2})\big)_1$, and $c_2' \leftarrow \mathsf{Eval}_{\mathsf{pk1}_2}(\mathsf{RS}[x_1, y_1, r], c_2)$, where $\mathsf{RS}[x_1, y_1, r](x_2) \equiv RS_C(x_1, x_2, y_1; r)$. By the no-signaling property y_1 is distributed identically to $y_1' \leftarrow \big(C(x_1, x_2)\big)_1$ and so c_2' is distributed as $\mathsf{Eval}_{\mathsf{pk1}_2}(\mathsf{RS}[x_1, y_1', r], c_2)$. Hence

$$\begin{aligned} \mathsf{Dec\text{-}Spooky}(\mathsf{sk\text{-}spooky}_1, c_1') = \mathsf{Dec}_{\mathsf{sk1}_1}\left(\mathsf{Enc}(\mathsf{pk2}_1, y_1')\right) &= y_1' \\ \text{and } \mathsf{Dec\text{-}Spooky}(\mathsf{sk\text{-}spooky}_2, c_2') = \mathsf{RS}[x_1, y_1', r](x_2) &= RS_C\big(x_1, x_2, y_1'; r\big)_2. \end{aligned}$$

By the definition of re-sampling, the joint distribution $\Big(\mathsf{Dec\text{-}Spooky}(\mathsf{sk\text{-}spooky}_1, c_1'), \mathsf{Dec\text{-}Spooky}(\mathsf{sk\text{-}spooky}_2, c_2')\Big)$ is identical to $C(x_1, x_2)$, as required.

Lemma 3. *The scheme* $(\mathsf{Gen\text{-}Spooky}, \mathsf{Enc\text{-}Spooky}, \mathsf{Dec\text{-}Spooky})$ *is a perfectly lossy encryption scheme.*

The probabilistic circuit $P[\mathsf{sk1},\mathsf{pk2}](c_1,\mathsf{pk},c)$:

Hardwired: a private-key $\mathsf{sk1}$ and a public-key $\mathsf{pk2}$.
Input: a ciphertext c_1 (presumably under $\mathsf{pk1}$),
and additional (presumably matching) public-key pk and ciphertext c.

```
1. Decrypt $x_1 \leftarrow \mathsf{Dec}_{\mathsf{sk1}}(c_1)$;[a]
2. Choose randomness $r, r' \leftarrow \{0,1\}^*$ for $C$ and $RS_C$, respectively;
3. Set $y_1 \leftarrow C(x_1, 0^{\ell_2}; r)_1$ and encrypt $c_1' \leftarrow \mathsf{Enc}_{\mathsf{pk2}}(y_1)$;
4. Define the circuit $\mathsf{RS}[x_1, r, r'](x_2) \equiv RS_C(x_1, x_2, \overbrace{C(x_1, 0^{\ell_2}; r)_1}^{=y_1}; r')$;
5. Compute homomorphically $c_2' \leftarrow \mathsf{Eval}_{\mathsf{pk}}(\mathsf{RS}[x_1, r, r'], c)$.
6. Output $\big((2, c_1'), (1, c_2')\big)$.[b]
```

piO based Spooky Encryption

- $\mathsf{Gen\text{-}Spooky}(1^\kappa)$:
  1. Select $(\mathsf{pk1},\mathsf{sk1}), (\mathsf{pk2},\mathsf{sk2}) \leftarrow \mathsf{Gen}(1^\kappa)$, and set $\tilde{P} \leftarrow \mathsf{piO}(1^\kappa, P[\mathsf{sk1},\mathsf{pk2}])$.
  2. Output the secret key $\mathsf{sk\text{-}spooky} = (\mathsf{sk1},\mathsf{sk2})$ and public key $\mathsf{pk\text{-}spooky} = \Big(\mathsf{pk1},\mathsf{pk2},\tilde{P}\Big)$.
- $\mathsf{Enc\text{-}Spooky}\Big((\mathsf{pk1},\mathsf{pk2},\tilde{P}), x\Big)$: Output $\big(1, \mathsf{Enc}_{\mathsf{pk1}}(x)\big)$.
- $\mathsf{Dec\text{-}Spooky}\big((\mathsf{sk1},\mathsf{sk2}), (tag, c)\big)$: If $tag = 1$ output $\mathsf{Dec}_{\mathsf{sk1}}(c)$, else output $\mathsf{Dec}_{\mathsf{sk2}}(c)$.
- $\mathsf{Spooky\text{-}Eval}\big((\mathsf{pk1}_1,\mathsf{pk2}_1,\tilde{P}_1), c_1, \ (\mathsf{pk1}_2,\mathsf{pk2}_2,\tilde{P}_2), c_2,\big)$: Output $\tilde{P}_1(c_1, \mathsf{pk1}_2, c_2)$.

[a] We assume that Dec always returns some value, even if c_1 is not a valid ciphertext.
[b] The tags "2", "1" signal to the decryption algorithm which secret key to use.

Fig. 1. piO based spooky encryption

The probabilistic circuit $P'[\mathsf{sk1},\mathsf{pk2}](c_1,\mathsf{pk},c)$:

The same as $P[\mathsf{sk1},\mathsf{pk2}](c_1,\mathsf{pk},c)$, but setting $c_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}_2}(0^{\ell_1})$ in Step 3 rather than $c_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}_2}(y_1)$.

The probabilistic circuit $P''[\mathsf{pk2}]$:

Hardwired: a public-key $\mathsf{pk2}$.
Input: a ciphertext c_1, a public-key pk and a ciphertext c (presumably under pk).

```
1. Encrypt $c_1' \leftarrow \mathsf{Enc}_{\mathsf{pk2}}(0^{\ell_1})$.
2. Choose randomness $r \leftarrow \{0,1\}^*$ for $C$, and define $f[r](\cdot) \equiv C(0^{\ell_1}, \cdot\, ; r)_2$.
3. Compute homomorphically $c_2' \leftarrow \mathsf{Eval}_{\mathsf{pk}}(f[r], c)$.
4. Output $\big((2, c_1), (1, c_2')\big)$.
```

Fig. 2. The probabilistic circuits $P'[\mathsf{sk1},\mathsf{pk2}]$ and $P''[\mathsf{pk2}]$

Proof. We need to show that there is an alternative key-generation procedure $\widetilde{\mathsf{Gen\text{-}Spooky}}$, producing public keys that are indistinguishable from the real ones, but such that ciphertexts encrypted relative to these keys contain no information about the encrypted plaintext.

The main challenge in establishing the lossiness of the scheme is in showing that the public-keys are indistinguishable from lossy keys despite the obfuscated programs in the public-key (which depend on the corresponding secret keys). Toward that end, we will (gradually) show that these obfuscated programs are computationally indistinguishable from programs that do not depend on the secret keys.

Below we state and prove a few claims, where we consider the distributions $(\mathsf{pk1}, \mathsf{sk1}), (\mathsf{pk2}, \mathsf{sk2}) \leftarrow \mathsf{Gen}(1^\kappa)$ and $\widetilde{\mathsf{pk1}}, \widetilde{\mathsf{pk2}} \leftarrow \widetilde{\mathsf{Gen}}(1^\kappa)$, where $\widetilde{\mathsf{Gen}}$ is the lossy key-generation of the underlying encryption scheme.

Claim 4.1. $\Big(\mathsf{pk1}, \mathsf{pk2}, \mathsf{piO}(1^\kappa, P[\mathsf{sk1}, \mathsf{pk2}])\Big) \overset{c}{=} \Big(\mathsf{pk1}, \widetilde{\mathsf{pk2}}, \mathsf{piO}(1^\kappa, P[\mathsf{sk1}, \widetilde{\mathsf{pk2}}])\Big)$.

Proof. Follows from the indistinguishability between standard and lossy public-keys of the underlying scheme.

Claim 4.2. $\Big(\mathsf{pk1}, \widetilde{\mathsf{pk2}}, \mathsf{piO}(1^\kappa, P[\mathsf{sk1}, \widetilde{\mathsf{pk2}}])\Big) \overset{c}{=} \Big(\mathsf{pk1}, \widetilde{\mathsf{pk2}}, \mathsf{piO}(1^\kappa, P'[\mathsf{sk1}, \widetilde{\mathsf{pk2}}])\Big)$, where $P'[\mathsf{sk1}, \widetilde{\mathsf{pk2}}]$ is similar to $P[\mathsf{sk1}, \widetilde{\mathsf{pk2}}]$ except that it encrypts 0^{ℓ_1} rather than y_1 in Step 3, see Fig. 2.

Proof. Since $\widetilde{\mathsf{pk2}}$ is a *lossy* public-key, $\mathsf{Enc}_{\widetilde{\mathsf{pk2}}}(0^{\ell_1})$ and $\mathsf{Enc}_{\widetilde{\mathsf{pk2}}}(y_1)$ are *identically* distributed. Hence P and P' have identical output distribution for every input, and so their piO-obfuscations are indistinguishable.

We proceed to the main claim:

Claim 4.3. $\Big(\mathsf{pk1}, \widetilde{\mathsf{pk2}}, \mathsf{piO}(1^\kappa, P'[\mathsf{sk1}, \widetilde{\mathsf{pk2}}])\Big) \overset{c}{=} \Big(\mathsf{pk1}, \widetilde{\mathsf{pk2}}, \mathsf{piO}(1^\kappa, P''[\widetilde{\mathsf{pk2}}])\Big)$, where the program $P''[\widetilde{\mathsf{pk2}}]$, defined in Fig. 2, does not have the secret key $\mathsf{sk1}$ (hence it cannot recover x_1 or compute y_1), so on $c = \mathsf{Enc}_{\mathsf{pk}}(x_2)$ it evaluates homomorphically $f''(x_2) = C(0^{\ell_1}, x_2)_2$ rather than $f'(x_2) = RS_C(x_1, x_2, y_1)$.

Proof. We will show that for every valid secret key $\mathsf{sk1}$ and arbitrary public key $\widetilde{\mathsf{pk2}}$, the randomized programs $P'[\mathsf{sk1}, \widetilde{\mathsf{pk2}}]$ and $P''[\widetilde{\mathsf{pk2}}]$ are functionally identical, in the sense that their outputs are identically distributed for every input. The claim will then follow from the fact that piO is a probabilistic indistinguishability obfuscator (see Definition 4).

Note that the first output $c_1' = \mathsf{Enc}_{\widetilde{\mathsf{pk2}}}(0^{\ell_1'})$ is generated identically by the two programs, and is independent of everything else that happens in these programs, so we only need to show that the second output c_2' is identically distributed. To show this, we first establish that c_2' is an encryption under pk of a value y_2 that is distributed identically in the two programs, and then we appeal to the malicious circuit-privacy of the underlying scheme to conclude that also c_2' itself is identically distributed.

For starters, fix some arbitrary $x_1 \in \{0,1\}^{\ell_1}$ and $x \in \{0,1\}^{\ell_2}$, and consider the following distributions

$$\begin{aligned}
\mathcal{D}_1[x_1,x] &= \{y_1 \leftarrow C(x_1,0^{\ell_2})_1,\ \text{output } y_2 \leftarrow RS_C(x_1,x,y_1)\}, && \text{// Output distribution of } P' \\
\mathcal{D}_2[x_1,x] &= \{y_1 \leftarrow C(x_1,x)_1,\ \ \text{output } y_2 \leftarrow RS_C(x_1,x,y_1)\}, \\
\mathcal{D}_3[x_1,x] &= \{\text{output } y_2 \leftarrow C(x_1,x)_2\}, \\
\mathcal{D}_4[x] &= \{\text{output } y_2 \leftarrow C(0^{\ell_1},x)_2\}. && \text{// Output distribution of } P''
\end{aligned}$$

Since C is a no-signaling circuit then $\mathcal{D}_1[x_1,x] = \mathcal{D}_2[x_1,x]$ and $\mathcal{D}_3[x_1,x] = \mathcal{D}_4[x]$, and since R_C is the re-sampling circuit for C then we also have $\mathcal{D}_2[x_1,x] = \mathcal{D}_3[x_1,x]$. We therefore conclude that the two distributions $\mathcal{D}_1[x_1,x]$ and $\mathcal{D}_4[x]$ are identical for every x_1,x.

Now consider $x_1 = \mathsf{Dec}_{\mathsf{sk1}}(c_1)$, and x which is the "effective plaintext" for pk, c (such x must exist since the underlying scheme is malicious circuit-private). Recall that the second output of $P'[\mathsf{sk1},\widetilde{\mathsf{pk}2}]$ consists of a homomorphic evaluation of $\mathcal{D}_1[x_1,x]$, while the second output of $P''[\widetilde{\mathsf{pk}2}]$ consists of homomorphic evaluation of $\mathcal{D}_4[x]$. Using perfect malicious circuit privacy, we conclude that these outputs are identically distributed.

Having established that the output distributions of $P'[\mathsf{sk1},\widetilde{\mathsf{pk}2}]$ and $P''[\widetilde{\mathsf{pk}2}]$ are identical (for every input), the claim follows because piO is a probabilistic indistinguishability obfuscator.

Claim 4.4. $\left(\mathsf{pk1},\widetilde{\mathsf{pk}2},\mathsf{piO}(P''_{\widetilde{\mathsf{pk}2}})\right) \overset{c}{=} \left(\widetilde{\mathsf{pk}1},\widetilde{\mathsf{pk}2},\mathsf{piO}(P''_{\widetilde{\mathsf{pk}2}})\right)$.

Proof. This claim follows from the indistinguishability between standard and lossy public-keys of the underlying scheme.

Combining Claims 4.1-4.3, the two distributions $\left(\mathsf{pk1},\mathsf{pk2},\mathsf{piO}(P_{\mathsf{sk1},\mathsf{pk2}})\right)$ and $\left(\widetilde{\mathsf{pk}1},\widetilde{\mathsf{pk}2},\mathsf{piO}(P''_{\widetilde{\mathsf{pk}2}})\right)$ are computationally indistinguishable. We complete the proof of Lemma 3 by observing that keys drawn from the latter distribution are lossy, since the key $\widetilde{\mathsf{pk}1}$ is lossy, the $\mathsf{Enc\text{-}Spooky}$ procedure just uses the underlying encryption procedure with $\widetilde{\mathsf{pk}1}$, and the program $P''[\mathsf{pk2}]$ that we obfuscate is independent of $\widetilde{\mathsf{pk}1}$.

4.3 piO Based Multi-key Spooky Encryption

To obtain spooky encryption for more than two inputs, we would like to invoke our general transformation from 2-key spooky encryption to n-key spooky encryption (see Theorem 8). The scheme in Theorem 6 supports spooky multiplication, but we need it to support *multiple alternating hops* of (single-key) additive homomorphism and spooky multiplication. This is obtained by the following lemma:

Lemma 4. *Assume the existence of (1)* piO *and (2) a lossy encryption scheme that is homomorphic for all one-bit to one-bit functions with perfect malicious* *circuit privacy. Then, for every $d = d(\kappa)$, there exists an encryption scheme that supports d interleaved levels of AFS-2-spooky multiplications and single-key additions.*

Proof (Proof Sketch). To obtain an additive homomorphism, we use a construction of Canetti *et al.* [6] which, assuming piO, transforms any lossy encryption into a d-leveled FHE. This is done by taking d copies of keys of the original lossy scheme and publishing $d-1$ obfuscated programs where the i^{th} obfuscated program takes as input two ciphertexts encrypted under the i^{th} key, decrypts them (using the i^{th} private-key which is hard-wired) applies one operation (AND, XOR, NAND, etc.) and encrypts the result under the $(i+1)^{\text{th}}$ key. Using the fact that the scheme is lossy, Canetti *et al.* show that the piO obfuscation hides the hard-wired private keys and semantic security is maintained.

For our application, we need to compute multiple spooky multiplications, and then sum them up with single-key addition. To get n-input AFS-spooky we need to sum up n ciphertexts, which can be done using an addition tree of depth $d = \log n$.

Looking more closely at the construction from [6], we observe that by setting $d = i \log n$ we can already support i interleaving hops of (single-key) additive homomorphism and 2-input spooky multiplications. This follows since the transformation in [6] has the property that after every additive homomorphic operation, we obtain a fresh ciphertext (under a new-key).

Using the scheme from Lemma 4 and applying Theorem 8, we get:

Theorem 7 (n-Key Spooky from piO). *Assume existence of (1) piO and (2) a lossy encryption scheme that is homomorphic for all single-bit to single-bit functions with perfect malicious circuit privacy. Then there exists a leveled AFS-spooky encryption scheme.*

5 From 2-Input to n-Input AFS-Spooky

Theorem 8 (2-Spooky to n-Spooky). *Let $d = d(\kappa)$ and assume that there exists a public-key bit-encryption scheme that supports $2d$ (interleaving) hops of (1) single-key compact additive homomorphism and (2) two-key spooky multiplication. Then, that same scheme is a d-level AFS-spooky encryption.*

Proof. Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be the encryption scheme in the theorem statement, let $\mathsf{Spooky\text{-}Mult}$ be the spooky multiplication PPT algorithm and let Eval be the single-key homomorphic evaluation algorithm (that supports compact additive homomorphism). We show a procedure that given as input:

1. A depth-d, fan-in-2, n-input arithmetic circuit over $\mathsf{GF}(2)$, $C : (\{0,1\}^*)^n \to \{0,1\}$;
2. n public-keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$; and
3. n ciphertexts $c_1, \ldots, c_n$, where $c_j = \mathsf{Enc}(\mathsf{pk}_j, x_j)$,

outputs a sequence of ciphertexts $c'_1, \dots, c'_n$ such that $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c'_j) = C(x_1, \dots, x_n)$ (where addition is over $\mathsf{GF}(2)$).

The procedure processes the circuit wire by wire. We maintain the invariant that whenever a wire w is processed, the procedure generates ciphertexts $c_1^{(w)}, \dots, c_n^{(w)}$ such that $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(w)})$ is the correct value of the wire w when the circuit C is evaluated on input $(x_1, \dots, x_n)$. Furthermore, if the wire w is at distance i from the input then $c_1^{(w)}, \dots, c_n^{(w)}$ have passed at most $2i$ hops of homomorphic operations. In particular, at the end of the process the procedure will have generated the sequence of ciphertexts $c_1^{\text{out}}, \dots, c_n^{\text{out}}$ such that $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{\text{out}})$ is equal to the output value of the circuit, as required. We proceed to describe how the wires are (inductively) processed.

Consider an *input* wire w, corresponding to an input bit b which is part of the i^{th} input x_i, and for which we are given the input ciphertext $c = \mathsf{Enc}_{\mathsf{pk}_i}(b)$. For that wire we set $c_i^{(w)} = c$ and $c_j^{(w)} = \mathsf{Enc}_{\mathsf{pk}_{j'}}(0)$ for all $j \neq i$. Hence, $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(w)}) = \mathsf{Dec}_{\mathsf{sk}_i}(c) = b$, which is the correct value for the wire w.

Consider a gate g with input wires u, v and output wire w. Let b_u (resp., b_v) be the value on the wire u (resp., v) when C is evaluated on input $(x_1, \dots, x_n)$. By induction, we have already generated ciphertexts $c_1^{(u)}, \dots, c_n^{(u)}$ and $c_1^{(v)}, \dots, c_n^{(v)}$ such that $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(u)}) = b_u$ and $\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(v)}) = b_v$.

For the case that g is an addition gate, we set $c_j^{(w)} = \mathsf{Eval}\left(\mathsf{pk}_j, \oplus, c_j^{(u)}, c_j^{(v)}\right)$ and we get:

$$\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(w)}) = \sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(\mathsf{Eval}_{\mathsf{pk}_j}(\oplus, c_j^{(u)}, c_j^{(v)})) = \sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(u)}) \oplus \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(v)}) = b_u \oplus b_v,$$

which is the correct value for the wire w. Furthermore, each new ciphertext was obtained by just a single homomorphic operation.

Now consider the case that g is a multiplication gate. We first compute auxiliary ciphertexts $(f_{j,j'}, g_{j,j'}) = \mathsf{Spooky\text{-}Mult}(\mathsf{pk}_j, \mathsf{pk}_{j'}, c_j^{(u)}, c_{j'}^{(v)})$, for every $j, j' \in [n]$. We then set

$$c_j^{(w)} = \mathsf{Eval}_{\mathsf{pk}_j}(\oplus, f_{j,1}, \dots, f_{j,n}, g_{1,j}, \dots, g_{n,j}).$$

We obtain that:

$$\begin{aligned}
\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}\left(c_j^{(w)}\right) &= \sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}\left(\mathsf{Eval}_{\mathsf{pk}_j}(\oplus, x_{j,1}, \dots, x_{j,n}, y_{1,j}, \dots, y_{n,j})\right) \\
&= \sum_{j \in [n]} \sum_{j' \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(f_{j,j'}) \oplus \mathsf{Dec}_{\mathsf{sk}_j}(g_{j',j}) \\
&= \sum_{j \in [n]} \sum_{j' \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(u)}) \cdot \mathsf{Dec}_{\mathsf{sk}_j}(c_{j'}^{(v)}) \\
&= \Big(\sum_{j \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_j^{(u)}) \Big) \cdot \Big(\sum_{j' \in [n]} \mathsf{Dec}_{\mathsf{sk}_j}(c_{j'}^{(v)}) \Big) \quad = \quad b_u \cdot b_v,
\end{aligned}$$

which is the correct value for the wire w (where the fourth equality is due to the Spooky-Mult guarantee). Furthermore, each new ciphertext was obtained by applying two hops of homomorphic operations.

6 Applications of Spooky Encryption

6.1 Counter Example for the [1] Heuristic

Building on [11], we show that AFS-2-spooky encryption gives a counter-example to a natural method proposed by Aiello et al. [1] for building succinct arguments for NP, resolving a question posed by [11]. The suggestion of Aiello *et al.* [1] was to take any multi-prover interactive proof-system (MIP) and to use that proof-system using only a *single* prover by sending all of the MIP queries encrypted under independents keys of a homomorphic encryption scheme.[8] The fact that the scheme is homomorphic allows the honest prover to answer the different queries (homomorphically) and the intuition was that the use of different keys means that only local homomorphisms are possible. Dwork *et al.* [11] questioned this intuition and raised the question of whether there exist spooky encryption schemes that allow for other kinds of attacks which can break the soundness of the [1] protocol. We show that this is indeed the case: there exists an MIP (suggested by [11]) which, when combined with any AFS-2-spooky encryption scheme via the [1] transformation, yields an insecure protocol. The MIP that we use is based on a PCP for 3-coloring due to Petrank [27]:

Theorem 9 [27]. *There exists a universal constant $\varepsilon > 0$ such that distinguishing between the following two types of graphs is NP complete:*

- *G is 3-colorable.*
- *Every 3-coloring of G has at least ε fraction of monochromatic edges.*

This PCP leads to the following natural MIP protocol between a verifier V and two non-communicating provers P_1 and P_2 (who, in case G is 3-colorable, also have access to the same 3-coloring of G).

1. V chooses a random edge $(u, v) \in E$, then with probability $1/3$ it sets $q_1 = u$ and $q_2 = v$, with probability $1/3$ it sets $q_1 = u$ and $q_2 = u$, and with probability $1/3$ it sets $q_1 = v$ and $q_2 = v$. V sends q_1 to P_1 and q_2 to P_2.
2. Each P_i sends the color $a_i \in \{0, 1, 2\}$ of the vertex q_i (encoded as two bits).
3. V accepts if $q_1 = q_2$ and $a_1 = a_2$, or if $q_1 \neq q_2$ and $a_1 \neq a_2$.

Completeness and soundness are easy to see, some details are given in the full version [9].

Insecurity of the 3-Coloring MIP. Composed the foregoing MIP with any AFS-2-spooky encryption scheme yields an insecure protocol. More specifically, the

[8] Actually, the original suggestion in [1] was to use a PCP (rather than an MIP). Dwork *et al.* [11] show that using PCPs is not sound and raise the question of whether soundness can be obtained by replacing the PCP with an MIP.

cheating prover is given ciphertexts $c_1 = \mathsf{Enc}_{\mathsf{pk}_1}(q_1)$ and $c_2 = \mathsf{Enc}_{\mathsf{pk}_2}(q_2)$. Loosely speaking, using the spooky evaluation algorithm it can produce ciphertexts $\mathsf{Enc}_{\mathsf{pk}_1}(a_1)$ and $\mathsf{Enc}_{\mathsf{pk}_2}(a_2)$ for bits $a_1, a_2 \in \{0,1\}$ such that $a_1 = a_2$ if and only if $u = v$. It sends as its answers to V the ciphertext $\big(\mathsf{Enc}_{\mathsf{pk}_1}(0), \mathsf{Enc}_{\mathsf{pk}_1}(a_1)\big)$ as its answer to the first query and $\big(\mathsf{Enc}_{\mathsf{pk}_1}(0), \mathsf{Enc}_{\mathsf{pk}_1}(a_2)\big)$ as its answer to the second query (the extra encryption of 0 is used simply because the verifier expects an answer with 2 bits).

Now, if the verifier choose $q_1 = u$ and $q_2 = v$ (corresponding to the first of the three possibilities) then $q_1 \neq q_2$ and so $a_1 \neq a_2$ and the verifier accepts. Otherwise, (i.e. if $q_1 = q_2$) then we have that $a_1 = a_2$ and again the verifier accepts. Hence, we have shown a strategy that breaks the soundness of the scheme with probability 1.

6.2 2-Round MPC from AFS-Spooky Encryption

AFS-spooky encryption seems to be a useful tool for minimally-interactive multi-party protocols: it lets each party broadcast an encryption of its input under its own key, then everyone individually performs the AFS-spooky evaluation locally, and each party can locally decrypt and recover a share of the output (relative to an additive n-out-of-n secret-sharing scheme). Finally another round of communication can be used to recover the secret from all the shares. Implementing this the approach requires attention to some details, such as ensuring that the spooky evaluation is deterministic (so that all the parties arrive at the same sharing) and making the shares simulatable. The latter can be done by having each party distribute a random additive sharing of 0 in the first round, and then adding all their received shares to their spooky generated share before broadcasting it in the second round.

A different (but similar) avenue for implementing 2-round MPC, is by reducing AFS-spooky encryption to *multi-key FHE with threshold decryption (TMFHE)*. This primitive was recently formalized by Mukherjee and Wichs [22], who showed how to use it to generically construct 2-round MPC. Just like spooky encryption, a TMFHE scheme can homomorphically process n ciphertexts $c_1, \ldots, c_n$, encrypting values $x_1, \ldots, x_n$ under independent public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, producing for any function f a ciphertext $c^* = \mathsf{Eval}(f, (\mathsf{pk}_1, c_1), \ldots, (\mathsf{pk}_n, c_n))$. The ciphertexts c^* cannot be decrypted by any single secret keys sk_i individually, but each party can compute a partial decryption $y_i = \mathsf{PartDec}_{\mathsf{sk}_i}(c^*)$ and these y's can be combined to get $y = \mathsf{FinDec}(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$. For security, Mukherjee and Wichs required that for each individual i, the partial decryption y_i can be simulated given the evaluated ciphertext c^*, the final output y and the secret key sk_j for $j \neq i$ (see [22] for formal definitions).

We observe that an AFS-spooky encryption with perfect correctness immediately yields a TMFHE scheme. The homomorphic evaluation procedure Eval of the TMFHE runs the $\mathsf{Spooky\text{-}Eval}$ procedure of the AFS-spooky encryption and sets $c^* = (c'_1, \ldots, c'_n)$ to be the resulting ciphertexts. The partial decryption procedure $\mathsf{PartDec}_{\mathsf{sk}_i}(c^*)$ outputs $y_i = \mathsf{Dec}_{\mathsf{sk}_i}(c'_i)$ and the combination procedure

$\mathsf{FinDec}(y_1, \ldots, y_n)$ outputs $y = \bigoplus_{i=1}^{n} y_i$. For security, we observe that each partial decryption y_i can be simulated given $c^* = (c'_1, \ldots, c'_n)$, y and sk_j for $j \neq i$ by computing $y_j = \mathsf{Dec}_{\mathsf{sk}_j}(c'_j)$ and setting $y_i = y \oplus (\bigoplus_{j \neq i} y_j)$.[9] This proves the following theorem.

Theorem 10. *An AFS-spooky encryption scheme with perfect correctness implies a multi-key FHE with threshold decryption (TMFHE).*

Using the above theorem and the results of [22] which constructs a 2-round MPC from TMFHE, we get the following corollaries.

Corollary 2. *Assuming the existence of a weak AFS-spooky encryption scheme:*

- *There exists a 2-round* MPC *protocol with semi-honest security. If the encryption scheme is in the plain model then so is the* MPC *protocol and if the encryption scheme requires a CRS then so does the* MPC *protocol.*
- *Furthermore, assuming the existence of NIZKs in the CRS model, there exists a 2-round* MPC *protocol with malicious security in the CRS model.*

Combining this with our construction of AFS-spooky encryption without a CRS from iO, we get the first construction of a 2-round semi-honest MPC protocol in the plain model.

Corollary 3. *Assume existence of (1)* piO *and (2) a lossy encryption scheme that is homomorphic for all single-bit to single-bit functions with perfect malicious circuit privacy. Then, there exists a 2-round* MPC *protocol with semi-honest security in the plain model.*

6.3 Function Secret Sharing

Function secret sharing (FSS), recently introduced by Boyle, Gilboa and Ishai [4], allows a dealer to split a function f into k succinctly described functions $\hat{f}_1, \ldots, \hat{f}_k$ such that (1) any strict subset of the $\hat{f}_i$'s reveals nothing about f and (2) for any x it holds that the values $\hat{f}_1(x), \ldots, \hat{f}_k(x)$ are an additive secret sharing of $f(x)$. Boyle *et al.* gave constructions under standard assumptions for certain restricted families of functions and a general construction for *any* poly-size circuit, based on piO. We show how to construct such a general FSS scheme given any AFS-spooky encryption scheme. In particular, we obtain a leveled FSS scheme assuming only LWE.

To construct such an FSS scheme, the dealer first generates a k-out-of-k secret sharing $f_1, \ldots, f_k$ of the *description* of the function f. The dealer also generates k key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)_{i \in [k]}$ for the AFS spooky scheme and publishes $\hat{f}_i \stackrel{\text{def}}{=} \big(\mathsf{sk}_i, \mathsf{pk}_1, \ldots, \mathsf{pk}_k, \mathsf{Enc}_{\mathsf{pk}_1}(f_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_k}(f_k)\big)$ as the i^{th} share. Assuming

[9] We note that imperfect correctness of the AFS-spooky scheme will translate into a security problem for the TMFHE scheme, as the simulated y_i will have a different distribution than the real ones.

the scheme is semantically secure, any strict subset of the $\hat{f}_i$'s hides the original function f (upto its description length).

For the FSS functionality, given an input x we can consider the circuit C_x that takes as input k shares of a function f, adds them up and applies the resulting function to the input x (which, say, is hardwired). To evaluate $\hat{f}_i$ on x, we run the spooky evaluation algorithm, which we assume wlog is deterministic, on $\mathsf{Enc}_{\mathsf{pk}_1}(f_1), \ldots, \mathsf{Enc}_{\mathsf{pk}_k}(f_k)$ with respect to the circuit C_x. Thus, given each $\hat{f}_i$ separately, we can generate the same ciphertexts $c_1, \ldots, c_k$ which are encryptions of an additive secret sharing of $f(x)$. Each function $\hat{f}_i$ can then be used to decrypt c_i and publish its share of $f(x)$.

A De-centralized View. We remark that the above construction can be viewed as a de-centralized FSS. More specifically, we may have some k (not necessarily secret or functional) shares $f_1, \ldots, f_k$ of a function f, where each share is owned by a different player. Player i can generate a key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ and broadcast $(\mathsf{pk}_i, \mathsf{Enc}_{\mathsf{pk}_i}(f_i))$ to all other players. Using our scheme, after learning the input x, the players can (non-interactively) generate an additive secret sharing of $f(x)$.

7 Spooky-Free Encryption

We turn now to study *spooky-free* encryption, i.e. an encryption scheme that ensures that no spooky relations can be realized by an adversary. The formal definition roughly states that any correlation that an attacker can induce between the original messages $(m_1, \ldots, m_n)$ and "tampered messages" $(m'_1, \ldots, m'_n)$, can be simulated by a "local simulator" that produces m'_i only as a function of m_i (and some shared randomness).

Definition 6 (Spooky-Free Encryption). *An encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* spooky-free *if for every PPT adversary $\mathcal{A}$ there exists a PPT simulator $\mathcal{S}$ such that for all PPT message distributions $\mathcal{D}$, the two distributions* $\mathbf{REAL}_{\mathcal{D},\mathcal{A}}(\kappa)$ *and* $\mathbf{SIM}_{\mathcal{D},\mathcal{S}}(\kappa)$ *specified below are computationally indistinguishable:*

$\underline{\mathbf{REAL}_{\mathcal{D},\mathcal{A}}(\kappa)}$:
1. *Sample* $(m_1, \ldots, m_n, \alpha) \leftarrow \mathcal{D}(1^\kappa)$; // *$\alpha$ is auxiliary information*
2. *Choose* $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(1^\kappa)$, *set* $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}_i}(m_i)$ *for* $i = 1, \ldots, n$;
3. *Let* $(c'_1, \ldots, c'_n) \leftarrow \mathcal{A}(\mathsf{pk}_1, \ldots, \mathsf{pk}_n, c_1, \ldots, c_n)$;
4. *Set* $m'_i = \mathsf{Dec}_{\mathsf{sk}_i}(c_i)$ *for* $i = 1, \ldots, n$;
5. *Output* $(m_1, \ldots, m_n, m'_1, \ldots, m'_n, \alpha)$.

$\underline{\mathbf{SIM}_{\mathcal{D},\mathcal{S}}(\kappa)}$:
1. *Sample* $(m_1, \ldots, m_n, \alpha) \leftarrow \mathcal{D}(1^\kappa)$; // *$\alpha$ is auxiliary information*
2. *Sample a random r, let* $m'_i = \mathcal{S}(1^\kappa, 1^n, i, m_i; r)$ *for* $i = 1, \ldots, n$;
3. *Output* $(m_1, \ldots, m_n, m'_1, \ldots, m'_n, \alpha)$.

It is not hard to see that spooky-freeness for $n \geq 2$ implies semantic security. As a small subtlety, here the attacker must choose the messages it claims to distinguish before seeing the public-key, since the message sampler $\mathcal{D}$ does not know anything public keys used in the real experiment. (We defined it this way since stronger security was not needed for our delegation application.) Of course, this minor difference from standard semantic security is without loss of generality when the message space is polynomial small (e.g., for bit encryption).

Lemma 5. *A spooky-free scheme for $n \geq 2$ is semantically secure (in the "selective" sense discussed above).*

Proof. Suppose that a scheme $(\mathsf{Enc}, \mathsf{Dec}, \mathsf{Gen})$ is *not semantically secure*, and let B be an attacker that can distinguish $\mathsf{Enc}_{\mathsf{pk}}(x_0)$ from $\mathsf{Enc}_{\mathsf{pk}}(x_1)$. We use B to construct a sampler $\mathcal{D}$ and attacker $\mathcal{A}$ that can fool any simulator $\mathcal{S}$ with non-negligible probability. We assume that $\mathcal{D}$ and $\mathcal{A}$ (and S) know the messages x_0 and x_1 whose encryption B can distinguish.

$\mathcal{D}$ draws at random $m_1 \leftarrow \{x_0, x_1\}$ and sets $m_i := 0$ for $i > 1$. Upon seeing n ciphertexts $c_1, \ldots, c_n$, $\mathcal{A}$ gives c_1 to B, asking him to guess whether it encrypts x_0 or x_1. Let σ be the guess that B makes, then we know that $m_1 = x_\sigma$ with probability $\geq 1/2 + \varepsilon$. $\mathcal{A}$ then sets $c'_i = c_i$ for all $i \neq 2$, and sets c'_2 to be a fresh encryption of x_σ under pk_2.

As we can see, the output of the real experiment has the tuple (m_1, m'_2) distributed as (x_b, x_σ), where b is a random bit and $\sigma = b$ with probability $\geq 1/2 + \varepsilon$. On the other hand, the simulator for the second message m'_2 is only given $m_2 = 0$ as the input, and has to guess σ' s.t., $\Pr[b = \sigma'] \geq 1/2 + \varepsilon$, which is impossible information-theoretically.

In the full version of this work [9] we show that spooky-free homomorphic encryption is exactly the ingredient needed to instantiate the idea of Aiello et al. [1] for converting general multi-prover (MIP) systems into single-prover arguments.[10] We also show there that non-malleable encryption is always spooky-free (albeit without any homomorphic capabilities), and we construct a spooky-free FHE scheme using a strong security component called succinct non-interactive argument of knowledge (SNARK).[11]

Spooky-Free Encryption with CRS. Definition 6 can be naturally extended to the common-reference-string model. We use this relaxation in the full version to gain somewhat better efficiency (at the price of a slightly harder proof of security). We note that, unlike the setting of spooky encryption from Sect. 3, we do not need the CRS to get the desired functionality, but rather use it only to improve efficiency. Our construction remains spooky-free (but slower) if all the public keys are chosen completely independently.

[10] An alternate route for instantiating the [1] idea due to [18,19] is to use special types of MIP, which satisfy a stronger soundness condition, together with any (possibly spooky) homomorphic encryption scheme.

[11] Of course, this construction does not give any new one-round delegation schemes, since SNARKs trivially imply the existence of such a scheme directly (i.e., without building spooky-free encryption). Still, if better constructions of spooky-free FHE are found, they would immediately imply new delegation schemes for NP.

Acknowledgments. This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant #CNS-1523467.

The first author was partially supported by gifts from VMware Labs and Google, and NSF grants 1319051, 1314568, 1065288, 1017471.

The second author was supported in part by the Defense Advanced Research Projects Agency (DARPA) and Army Research Office(ARO) under Contract No. W911NF-15-C-0236.

The third author was supported by NSF MACS - CNS-1413920, DARPA IBM - W911NF-15-C-0236, SIMONS Investigator award Agreement Dated 6-5-12 and DARPA NJIT - W911NF-15-C-0226.

The last author was supported in part by NSF grants CNS-1347350, CNS-1314722, CNS-1413964.

References

1. Aiello, W., Bhatt, S., Ostrovsky, R., Rajagopalan, S.R.: Fast verification of any remote procedure call: short witness-indistinguishable one-round proofs for NP. In: Welzl, E., Montanari, U., Rolim, J.D.P. (eds.) ICALP 2000. LNCS, vol. 1853, pp. 463–474. Springer, Heidelberg (2000)
2. Bitansky, N., Chiesa, A.: Succinct arguments from multi-prover interactive proofs and their efficiency benefits. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 255–272. Springer, Heidelberg (2012)
3. Boneh, D., Segev, G., Waters, B.: Targeted malleability: homomorphic encryption for restricted computations. In: Goldwasser, S. (ed.) Innovations in Theoretical Computer Science, Cambridge, MA, USA, 8–10 January 2012, pp. 350–366. ACM (2012)
4. Boyle, E., Gilboa, N., Ishai, Y.: Function secret sharing. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 337–367. Springer, Heidelberg (2015)
5. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) Symposium on Theory of Computing Conference, STOC 2013, Palo Alto, CA, USA, 1–4 June 2013, pp. 575–584. ACM (2013)
6. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (2015)
7. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (2015)
8. Cramer, R., Hanaoka, G., Hofheinz, D., Imai, H., Kiltz, E., Pass, R., Shelat, A., Vaikuntanathan, V.: Bounded CCA2-secure encryption. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 502–518. Springer, Heidelberg (2007)
9. Dodis, Y., Halevi, S., Rothblum, R.D., Wichs, D.: Spooky encryption and its applications. IACR Cryptology ePrint Archive 2016:272 (2016)
10. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient public-key cryptography in the presence of key leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)

11. Dwork, C., Langberg, M., Naor, M., Nissim, K., Reingold, O.: Succinct proofs for NP and spooky interactions (2004, Unpublished manuscript). http://www.cs.bgu.ac.il/~kobbi/papers/spooky_sub_crypto.pdf
12. Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 74–94. Springer, Heidelberg (2014)
13. Gentry, C., Halevi, S., Vaikuntanathan, V.: i-hop homomorphic encryption and rerandomizable Yao circuits. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 155–172. Springer, Heidelberg (2010). http://eprint.iacr.org/2010/145
14. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: STOC, pp. 99–108 (2011)
15. Gilboa, N., Ishai, Y.: Distributed point functions and their applications. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 640–658. Springer, Heidelberg (2014)
16. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, New York, USA, pp. 218–229 (1987)
17. Ishai, Y., Kushilevitz, E.: Randomizing polynomials: a new representation with applications to round-efficient secure computation. In: 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12–14 November 2000, Redondo Beach, California, USA, pp. 294–304 (2000)
18. Kalai, Y.T., Raz, R., Rothblum, R.D.: Delegation for bounded space. In: STOC, pp. 565–574 (2013)
19. Kalai, Y.T., Raz, R., Rothblum, R.D.: How to delegate computations: the power of no-signaling proofs. In: Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31–June 03 2014, pp. 485–494 (2014)
20. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 335–354. Springer, Heidelberg (2004)
21. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, 19–22 May 2012, pp. 1219–1234 (2012)
22. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 735–763. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49896-5_26. http://eprint.iacr.org/2015/345
23. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, 7–9 January 2001, Washington, DC, USA, pp. 448–457 (2001)
24. Ostrovsky, R., Paskin-Cherniavsky, A., Paskin-Cherniavsky, B.: Maliciously circuit-private FHE. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 536–553. Springer, Heidelberg (2014). https://eprint.iacr.org/2013/307
25. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31–June 2 2009, pp. 333–342. ACM (2009)

26. Peikert, C., Shiehian, S.: Multi-key FHE from LWE, revisited. Cryptology ePrint Archive, report 2016/196 (2016). http://eprint.iacr.org/
27. Petrank, E.: The hardness of approximation: gap location. Comput. Complex. **4**, 133–157 (1994)
28. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 34:1–34:40 (2009)