2018 IEEE 59th Annual Symposium on Foundations of Computer Science

Laconic Function Evaluation and Applications

Willy Quach
Northeastern University
quach.w@husky.neu.edu

Hoeteck Wee
CNRS and ENS
wee@di.ens.fr

Daniel Wichs
Northeastern University
wichs@ccs.neu.edu

***Abstract*—We introduce a new cryptographic primitive called *laconic function evaluation* (LFE). Using LFE, Alice can compress a large circuit f into a small digest. Bob can encrypt some data x under this digest in a way that enables Alice to recover $f(x)$ without learning anything else about Bob's data. For the scheme to be laconic, we require that the size of the digest, the run-time of the encryption algorithm and the size of the ciphertext should all be small, much smaller than the circuit-size of f. We construct an LFE scheme for general circuits under the learning with errors (LWE) assumption, where the above parameters only grow polynomially with the depth but not the size of the circuit. We then use LFE to construct secure 2-party and multi-party computation (2PC, MPC) protocols with novel properties:**

- **We construct a 2-round 2PC protocol between Alice and Bob with respective inputs x_A, x_B in which Alice learns the output $f(x_A, x_B)$ in the second round. This is the first such protocol which is "Bob-optimized", meaning that Alice does all the work while Bob's computation and the total communication of the protocol are smaller than the size of the circuit f or even Alice's input x_A. In contrast, prior solutions based on fully homomorphic encryption are "Alice-optimized".**
- **We construct an MPC protocol, which allows N parties to securely evaluate a function $f(x_1, ..., x_N)$ over their respective inputs, where the total amount of computation performed by the parties *during* the protocol execution is smaller than that of evaluating the function itself! Each party has to individually *pre-process* the circuit f before the protocol starts and *post-process* the protocol transcript to recover the output after the protocol ends, and the cost of these steps is larger than the circuit size. However, this gives the first MPC where the computation performed by each party during the actual protocol execution, from the time the first protocol message is sent until the last protocol message is received, is smaller than the circuit size.**

I. Introduction

We introduce a new and natural cryptographic primitive, which we call *laconic function evaluation (LFE)*. In an LFE scheme, Alice has a large circuit f, potentially containing various hard-coded data. She can deterministically compress f to derive a short digest $\mathsf{digest}_f = \mathsf{Compress}(f)$. Bob can encrypt some data x under this digest, resulting in a ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{digest}_f, x)$, which Alice is able to decrypt using only her knowledge of f to recover the output $f(x) = \mathsf{Dec}(f, \mathsf{ct})$. Security ensures that Alice does not learn anything about Bob's input x beyond the output $f(x)$, as formalized via the simulation paradigm. The *laconic* aspect of LFE requires that Bob's computational complexity is small, and in particular, the size of digest_f, the run-time of the encryption algorithm $\mathsf{Enc}(\mathsf{digest}_f, x)$ and the size of the ciphertext ct should be much smaller than the circuit-size of f.

As an example, we can imagine that the FBI (acting as Alice) has a huge database D of suspected terrorists and publishes a short digest of the circuit $f_D(x)$ which checks if a person x belongs to the database D. An airline (acting as Bob) can use this digest to encrypt the identity of a passenger on their flight manifest, which lets the FBI learn whether the passenger is on the suspected terrorist list, while preserving passenger privacy otherwise.[1] This is a special case of secure 2-party computation and indeed we will show that LFE has applications to general secure 2-party and multi-party computation with novel properties which weren't achievable using previously known techniques.

We emphasize that, in spite of all of the recent advances in succinct computation on encrypted data (from fully homomorphic encryption through functional encryption), the very natural notion of LFE has not been considered in the literature, nor does it follow readily from existing cryptographic tools and primitives. We discuss the most related primitives below.

a) Relation to Laconic OT: LFE can be seen as a generalization of the recently introduced notion of *laconic oblivious transfer (LOT)* [CDG+17]. In an LOT scheme, Alice has a large database $D \in \{0,1\}^M$ which she compresses into a short digest $\mathsf{digest}_D = \mathsf{Compress}(D)$. Bob can choose any location $i \in [M]$ and two messages m_0, m_1, and create a ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{digest}_D, (i, m_0, m_1))$. Alice recovers the message $m_{D[i]}$, where $D[i]$ is the i'th bit of D, without learning anything about the other message $m_{1-D[i]}$. In other words, we can think of LOT as a highly restricted form of LFE for functions of the form $f_D^{\text{LOT}}((i, m_0, m_1)) = (i, m_{D[i]})$, whereas LFE works for arbitrary circuits.[2] Using the ideas of [CDG+17], it is possible to combine LOT with garbled circuits to achieve a "half laconic" $\frac{1}{2}$LFE scheme for arbitrary circuits, where the digest is short but the run-time of the encryption algorithm and the size of the ciphertext are larger than the circuit size of f. (Essentially, Alice sends an LOT digest corresponding to a description of the circuit f and Bob sends a garbled universal circuit that takes as input f and outputs $f(x)$, along with LOT ciphertexts

[1] We may also desire "function privacy", discussed later, to ensure that the digest does not reveal anything about D.

[2] In fact, LOT can be seen as a restricted form of "attribute-based LFE" (AB-LFE) discussed later.

2575-8454/18/$31.00 ©2018 IEEE
DOI 10.1109/FOCS.2018.00086

for the labels of the input wires.) Although such $\frac{1}{2}$LFE is already interesting and can be constructed under several different assumptions (e.g., DDH, LWE, etc.) by leveraging the recent works of [CDG+17], [DG17], [BLSV18], it will be insufficient for our applications which crucially rely on a fully laconic LFE scheme.

b) Relation to Functional Encryption: LFE also appears to be related to (succinct, single-key) *functional encryption* (FE) [SW05], [BW07], [KSW08], [SS10], [BSW11], [GVW12], [GVW13], [GKP+13], [BGG+14], [GVW15], [Agr17]. However, despite some similarity, the two primitives are distinct and have incomparable requirements. An FE scheme has a master authority which creates a master public key mpk and master secret key msk. For any function f, the master authority can use msk to generate a function secret key sk_f, which it can give to Alice. Bob can then use mpk to compute an encryption $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x)$ of some value x, which Alice can decrypt using sk_f to recover $f(x)$ without learning anything else about x. In a succinct FE scheme, the size of mpk, the run-time of the encryption algorithm and the ciphertext size are smaller than the circuit size of f. There are important differences between FE and LFE:

- In FE there is a master authority which is a separate party distinct from the users Alice/Bob and which gives sk_f to Alice and mpk to Bob. In LFE there is no such additional authority.
- In FE the ciphertext ct is not tied to any specific function f, but is created using a master public key mpk. Depending on which secret key sk_f Alice gets she is able to decrypt $f(x)$, but the master authority is able to learn all of x. In LFE, the ciphertext ct is created using digest_f which ties it to a specific function f, and the ciphertext does not reveal anything beyond $f(x)$ to anyone.

It turns out that LFE *generically* implies succinct, single-key FE (see the full version for more details [QWW18]).[3] However, we do not know of any implication in the reverse direction and it appears unlikely that succinct FE would imply LFE. Nevertheless, as we will discuss later, our concrete construction of LFE under the learning with errors assumption borrows heavily from techniques developed in the context of attribute-based encryption and succinct functional encryption.

A. Main Results for LFE

In this work, we construct an LFE scheme under the *learning with errors (LWE)* assumption. The size of the digest, the complexity of the encryption algorithm and the size of the ciphertext only scale with the depth but not the size of the circuit. In particular, for a circuit $f : \{0,1\}^k \to \{0,1\}^\ell$ of size $|f|$ and depth d, and for security parameter λ, our LFE has the following parameters:

- The size of the digest is $\mathsf{poly}(\lambda)$ and the run-time of the encryption algorithm and the size of the ciphertext are $\widetilde{O}(k+\ell) \cdot \mathsf{poly}(\lambda, d)$.
- The run-time of the compression and the decryption algorithms is $\widetilde{O}(|f|) \cdot \mathsf{poly}(\lambda, d)$.

As in many other prior works building advanced cryptosystems under LWE, we require LWE with a sub-exponential modulus-to-noise ratio, which follows from the hardness of worst-case lattice problems with sub-exponential approximation factors and is widely believed to hold.

a) Necessity of a CRS: It turns out that LFE schemes require some sort of a public seed, which we refer to as a *common reference string (CRS)*. This is for the same reason that collision-resistant hash functions (CRHFs) need a public seed; in fact, it is easy to show that the compression function which maps a circuit f to a digest digest_f is a CRHF. In our case, the CRS is a *uniformly random string*, which is given as an input to all of the algorithms of the LFE. The CRS is of size $k \cdot \mathsf{poly}(\lambda, d)$. For simplicity, we will usually ignore the CRS in our simplified notation used throughout the introduction.

b) Selective vs. Adaptive Security: One caveat of our schemes is that we only achieve *selective* security where Bob's input x must be chosen non-adaptively before the CRS is known. We also consider *adaptive* security where Bob's input x can adaptively depend on the CRS and show that it follows from a natural and easy to state strengthening of the LWE assumption, which we call *adaptive LWE* (see the "Our Techniques" section). Currently, we can only prove the security of adaptive LWE from standard LWE via a reduction with an exponential loss of security, but it may be reasonable to assume that the actual security level of adaptive LWE is much higher than this reduction suggests.

c) Function Hiding: For the default notion of LFE, the compression function that computes $\mathsf{digest}_f = \mathsf{Compress}(f)$ is deterministic and the output digest_f may reveal some partial information about the circuit f. However, in many cases we may also want a *function hiding* property, meaning that the digest should completely hide the function f. In this case the compression function has to use private randomness. We show that there is a generic way to convert any LFE scheme with a deterministic compression function and without function hiding into one which has a randomized compression function and is also *statistically* function hiding. We also give an alternate more direct approach to achieving statistical function hiding for our LWE-based construction of LFE.

In a function-hiding LFE with a randomized compression function, Alice will also need to remember the random coins r that she used to create $\mathsf{digest}_f = \mathsf{Compress}(f; r)$ as a secret key, which is needed to decrypt ciphertexts created under this digest. One additional advantage of an LFE with function hiding security is that, while previously anybody (not just Alice) could decrypt a ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{digest}_f, x)$ to recover $f(x)$ by just knowing the function f, the function hiding property implicitly guarantees that

[3] We thank Alex Lombardi for pointing out this implication to us.

only Alice can recover $f(x)$ using her private randomness r, while others do not learn anything about x. This is because an external observer cannot differentiate between a correctly generated digest_f and digest_{null}, where the latter is the digest of the null function that always outputs 0. Given digest_{null} and a ciphertext ct encrypting x the attacker does not learn anything about x even given the randomness used to generate the digest.

We note that, although the compression function in function-hiding LFE is randomized, our compiler guarantees that Bob's security holds even if Alice chooses the randomness of the compression function maliciously.

d) LFE implies succinct FE: We show that selective (respectively adaptive) LFE generically implies succinct, single-key selective (respectively adaptive) FE. In a nutshell, this is because, starting from a non-succinct FE, we can build a succinct version by using the *LFE encryption* as the function for the non-succinct FE. Now during decryption, we can generate an LFE encryption using the non-succinct FE, and decrypt the output with the LFE. Meanwhile, succinctness is guaranteed as the circuit which computes an LFE encryption is small.

B. Applications of LFE

As an application of LFE, we construct 2-party and multi-party computation (2PC, MPC) protocols with novel properties that weren't previously known.

a) Bob-Optimized 2-Round 2PC: Consider a 2-round 2PC protocol between Alice and Bob with respective inputs x_A, x_B where Alice learns the output. Without loss of generality, Alice initiates the protocol by sending the first round message to Bob and learns the output $y = f(x_A, x_B)$ after receiving the second round message from Bob.

If we didn't care about security we would have two basic *insecure* approaches to the problem. The "Alice-optimized" approach is for Alice to send her input x_A to Bob and for Bob to compute $y = f(x_A, x_B)$ and send it to Alice; Alice's computation and the total communication of the protocol are equal to $|x_A| + |y|$ while Bob's computation is equal to $|f|$. The "Bob-optimized" approach is for Alice to ask Bob to send her his input x_B and for Alice to then compute $y = f(x_A, x_B)$; Bob's computation and the total communication of the protocol are equal to $|x_B|$ while Alice's computation is equal to $|f|$. The second approach appears more natural – after all, if Alice wants to learn the output, she should do the work!

Can we get *secure* analogues of the above two approaches? Using garbled circuits and oblivious transfer, we get a protocol which is neither Alice-optimized nor Bob-optimized; the computation of both parties and the total communication are linear in the circuit size $|f|$. Using laconic OT, Alice's first message can become short but there is no improvement in either party's computation or the total communication otherwise. Using fully homomorphic encryption (FHE) [Gen09], [BV11], [GSW13], [BV14], we can get an Alice-optimized protocol: Alice encrypts her input x_A in the first round and Bob homomorphically computes an encryption of $f(x_A, x_B)$ in the second round. Alice's computation and the total communication of the protocol are $(|x_A| + |y|) \cdot \mathsf{poly}(\lambda)$ while Bob's computation is $|f| \cdot \mathsf{poly}(\lambda)$. But can we get a Bob-optimized protocol? Previously, no such 2-round protocol was known.[4]

Approach	CRS	Communication			Computation		Assumptions
		Alice +	Bob =	Total	Alice	Bob	
Garbled Circuits	–	k_A	$\|f\|$	$\|f\|$	$\|f\|$	$\|f\|$	OT
Laconic OT	$O(1)$	$O(1)$	$\|f\|$	$\|f\|$	$\|f\|$	$\|f\|$	DDH, etc.
FHE	–	k_A	ℓ	$k_A + \ell$	$k_A + \ell$	$\|f\|$	LWE
Our Work	k_B	$O(1)$	$k_B + \ell$	$k_B + \ell$	$\|f\|$	$k_B + \ell$	LWE

Fig. 1. *Summary of semi-honest 2-round 2PC where Alice holds $x_A \in \{0,1\}^{k_A}$, Bob holds $x_B \in \{0,1\}^{k_B}$, and Alice learns $y = f(x_A, x_B) \in \{0,1\}^{\ell}$. We suppress multiplicative factors that are polynomial in the security parameter λ or, for the last row, the circuit depth d.*

In this work, using LFE, we get the first "Bob-optimized" 2-round protocol where Bob's computation and the total communication are smaller than the circuit size or even the size of Alice's input – in particular, they are bounded by $(|x_B| + |y|) \cdot \mathsf{poly}(\lambda, d)$ with d being the depth of the circuit f. The linear dependence on the output size $|y|$ is inherent for any (semi-malicious) secure protocol, as shown in [HW15] but it remains an interesting open problem to get rid of the dependence on the circuit depth d. We summarize the comparison of different approaches in Figure 1.

Our protocol using LFE is extremely simple: Alice sends a digest for the function $f(x_A, \cdot)$ in the first round and Bob uses the digest to encrypt x_B in the second round. To get security for Alice, we use a function-hiding LFE scheme and we even ensure that Alice's security holds statistically. The above gives us a 2PC with semi-honest security. We can think of this as a protocol in the CRS model, or we can even allow Alice to choose the CRS on her own and send it in the first round to get a protocol in the plain model. If we think of it as a protocol in the CRS model, we even get "semi-malicious" security where corrupted parties follow the protocol but can use maliciously chosen randomness. To get fully malicious security we would need to additionally rely on succinct non-interactive zero-knowledge arguments of knowledge (ZK-SNARKs) [Mic94], [Gro10], [BCCT13], [BCI+13], [GGPR13] and have Alice prove that she computed the digest correctly.

b) MPC with Small Online Computation: As our second application, we construct an MPC protocol that allows N parties to securely evaluate some function $f(x_1, ..., x_N)$ over their respective inputs, where the total amount of computation performed by the parties during the protocol execution is smaller than that of evaluating the function itself! Of course, the work of the computation must be performed at some point, even if we didn't care about security. In our case, the MPC protocol requires each party to individually *pre-process* the function f step before the

[4]If we allow additional rounds, we can use FHE to get a Bob-optimized 3-round protocol: Bob encrypts his input x_B under FHE, Alice homomorphically computes $f(x_A, x_B) \oplus k$ where k is a random one-time pad, Bob decrypts and sends the plaintext $f(x_A, x_B) \oplus k$ to Alice who recovers $y = f(x_A, x_B)$.

protocol starts and to *post-process* the protocol transcript and recover the output after the protocol ends, where the computational complexity of these steps can exceed the circuit size of f. The pre-processing step is deterministic and can be performed once per function being evaluated and reused across many executions. However, the main novelty of our MPC is that the total computation performed by the parties *online*, from the time they send the first protocol message and until they receive the last protocol message, is much smaller than the circuit size of f. This also implies a correspondingly small communication complexity of the protocol.

(We note there are prior MPC protocols that have a separate "offline" phase, which occurs before the inputs of the computation are known, and an "online" phase which occurs after the inputs are known and where the online phase is very efficient. However, in these schemes the term "offline" is a misnomer since the offline phase involves running a protocol and interacting with the other parties. For example, the parties may run an MPC protocol to compute a garbled circuit for the function f in the offline phase and then, once their inputs are known, they run another much more efficient MPC protocol to compute the garbled input for the garbled circuit. In contrast, ours is the first MPC that has a truly offline pre-processing and post-processing phase, where the parties do not interact with each other, while the computational complexity of the entire online phase that involves interaction is smaller than the circuit size.)

Our MPC construction uses an LFE scheme and proceeds as follows:

Pre-processing (offline). Each party individually pre-processes the circuit f by locally computing $\mathsf{digest}_f = \mathsf{Compress}(f)$. We rely on an LFE scheme with deterministic compression so all parties compute the same digest. This step can be performed once and can be reused across many executions.

Actual protocol (online). The actual protocol execution between the N parties holding inputs x_i just invokes a generic MPC protocol to compute an LFE ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{digest}_f, (x_1, \ldots, x_N))$ which encrypts their joint inputs. Here, we use the fact that the run-time of Enc is small.

Post-processing (offline). After the protocol finishes, each party individually does a post-processing step in which it runs the LFE decryption algorithm on ct to recover the output $f(x_1, \ldots, x_N)$.

The computational complexity of the pre-processing and post-processing steps is $|f| \cdot \mathsf{poly}(\lambda, d)$, where d is the depth of the circuit f. The computational complexity of each party in the online protocol execution and the total communication complexity are only $\mathsf{poly}(\lambda, d, k, \ell, N)$, where k is the input size of each party and ℓ is the output size and N is the number of parties. In particular, the communication and computational complexity of the online phase can be much smaller than the circuit size of f. We inherit semi-honest or fully-malicious security depending on the MPC used in online phase. We think of the above protocol as being in the CRS model. However, in the case of semi-honest security, we can also think of it as a protocol in the plain model by having a single designated party (e.g., party 1) choose the CRS for the LFE and compute the digest digest_f (no other parties do any pre-processing) and use these as its inputs to the MPC protocol executed in the online phase.

By taking our scheme from the previous paragraph and using a 2-round MPC [MW16], [PS16], [BP16], [GS17], [BL18], [GS18] in the online phase (without any additional efficiency constraints), we get a 2-round MPC where the total computation performed by each party (including pre-processing and post-processing) is $|f| \cdot \mathsf{poly}(\lambda, d) + \mathsf{poly}(\lambda, k, \ell, d, N)$ and the total communication is $\mathsf{poly}(\lambda, k, \ell, d, N)$. In all prior 2-round MPC protocols with N parties (even with semi-honest security, even in the CRS model), the computation performed by each party is at least $|f| \cdot N^2$. Therefore our result gives improved computational efficiency for 2-round MPC even if we did not distinguish between online and offline work. In the case of semi-honest security, if we use a 2-round MPC in the plain model [GS17], [BL18], [GS18] in the online phase we get a 2-round MPC in the plain-model with the above efficiency. In particular, we get a semi-honest 2-round MPC in the plain model with communication which is smaller than the circuit size, matching the recent work of Ananth et al. [ABJ+18] which previously got such result using functional-encryption combiners.

C. Our Techniques

Our construction of LFE relies on an adaptation of techniques developed in the context of attribute-based and functional encryption [BGG+14], [GKP+13]. The construction proceeds in two steps. We start by considering LFE for a restricted class of functionalities, which we call *attribute-based LFE (AB-LFE)* in analogy to attribute-based encryption (ABE). In an AB-LFE, Alice computes a digest digest_f for a function f. Bob computes a ciphertext $\mathsf{Enc}(\mathsf{digest}_f, x, \mu)$ with an attribute x and a message μ such that Alice recovers μ if $f(x) = 0$ and otherwise doesn't learn anything about μ. (For technical reasons, it will be easier to use the semantics where 0 denotes "qualified to decrypt" and 1 denotes "unqualified to decrypt".) However, the attribute x is always revealed to Alice. We can think of AB-LFE as an LFE for the "conditional disclosure functionality" $\mathsf{CDF}[f](x, \mu)$, which outputs (x, μ) if $f(x) = 0$ and $(x, \perp)$ otherwise. As our first step, we construct AB-LFE under the LWE assumption. As our second step, we show how to generically compile any AB-LFE into an LFE by additionally relying on fully homomorphic encryption.

a) First Step: Constructing AB-LFE: Our construction adapts the techniques developed by Boneh et al. [BGG+14] to construct attribute-based encryption (ABE). In some sense, our construction of AB-LFE is simpler than that of ABE and we essentially avoid relying on "lattice trapdoors". Our proof of security also departs significantly from that

of [BGG+14] and appears to be simpler since we avoid embedding lattice trapdoors in the CRS. One advantage of our simplified proof is that, while for both ABE and AB-LFE we only get *selective* security under LWE, for AB-LFE (but not ABE) our proof extends to showing that *adaptive* security follows from a simple to state and natural "adaptive LWE" assumption, which we describe below. We note that, although we manage to construct AB-LFE using similar techniques as previously used in constructing of ABE, there does not appear to be a "black-box" relationship between these primitives where one would imply the other.

We rely on two algorithms $\mathsf{EvalPK}, \mathsf{EvalCT}$ that were defined by [BGG+14]. Let $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ be a fixed "gadget matrix" from [MP12]. Let $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$ be arbitrary matrices, $\mathbf{b}_i \in \mathbb{Z}_q^m$ be vectors, $f : \{0,1\}^k \to \{0,1\}$ be some circuit and $x \in \{0,1\}^k$ be an input.

- $\mathsf{EvalPK}(f, \{\mathbf{A}_i\}_{i \in [k]})$ outputs a matrix $\mathbf{A}_f \in \mathbb{Z}_q^{n \times m}$.
- $\mathsf{EvalCT}(f, \{\mathbf{A}_i\}_{i \in [k]}, \{\mathbf{b}_i\}_{i \in [k]}, x)$ outputs a vector $\mathbf{b}_f \in \mathbb{Z}_q^n$.

These algorithms are deterministic and have the property that if $\mathbf{b}_i = \mathbf{s}^\top (\mathbf{A}_i - x_i \mathbf{G}) + \mathbf{e}_i$ for $i \in [k]$ then:

$$\mathbf{b}_f = \mathbf{s}^\top (\mathbf{A}_f - f(x)\mathbf{G}) + \mathbf{e}^* \tag{1}$$

where $\mathbf{e}_i$ and $\mathbf{e}^*$ are some "small" errors. Our basic AB-LFE scheme works as follows.

- The CRS consists of uniformly random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_k$ with $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$.
- To compress a circuit $f : \{0,1\}^k \to \{0,1\}$ we set the digest to be $\mathsf{digest}_f = \mathbf{A}_f = \mathsf{EvalPK}(f, \{\mathbf{A}_i\}_{i \in [k]})$.
- The encryption algorithm $\mathsf{Enc}(\mathsf{digest}_f, x, \mu)$ encrypts a message $\mu \in \{0,1\}$ with respect to an attribute $x \in \{0,1\}^k$ under a digest $\mathsf{digest}_f = \mathbf{A}_f$. It chooses a random LWE secret $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and computes LWE samples $\mathbf{b}_i = \mathbf{s}^\top (\mathbf{A}_i - x_i \mathbf{G}) + \mathbf{e}_i$. It also chooses a random "short" vector $\mathbf{t}$ and sets $\mathbf{d} = \mathbf{A}_f \cdot \mathbf{t}$. Lastly it sets $\beta = \langle \mathbf{s}, \mathbf{d} \rangle + e' + \mu \cdot \lfloor q/2 \rceil$ and outputs the ciphertext $\mathsf{ct} = (\{\mathbf{b}_i\}, \beta, \mathbf{t}, x)$.
- The decryption algorithm computes $\mathbf{b}_f = \mathsf{EvalCT}(f, \{\mathbf{A}_i\}_{i \in [k]}, \{\mathbf{b}_i\}_{i \in [k]}, x)$. By (1), we have $\mathbf{b}_f = \mathbf{s}^\top (\mathbf{A}_f - f(x)\mathbf{G}) + \mathbf{e}^*$. If $f(x) = 0$ then this allows us to recover the message by computing $\beta - \langle \mathbf{b}_f, \mathbf{t} \rangle \approx \mu \cdot \lfloor q/2 \rceil$.

To prove security, we assume that $f(x) = 1$ and need to show that the ciphertext doesn't reveal anything about the encrypted message μ. We can rely on the LWE assumption with a uniformly random secret $\mathbf{s}$ and coefficient $(\mathbf{A}_i - x_i \mathbf{G})$ to argue that the samples $\mathbf{b}_i$ are indistinguishable from uniform. However, to argue that β is also indistinguishable from uniform is slightly more complex. Firstly, we note that by (1), we have $\beta \approx \langle \mathbf{b}_f, \mathbf{t} \rangle + \mathbf{s}^\top \mathbf{G} \mathbf{t} + e' + \mu \cdot \lfloor q/2 \rceil$. Secondly, if we set $\mathbf{u} = \mathbf{G}\mathbf{t}$, then $\mathbf{u}$ is uniformly random and $\mathbf{t} \leftarrow \mathbf{G}^{-1}(\mathbf{u})$ can be efficiently sampled from $\mathbf{u}$, so we can efficiently sample β given $\mathbf{u}$, $\langle \mathbf{s}, \mathbf{u} \rangle + e' + \mu \cdot \lfloor q/2 \rceil$. Therefore, when we use the LWE assumption, we also get one additional sample with the coefficients $\mathbf{u}$ which we use to argue that β is indistinguishable from uniform. There is some subtlety in ensuring that the noise distribution in β is correct and we solve this using the standard "noise smudging" technique.

b) Adaptively Secure AB-LFE from Adaptive LWE: The above proof shows selective security, where the adversary chooses the attribute x ahead of time before seeing the CRS, but breaks down in the case of adaptive security. The issue is that, if the attribute x is chosen adaptively after the adversary sees the CRS = $\{\mathbf{A}_i\}$, then we can no longer argue that the LWE coefficients $(\mathbf{A}_i - x_i \mathbf{G})$ are uniformly random. However, the adversary has extremely limited ability to manipulate the coefficients. We formulate a new but natural "adaptive LWE" assumption where the adversary is first given matrices $\{\mathbf{A}_i\}_{i \in [k]}$, then adaptively chooses a value $x \in \{0,1\}^k$, and has to distinguish between LWE samples $\mathbf{s}^\top (\mathbf{A}_i - x_i \mathbf{G}) + \mathbf{e}_i$ and uniformly random values. Our proof above shows adaptive security of AB-LFE under adaptive LWE.

What can we say about adaptive LWE? A simple guessing argument allows us to prove the security of adaptive LWE based on standard LWE by incurring a 2^k security loss in the reduction. In other words, adaptive LWE follows from the sub-exponential security of standard LWE, if we choose the parameters appropriately. However, it seems plausible to assume that adaptive LWE has a much higher level of security than this reduction suggests. As far as we can tell, it seems reasonable to assume that it could have essentially the same level of security as standard LWE and perhaps there is a reduction which only incurs a polynomial loss. We leave this fascinating question for future work.

Another interesting question is whether the adaptive LWE assumption could be useful in proving adaptive security in other contexts, most notably for attribute-based encryption (ABE) of Boneh et al. [BGG+14]. As far as we can tell, this does not appear to be the case, and the reason that we are able to prove adaptive security of AB-LFE under adaptive LWE is that our proof differs significantly from that of the Boneh et al. ABE and does not rely on embedding lattice trapdoors in the CRS. Nevertheless, we leave this as an interesting possibility for future work.

c) Second Step: From AB-LFE to LFE: As our second step, we show how to generically compile AB-LFE to LFE using fully-homomorphic encryption (FHE). This compiler is essentially identical to that of Goldwassser et al. [GKP+13], which showed how to compile attribute-based encryption (ABE) to (single key) functional encryption (FE). In addition to FHE, the compiler relies on garbled circuits which can be constructed from one-way functions.

The high level idea is that, in the LFE scheme, the encryptor Bob first uses an FHE to encrypt his input x, resulting in an FHE ciphertext $\widehat{x}$. He then constructs a garbled circuit $\widehat{C}$ which has the FHE secret key hard-coded inside of it and performs an FHE decryption. Lastly, he uses an AB-LFE scheme to encrypt all of the labels of the input wires of the garbled circuit with respect to the attribute $\widehat{x}$

in such a way that Alice recovers exactly the garbled labels that correspond to the homomorphically evaluated ciphertext $\widehat{f(x)} = \mathsf{FHE.Eval}(f, \widehat{x})$. Bob sends $\widehat{x}, \widehat{C}$ and the AB-LFE ciphertexts to Alice as his LFE ciphertext.

d) Optimizing for Long Output and Final Parameters: The above ideas, combined together, give an LFE scheme for circuits $f : \{0,1\}^k \to \{0,1\}$ with 1-bit output and depth d, where the digest is of size $\mathsf{poly}(\lambda, d)$ and the encryption run-time and ciphertext size is $k \cdot \mathsf{poly}(\lambda, d)$. The compression and decryption run-time is $|f| \cdot \mathsf{poly}(\lambda, d)$.

A naive way to get an LFE for circuits $f : \{0,1\}^k \to \{0,1\}^\ell$ with ℓ-bit output is to invoke a separate LFE for each output bit. This would blow up all of the efficiency measures by a multiplicative factor of ℓ. It turns out that we can do better. We do so by first constructing a "multi-bit output AB-LFE" that allows Alice to compute a digest digest_f for a circuit $f : \{0,1\}^k \to \{0,1\}^\ell$ with ℓ-bit output and for Bob to create a single ciphertext with an attribute x and messages $\mu_1, \ldots, \mu_\ell$ such that Alice recovers μ_i if and only if the i'th output bit of $f(x)$ is 0. We show how to modify our LWE based AB-LFE construction to get a "multi-bit output AB-LFE" where the encryption run-time and the ciphertext size is $(k + \ell) \cdot \mathsf{poly}(\lambda, d)$ and the compression/decryption run-times remains $|f| \cdot \mathsf{poly}(\lambda, d)$. However, the digest size in this construction grows by a factor of ℓ to $\ell \cdot \mathsf{poly}(\lambda, d)$. We then show how to compress the digest further to just $\mathsf{poly}(\lambda)$ independent of ℓ, d. Essentially, instead of using the original AB-LFE digest as is, we employ an additional layer of laconic OT (LOT) and give out an LOT digest of the AB-LFE digest. The encryptor creates a garbled version of the underlying AB-LFE encryption algorithm and encrypts the labels for the garbled circuit under LOT.

Combining all of the above we get an LFE scheme where the size of the CRS is $k \cdot \mathsf{poly}(\lambda, d)$, the size of the digest is $\mathsf{poly}(\lambda)$, the run-time of the encryption algorithm and the size of the ciphertext are $\widetilde{O}(k + \ell) \cdot \mathsf{poly}(\lambda, d)$ and the run-time of the compression and the decryption algorithms is $\widetilde{O}(|f|) \cdot \mathsf{poly}(\lambda, d)$.

II. Organization

We define the notion of LFE in Section IV. We give a construction of AB-LFE with a single-bit output from LWE in Section V. Due to space limitations we only present an AB-LFE with single-bit outputs and messages in this proceedings version. We refer to the full version [QWW18] for a more efficient AB-LFE supporting multi-bit outputs, and for the transformation from AB-LFE to LFE using FHE. We then describe in Section VI our MPC protocol with small online computation.

We refer to the full version for other results discussed above. This includes how to generically add function hiding security to LFE, the necessity of a CRS, and how to generically build a succint single-key Functional Encryption scheme from any LFE.

III. Notations and preliminaries

We will denote by λ the security parameter. The notation $\mathsf{negl}(\lambda)$ denotes any function f such that $f(\lambda) = \lambda^{-\omega(1)}$, and $\mathsf{poly}(\lambda)$ denotes any function f such that $f(\lambda) = \mathcal{O}(\lambda^c)$ for some $c > 0$. For a probabilistic algorithm $\mathsf{alg}(\mathsf{inputs})$, we might explicit the randomness it uses by writting $\mathsf{alg}(\mathsf{inputs}; \mathsf{coins})$. We will denote vectors by bold lower case letters (e.g. $\mathbf{a}$) and matrices by bold upper cases letters (e.g. $\mathbf{A}$). We will denote by $\mathbf{a}^\top$ and $\mathbf{A}^\top$ the transposes of $\mathbf{a}$ and $\mathbf{A}$, respectively. We will denote by $\lfloor x \rceil$ the nearest integer to x, rounding towards 0 for half-integers.

Definition 1 (B-bounded distribution)**.** *We say that a distribution χ over $\mathbb{Z}$ is B-bounded if*

$$\Pr[\chi \in [-B, B]] = 1.$$

We recall the definition of the (decision) *Learning with Errors* problem, introduced by Regev in [Reg05].

Definition 2 ((Decision) Learning with Errors [Reg05])**.** *Let $n = n(\lambda)$ and $q = q(\lambda)$ be integer parameters and $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}$. The Learning with Errors (LWE) assumption $LWE_{n,q,\chi}$ states that for all polynomials $m = \mathsf{poly}(\lambda)$ the following distributions are computationally indistinguishable:*

$$(\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}) \overset{c}{\approx} (\mathbf{A}, \mathbf{u})$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \chi^m, \mathbf{u} \leftarrow \mathbb{Z}_q^m$.

Just like many prior works, we rely on LWE security with the following range of parameters. We assume that for any polynomial $p = p(\lambda) = \mathsf{poly}(\lambda)$ there exists some polynomial $n = n(\lambda) = \mathsf{poly}(\lambda)$, some $q = q(\lambda) = 2^{\mathsf{poly}(\lambda)}$ and some $B = B(\lambda)$-bounded distribution $\chi = \chi(\lambda)$ such that $q/B \geq 2^p$ and the $LWE_{n,q,\chi}$ assumption holds. Throughout the paper, the *LWE assumption* without further specification refers to the above parameters. The *sub-exponentially secure LWE* assumption further assumes that $LWE_{n,q,\chi}$ with the above parameters is sub-exponentially secure, meaning that there exists some $\varepsilon > 0$ such that the distinguishing advantage of any polynomial-time distinguisher is $2^{-\lambda^\varepsilon}$.

The works of [Reg05], [Pei09] showed that the (sub-exponentially secure) LWE assumption with the above parameters follows from the worst-case (sub-exponential) quantum hardness SIVP and classical hardness of GapSVP with sub-exponential approximation factors.

We will use the following fact.

Lemma 3 (Smudging Lemma (e.g., [AJL+12]))**.** *Let $B = B(\lambda), B' = B'(\lambda) \in \mathbb{Z}$ be parameters and and let $e_1 \in [-B, B]$ be an arbitrary value. Let $e_2 \leftarrow [B', B']$ be chosen uniformly at random. Then the distribution of e_2 is statistically indistinguishable from that of $e_2 + e_1$ as long as $B/B' = \mathsf{negl}(\lambda)$.*

We recall the definition of the Gadget Matrix introduced in [MP12]. For an integer $q \geq 2$, define: $\mathbf{g} = (1, 2, \cdot, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}_q^{1 \times \lceil \log q \rceil}$. The *Gadget Matrix* $\mathbf{G}$

is defined as $\mathbf{G} = \mathbf{g} \otimes \mathbf{I_n} \in \mathbb{Z}_q^{n \times m}$ where $n \in \mathbb{N}$ and $m = n\lceil \log q \rceil$. There exists an efficiently computable deterministic function $\mathbf{G}^{-1} : \mathbb{Z}_q^n \to \{0,1\}^m$ such for all $\mathbf{u} \in \mathbb{Z}_q^n$ we have $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{u}) = \mathbf{u}$. We let $\mathbf{G}^{-1}(\$)$ denote the distribution obtained by sampling $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ uniformly at random and outputting $\mathbf{t} = \mathbf{G}^{-1}(\mathbf{u})$.

We rely on the following algorithms introduced in [BGG+14].

Claim 4 ([BGG+14])**.** *Let $n \in \mathbb{N}$ and $m = n\lceil \log q \rceil$. There exists two deterministic algorithms* EvalPK *and* EvalCT *with the following syntax:*

- $\mathsf{EvalPK}(C, \mathbf{A}_1, \ldots, \mathbf{A}_k)$ *takes as input a circuit $C : \{0,1\}^k \to \{0,1\}$ and matrices $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$, and outputs a matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times m}$;*
- $\mathsf{EvalCT}(C, \mathbf{A}_1, \ldots, \mathbf{A}_k, \mathbf{b}_1, \ldots \mathbf{b}_k, x)$ *takes as input a circuit $C : \{0,1\}^k \to \{0,1\}$, matrices $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$, vectors $\mathbf{b}_i \in \mathbb{Z}_q^m$ and an input $x \in \{0,1\}^k$, and outputs a vector $\mathbf{b}_C \in \mathbb{Z}_q^m$;*

such that if there exists some $\mathbf{s} \in \mathbb{Z}_q^n$ such that, for all $i \le k$, $\mathbf{b}_i = \mathbf{s}^\top(\mathbf{A}_i - x_i\mathbf{G}) + \mathbf{e}_i$ with $\|\mathbf{e}_i\|_\infty \le B$, then:

$$\mathbf{b}_C = \mathbf{s}^\top(\mathbf{A}_C - C(x)\mathbf{G}) + \mathbf{e}_C \text{ where } \|\mathbf{e}_C\|_\infty \le (m+1)^d \cdot B.$$

Furthermore, the run-time of $\mathsf{EvalPK}, \mathsf{EvalCT}$ *is $|C| \cdot \mathsf{poly}(n, \log q)$.*

IV. Definition of LFE

We define our notion of laconic function evaluation (LFE) for a class of circuits $\mathcal{C}$. We assume that the class $\mathcal{C}$ associates every circuit $C \in \mathcal{C}$ with some circuit parameters $C.\mathsf{params}$. For our default notion of LFE throughout this paper, unless specified otherwise, we will consider $\mathcal{C}$ to be the class of all circuits with $C.\mathsf{params} = (1^k, 1^d)$ consisting of the input size k and the depth d of the circuit.

Definition 5 (LFE)**.** *A laconic function evaluation (LFE) scheme for a class of circuits $\mathcal{C}$ consists of four algorithms* crsGen *,* $\mathsf{Compress}$ *,* Enc *and* Dec*.*

- $\mathsf{crsGen}(1^\lambda, \mathsf{params})$ *takes as input the security parameter 1^λ and circuit parameters* params *and outputs a uniformly random* common random string crs *of appropriate length.*[5]
- $\mathsf{Compress}(\mathsf{crs}, C)$ *is a deterministic algorithm that takes as input the common random string* crs *and a circuit $C \in \mathcal{C}$ and outputs a digest* digest_C*.*
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, x)$ *takes as input the common random string* crs*, a digest* digest_C *and a message x and outputs a ciphertext* ct*.*
- $\mathsf{Dec}(\mathsf{crs}, C, \mathsf{ct})$ *takes as input the common random string* crs*, a circuit $C \in \mathcal{C}$, and a ciphertext* ct *and outputs a message y.*

We require the following properties from those algorithms:

[5]It would also make sense to allow the crs to be a common *reference* string which is not necessarily uniform. However, since our constructions will have a uniformly random crs we restrict ourselves to this requirement throughout the paper.

a) Correctness: We require that for all λ, params *and $C \in \mathcal{C}$ with $C.\mathsf{params} = \mathsf{params}$:*

$$\Pr\left[y = C(x) \;\middle|\; \begin{array}{ccc} \mathsf{crs} & \leftarrow & \mathsf{crsGen}(1^\lambda, \mathsf{params}) \\ \mathsf{digest}_C & = & \mathsf{Compress}(\mathsf{crs}, C) \\ \mathsf{ct} & \leftarrow & \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, x) \\ y & \leftarrow & \mathsf{Dec}(\mathsf{crs}, C, \mathsf{ct}) \end{array} \right] = 1.$$

b) Security: We require that there exists a PPT simulator Sim *such that for all stateful PPT adversary* Adv*, we have:*

$$\left| \Pr\left[\mathrm{EXP}_{LFE}^{Real}(1^\lambda) = 1 \right] - \Pr\left[\mathrm{EXP}_{LFE}^{Ideal}(1^\lambda) \right] \right| \le \mathsf{negl}(\lambda),$$

for the experiments $\mathrm{EXP}_{LFE}^{Real}(1^\lambda)$ and $\mathrm{EXP}_{LFE}^{Ideal}(1^\lambda)$ defined below:

$\mathrm{EXP}_{LFE}^{Real}(1^\lambda)$:

0. $\mathsf{params} \leftarrow \mathsf{Adv}(1^\lambda)$
1. $\mathsf{crs} \leftarrow \mathsf{crsGen}(1^\lambda, \mathsf{params})$
2. $x^*, C \leftarrow \mathsf{Adv}(\mathsf{crs})$*:* $C \in \mathcal{C}$, $C.\mathsf{params} = \mathsf{params}$
3. $\mathsf{digest}_C = \mathsf{Compress}(\mathsf{crs}, C)$
4. $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, x^*)$
5. *Output* $\mathsf{Adv}(\mathsf{ct})$

$\mathrm{EXP}_{LFE}^{Ideal}(1^\lambda)$:

0. $\mathsf{params} \leftarrow \mathsf{Adv}(1^\lambda)$
1. $\mathsf{crs} \leftarrow \mathsf{crsGen}(1^\lambda, \mathsf{params})$
2. $x^*, C \leftarrow \mathsf{Adv}(\mathsf{crs})$*:* $C \in \mathcal{C}$, $C.\mathsf{params} = \mathsf{params}$
3. $\mathsf{digest}_C = \mathsf{Compress}(\mathsf{crs}, C)$
4. $\mathsf{ct} \leftarrow \mathsf{Sim}(\mathsf{crs}, C, \mathsf{digest}_C, C(x^*))$
5. *Output* $\mathsf{Adv}(\mathsf{ct})$

We refer to the above as adaptive *security. We also define a weaker version of* selective *security where the above experiments are modified so that* Adv *has to choose x^* at the very beginning of the experiment in step 0, before seeing* crs *(but can still choose C adaptively).*

c) Composability: Note that, in our security definition, the simulator is given a correctly generated crs as an input rather than being able to sample it itself. This guarantees composability. Given several ciphertexts ct_i encrypting various inputs x_i under the same or different digests digest_{C_j}, all using the same crs, we can simulate all of them simultaneously and security follows via a simple hybrid argument where we switch them from real to simulated one by one.

d) Efficiency: The above definition does not directly impose any efficiency restrictions and can therefore be satisfied trivially by setting $\mathsf{digest}_C = C$ and $\mathsf{Enc}(\mathsf{digest}_C, x) = C(x)$. The main goal will be to ensure that the LFE scheme is *laconic*, meaning that the size of $\mathsf{crs}, \mathsf{digest}_C, \mathsf{ct}$ and the run-time of Enc should all be as small as possible and certainly smaller than the circuit size of C. We will discuss the efficiency of our constructions as we present them.

V. Construction of AB-LFE from LWE

We construct LFE for all circuits under the LWE assumption with subexponential modulus-to-noise ratio. As a stepping stone to build LFE, we consider LFE for a restricted class of functionalities, which we call *attribute-based LFE (AB-LFE)* in analogy to attribute-based encryption (ABE). The transformation from AB-LFE to LFE is similar to the transformation from ABE to succinct, single-key functional encryption [GKP+13] and can be found in the full version of this paper.

Let $C : \{0,1\}^k \to \{0,1\}$ be a circuit. We define the *Conditional Disclosure Functionality* (CDF) of C as the function

$$\mathsf{CDF}[C](x,\mu) = \begin{cases} (x,\mu) \text{ if } C(x) = 0\,; \\ (x,\perp) \text{ if } C(x) = 1\,, \end{cases}$$

where $x \in \{0,1\}^k$, and $\mu \in \{0,1\}$.

Definition 6 (AB-LFE)**.** *An AB-LFE for a circuit family $\mathcal{C}$ is an LFE that supports circuits $\mathsf{CDF}[C]$, for all $C \in \mathcal{C}$. We define $\mathsf{CDF}[C].\mathsf{params} = C.\mathsf{params} = (1^k, 1^d)$ where k is the input size and d is the depth of C.*

We will consider canonical descriptions of $\mathsf{CDF}[C]$ such that one can efficiently recover C given $\mathsf{CDF}[C]$. To simplify notation for AB-LFE we will give the algorithms of the AB-LFE the circuit C as input rather than $\mathsf{CDF}[C]$. E.g., we will write $\mathsf{digest}_C \leftarrow \mathsf{Compress}(\mathsf{crs}, C)$ instead of the more cumbersome $\mathsf{digest}_{\mathsf{CDF}[C]} \leftarrow \mathsf{Compress}(\mathsf{crs}, \mathsf{CDF}[C])$.

A. Basic AB-LFE from LWE

We first present our simplest construction of an AB-LFE from LWE for the case of a single bit output.

a) Parameters: We fix integer parameters $n, m, q, B, B' \in \mathbb{Z}$ and a B-bounded distribution χ over $\mathbb{Z}$, all of which are functions of λ, d. In particular, we set $m = n\lceil \log q \rceil$ and the choice of n, q, χ, B comes from the LWE assumption subject to $n = \mathsf{poly}(\lambda, d), q = 2^{\mathsf{poly}(\lambda,d)}$, and $q/B = 8\cdot(m+1)^{d+1}\cdot 2^\lambda$. We set $B' = B\cdot(m+1)^{d+1}\cdot 2^\lambda$.

b) Construction: We define the procedures crsGen , $\mathsf{Compress}$, Enc and Dec as follows.

- $\mathsf{crsGen}(1^\lambda, \mathsf{params} = (1^k, 1^d))$: Pick k random matrices $\mathbf{A}_i \leftarrow \mathbb{Z}_q^{n\times m}$ and set:
$$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_k).$$
- $\mathsf{Compress}(\mathsf{crs}, C)$: Output:
$$\mathsf{digest}_C = \mathbf{A}_C = \mathsf{EvalPK}(C, \{\mathbf{A}_i\}_{i\le k})\,,$$
where EvalPK is defined in Section III.
- $\mathsf{Enc}(\mathsf{crs}, \mathbf{A}_C, (\mathbf{x}, \mu))$: Pick a random $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}_i \leftarrow \chi^m$ for $i \le k$, and compute:
$$\mathbf{b}_i = \mathbf{s}^\top(\mathbf{A}_i - x_i\mathbf{G}) + \mathbf{e}_i^\top, \quad i \le k.$$
Sample $\mathbf{t} \leftarrow \mathbf{G}^{-1}(\$)$,[6] set $\mathbf{d} = \mathbf{A}_C \cdot \mathbf{t}$, and compute
$$\beta = \mathbf{s}^\top\mathbf{d} + \widetilde{e} + \mu\cdot\lfloor q/2\rceil,$$
where $\widetilde{e} \leftarrow [-B', B']$. Output:
$$\mathsf{ct} = (\{\mathbf{b}_i\}_{i\le k},\, \beta,\, \mathbf{t},\, x).$$
- $\mathsf{Dec}(\mathsf{crs}, C, \mathsf{ct})$: If $C(x) = 1$ output $\perp$. Else compute:
$$\widetilde{\mathbf{b}} = \mathsf{EvalCT}(C, \{\mathbf{A}_i\}_{i\le k}, \{\mathbf{b}_i\}_{i\le k}, x)\,,$$
where EvalCT is defined in Section III. Output $\lfloor(\beta - \widetilde{\mathbf{b}}^\top\mathbf{t})/q\rceil$.

[6]Recall $\mathbf{G}^{-1}(\$)$ denotes the distribution of sampling $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ uniformly at random and outputting $\mathbf{t} = \mathbf{G}^{-1}(\mathbf{u})$.

c) Proof overview: The key observation underlying both correctness and security is that for honestly generated ciphertexts, we have: $\widetilde{\mathbf{b}} \approx \mathbf{s}^\top[\mathbf{A}_C - C(x)\mathbf{G}]$ and therefore:

$$\beta - \widetilde{\mathbf{b}}\cdot\mathbf{t} \approx C(x)\cdot\mathbf{s}^\top\mathbf{G}\cdot\mathbf{t}. \tag{2}$$

In the proof, we will use the above equation to simulate β as $\widetilde{\mathbf{b}}\cdot\mathbf{t} + C(x)\cdot\mathbf{s}^\top\mathbf{G}\cdot\mathbf{t}$.

Claim 7 (Correctness)**.** *The construction is correct.*

Proof. Assume $C(x) = 0$. Then, by correctness of EvalCT, we have:

$$\begin{aligned}\widetilde{\mathbf{b}} &= \mathsf{EvalCT}(C, \{\mathbf{A}_i\}_{i\le k}, \{\mathbf{b}_i\}_{i\le k}, x) \\ &= \mathbf{s}^\top[\mathbf{A}_C - C(x)\mathbf{G}] + \mathbf{e}_C^\top = \mathbf{s}^\top\mathbf{A}_C + \mathbf{e}_C^\top,\end{aligned}$$

with $\|\mathbf{e}_C\|_\infty \le (m+1)^d\cdot B$. Therefore $\widetilde{\mathbf{b}}^\top\mathbf{t} = \mathbf{s}^\top\mathbf{d} + \mathbf{e}_C^\top\cdot\mathbf{t}$ and

$$\begin{aligned}\beta - \widetilde{\mathbf{b}}^\top\mathbf{t} &= (\mathbf{s}^\top\mathbf{d} + \widetilde{e} + \mu\cdot\lfloor q/2\rceil) - (\mathbf{s}^\top\mathbf{d} + \mathbf{e}_C^\top\cdot\mathbf{t}) \\ &= \mu\cdot\lfloor q/2\rceil + \widetilde{e} - \mathbf{e}_C^\top\cdot\mathbf{t}\,,\end{aligned}$$

where $|\widetilde{e} - \mathbf{e}_C^\top\cdot\mathbf{t}| \le (m+1)^{d+1}\cdot B + B' < 2B\cdot(m+1)^{d+1}\cdot 2^\lambda < q/4$. □

Claim 8 (Security)**.** *The construction is selectively secure under the LWE assumption $\mathrm{LWE}_{n,q,\chi}$.*

Proof. Note that the simulator is given C, x^* and $\mathsf{CDF}[C](x^*, \mu^*)$ as input. In particular, if $C(x^*) = 0$ then the simulator is given (x^*, μ^*) in full and can therefore create the ciphertext honestly as in the Real experiment. So without loss of generality, we can concentrate on the case $C(x^*) = 1$. Define the following simulator:

- $\mathsf{Sim}(\mathsf{crs}, \mathsf{digest}_C, C, x^*)$: pick $\mathbf{b}_i \leftarrow \mathbb{Z}_q^m$ for $i \le k$, pick $\beta \leftarrow \mathbb{Z}_q$, and $\mathbf{t} \leftarrow \mathbf{G}^{-1}(\$)$. Output:
$$\mathsf{ct} = (\{\mathbf{b}_i\}_{i\le k},\, \beta,\, \mathbf{t},\, x^*).$$

We prove that the experiments $\mathrm{EXP}_{LFE}^{Real}(1^\lambda)$ and $\mathrm{EXP}_{LFE}^{Ideal}(1^\lambda)$ from Definition 5 are computationally indistinguishable via a sequence of hybrids.

Hybrid 0. This is the real experiment $\mathrm{EXP}_{LFE}^{Real}(1^\lambda)$.

Hybrid 1. Here the way β is computed is modified. After computing $\mathbf{b}_i = \mathbf{s}^\top(\mathbf{A}_i - x_i^*\mathbf{G}) + \mathbf{e}_i^\top$ and sampling $\mathbf{t} \leftarrow \mathbf{G}^{-1}(\$)$, compute $\widetilde{\mathbf{b}} = \mathsf{EvalCT}(C, \{\mathbf{A}_i\}_{i\le k}, \{\mathbf{b}_i\}_{i\le k}, x^*)$ and $\alpha = \mathbf{s}^\top\mathbf{G}\mathbf{t} + e_0 \in \mathbb{Z}_q$, where $e_0 \leftarrow \chi$. Set

$$\beta = \widetilde{\mathbf{b}}\cdot\mathbf{t} + \alpha + \widetilde{e} + \mu\cdot\lfloor q/2\rceil,$$

where $\widetilde{e} \leftarrow [-B', B']$.

By correctness of EvalCT and the fact that $C(x^*) = 1$, we have $\widetilde{\mathbf{b}} = \mathbf{s}^\top(\mathbf{A}_C - \mathbf{G}) + \mathbf{e}_C^\top$, so that

$$\widetilde{\mathbf{b}}\cdot\mathbf{t} + \alpha = \mathbf{s}^\top\mathbf{A}_C\cdot\mathbf{t} + \mathbf{e}_C^\top\cdot\mathbf{t} + e_0 = \mathbf{s}^\top\mathbf{d} + \mathbf{e}_C^\top\cdot\mathbf{t} + e_0$$

and therefore, in Hybrid 1, we have

$$\beta = \mathbf{s}^\top\mathbf{d} + \mu^*\cdot\lfloor q/2\rceil + \widetilde{e} + (\mathbf{e}_C^\top\cdot\mathbf{t} + e_0).$$

Then, by Lemma 3, the distributions of $\widetilde{e} + \mathbf{e}_C^\top\mathbf{t} + e_0$ and $\widetilde{e}$ are statistically close; and therefore the distribution of β in Hybrid 1 is statistically close to that in Hybrid 0.

Hybrid 2. In this hybrid we pick α and $\{\mathbf{b}_i\}_{i\le k}$ uniformly at random. We still sample $\mathbf{t} \leftarrow \mathbf{G}^{-1}(\$)$ and compute $\widetilde{\mathbf{b}} = \mathsf{EvalCT}(C, \{\mathbf{A}_i\}_{i\le k}, \{\mathbf{b}_i\}_{i\le k}, x)$ and $\beta = \widetilde{\mathbf{b}} \cdot \mathbf{t} + \alpha + \widetilde{e} + \mu^* \cdot \lfloor q/2 \rceil$ as previously.

We show that Hybrid 1 and Hybrid 2 are computationally indistinguishable under $\mathrm{LWE}_{n,q,\chi}$. More precisely, the reduction receives x^* from the adversary and gets LWE challenges $(\mathbf{u}, \alpha)$ and $(\mathbf{M_i}, \mathbf{b}_i)_{i\le k}$, where $\mathbf{u} \leftarrow \mathbb{Z}_q^n, \mathbf{M_i} \leftarrow \mathbb{Z}_q^{n\times m}$. It sets $\mathsf{crs} = \{\mathbf{A}_i = \mathbf{M_i} + x_i^* \mathbf{G}\}_{i\le k}$, and $\mathbf{t} = \mathbf{G}^{-1}(\mathbf{u})$, and computes β as previously. Then, if α and $\{\mathbf{b}_i\}$ are distributed as LWE samples, then the view of the adversary is as in Hybrid 1; if they are uniform, then its view is distributed as in Hybrid 2.

Note that Hybrid 2 corresponds to $\mathrm{EXP}_{LFE}^{Ideal}(1^\lambda)$ with the simulator Sim defined above since the values β and $\{\mathbf{b}_i\}_{i\le k}$ in Hybrid 2 are uniformly random. □

d) Comparison with the ABE from [BGG+14]: Our construction is very reminiscent of the ABE from [BGG+14]. In their ABE scheme, there is an additional matrix $\mathbf{A}_0$, and the relation $\mathbf{d} = \mathbf{A}_C \cdot \mathbf{t}$ is replaced with $\mathbf{d} = [\mathbf{A}_0 \mid \mathbf{A}_C] \cdot \mathbf{t}$. The quantities $\mathbf{A}_0, \mathbf{d}$ are part of the master public key and $\mathbf{t}$ is the secret key corresponding to the circuit C. The ABE security proof needs to take into account an adversary that sees many such $\mathbf{t}$'s with respect to the same $\mathbf{d}$.

There are several key differences between our AB-LFE and the prior ABE scheme. A crucial distinction is that our Encryption algorithm takes $\mathbf{A}_C$ as input, which allows it to sample $\mathbf{t}$ and then set $\mathbf{d} = \mathbf{A}_C \cdot \mathbf{t}$. The value $\mathbf{t}$ is a part of our ciphertext. In [BGG+14], the analogous operation is performed by having a trapdoor for $\mathbf{A}_0$ as a master secret key of the ABE and using this trapdoor to sample $\mathbf{t}$ that satisfies $\mathbf{d} = [\mathbf{A}_0 \mid \mathbf{A}_C] \cdot \mathbf{t}$ during key generation and including it as part of the secret key for the circuit C. The proof of security for the ABE scheme is also significantly more complex and involves carefully "puncturing" the public matrices in $\mathbf{A}_i$ in the crs so as to be able to provide secret keys for circuits C if and only if $C(x) = 1$. The fact that we don't need any trapdoors for our construction essentially paves our way to a much simpler proof.

e) Message-adaptivity: To argue indistinguishability between Hybrids 1 and 2 from LWE, the reduction needs to know the challenge attribute x^* ahead of time to create the crs using its LWE challenges, which makes the construction selectively secure. However, it doesn't need to know the message μ^* at that point. In particular, our construction is "*message-adaptive*", where the adversary has to choose the challenge attribute x^* before seeing the crs, but can choose the challenge message μ^* *after* seeing the crs.

f) Efficiency: For circuits with input length k, 1-bit output and depth d the above construction of AB-LFE from LWE has the following parameters.

- The crs is of size $k \cdot \mathsf{poly}(\lambda, d)$. The digest is of size $\mathsf{poly}(\lambda, d)$.
- The run-time of the encryption algorithm and the size of the ciphertext are $k \cdot \mathsf{poly}(\lambda, d)$.
- The run-time of the compression and the decryption algorithms is $|C| \cdot \mathsf{poly}(\lambda, d)$.

We refer to the full version for more optimized constructions when considering multi-bit output.

B. Adaptive AB-LFE Security from Adaptive LWE

We only proved the selective security of the AB-LFE above under LWE. We now show how to prove adaptive security by relying on a new but natural LWE assumption where we give the adversary some very limited choice over the coefficients used to create LWE samples. In more detail, the adversary is given matrices $\mathbf{A}_1, \ldots, \mathbf{A}_k$, adaptively chooses $x \in \{0,1\}^k$ and gets LWE samples with the coefficients $\mathbf{A}_i - x_i \mathbf{G}$. We also give the adversary arbitrarily many additional LWE samples with random coefficients.

Definition 9 ((Decision) Adaptive LWE)**.** *We define the decision adaptive LWE assumption $ALWE_{n,k,q,\chi}$ with parameter $n, k, q \in \mathbb{Z}$ and a distribution χ over $\mathbb{Z}$ which are all parametrized by the security parameter λ. Let $m = n\lceil \log q \rceil$. We let $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ be the gadget matrix (see Section III).*[7] *For any polynomial $m' = m'(\lambda)$ we consider the following two games $\mathbf{Game}^\beta$ with $\beta \in \{0,1\}$ between a challenger and an adversary $\mathcal{A}$.*

- *The Challenger picks k random matrices $\mathbf{A}_i \leftarrow \mathbb{Z}_q^{n\times m}$ for $i \le k$, and sends them to $\mathcal{A}$.*
- *$\mathcal{A}$ adaptively picks $x \in \{0,1\}^k$, and sends it to the Challenger.*
- *The Challenger samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. For all $i \le k$, it picks $\mathbf{e}_i \leftarrow \chi^m$ and computes :*
$$\begin{cases} \mathbf{b}_i = \mathbf{s}^\top (\mathbf{A}_i - x_i \cdot \mathbf{G}) + \mathbf{e}_i^\top & \text{if } \beta = 0, \\ \mathbf{b}_i \leftarrow \mathbb{Z}_q^m & \text{if } \beta = 1. \end{cases}$$
It also picks $\mathbf{A}_{k+1} \leftarrow \mathbb{Z}_q^{n\times m'}$, $\mathbf{e}_{k+1} \leftarrow \chi^{m'}$ and computes
$$\begin{cases} \mathbf{b}_{k+1} = \mathbf{s}^\top \mathbf{A}_{k+1} + \mathbf{e}_{k+1}^\top & \text{if } \beta = 0, \\ \mathbf{b}_{k+1} \leftarrow \mathbb{Z}_q^{m'} & \text{if } \beta = 1. \end{cases}$$
It sends $\mathbf{A}_{k+1}$, $\{\mathbf{b}_i\}_{i\le k+1}$ to $\mathcal{A}$.

The $ALWE_{n,k,q,\chi}$ assumption states that for all polynomial $m = m(\lambda)$ the games $\mathbf{Game}^0$ and $\mathbf{Game}^1$ are computationally indistinguishable.

a) Adaptive LWE implies Adaptive AB-LFE: We notice that the construction of AB-LFE above is adaptively secure under the adaptive $ALWE_{n,k,q,\chi}$ assumption where k is the input size of the circuit. This follows directly from the proofs of indistinguishability - in particular, the indistinguishability of Hybrids 1 and 2 which relied on the LWE assumption. (The additional LWE challenge values $\mathbf{A}_{k+1}, \mathbf{b}_{k+1}$ with non-adaptively selected coefficients are used to create $(\mathbf{u}, \alpha)$ needed to generate the ciphertext.)

[7] The assumption would be meaningful for any other choice of matrix $\mathbf{G}$ as well.

b) Security of Adaptive LWE: Note that if the adversary was forced to choose x before seeing $\{\mathbf{A}_i\}$ then the security of ALWE would follow directly from LWE. We also have a reduction from $LWE_{n,q,\chi}$ to $ALWE_{n,k,q,\chi}$ with exponential security loss 2^k, where the reduction *guesses* in advance the adaptive choice x that the adversary makes during the interaction with the challenger. Therefore, under the sub-exponential security of LWE we can show the security of ALWE but all of the parameters (lattice dimension, modulus size etc.) have to scale polynomially with k. Nevertheless, it seems plausible to assume that ALWE has a much higher level of security than this reduction implies and that the parameters do not have to scale polynomially with k. We summarize the above by outlining two possible parameter settings for ALWE.

- *Optimistic Parameters:* For any polynomial $p = p(\lambda)$ there exists some polynomial $n = n(\lambda)$, some $q = q(\lambda) = 2^{\mathsf{poly}(\lambda)}$ and some $B = B(\lambda)$-bounded distribution $\chi = \chi(\lambda)$ such that $q/B \geq 2^p$ and the $ALWE_{n,k,q,\chi}$ assumption holds for all polynomial $k = k(\lambda)$.
- *Provable Parameters:* For any polynomials $p = p(\lambda), k = k(\lambda)$ there exists some polynomial $n = n(\lambda)$, some $q = q(\lambda) = 2^{\mathsf{poly}(\lambda)}$ and some $B = B(\lambda)$-bounded distribution $\chi = \chi(\lambda)$ such that $q/B \geq 2^p$ and the $LWE_{n,k,q,\chi}$ assumption holds.

Note the difference – with the provable parameters, the choice of n, q, χ can depend on k while with the optimistic parameters they do not. The ALWE assumption with provable parameters follows from the sub-exponential security of LWE. The ALWE assumption with optimistic parameters is plausibly secure but we do not have any meaningful reduction from LWE.

VI. MPC WITH SMALL ONLINE COMPUTATION

As an application of our LFE, we construct an MPC protocol where N parties with respective inputs $x_1, \ldots, x_N$ can securely evaluate some function $y = f(x_1, \ldots, x_N)$ over their inputs so that every party learns y. The protocol consists of three phases. There is a *pre-processing phase* in which each party does some local deterministic computation over the circuit f, but independent of the party's input. This phase can be reused across many protocol executions. Then there is an *online* phase in which the parties communicate with each other and execute some protocol. Lastly, there is a *post-processing phase* in which each party does some local computation over the protocol transcript to recover the output y.

Let LFE = (crsGen , Compress , Enc, Dec) be an LFE scheme as in Definition 5 (with a deterministic compression function). Let ψ be a generic MPC protocol without any special efficiency requirements. For a function $f : (\{0,1\}^k)^N \to \{0,1\}^\ell$, we define the following two related protocols π_f^{SH} (a semi-honest secure protocol in the plain model) and π_f^M (a malicious secure protocol in the CRS model):

a) The Protocol π_f^M: A malicious secure protocol in the CRS model.

- *CRS Generation:* Sample $\mathsf{crs} \leftarrow \mathsf{LFE.crsGen}(1^\lambda, f.\mathsf{params})$.
- *Pre-Processing:* Each party computes $\mathsf{digest}_f = \mathsf{LFE.Compress}(\mathsf{crs}, f)$.
- *Online Phase:* The parties run a generic MPC protocol ψ for the function

$$\mathsf{ct} = \mathsf{LFE.Enc}(\mathsf{crs}, \mathsf{digest}_f, (x_1, \ldots, x_N)\,; \bigoplus_{i \in [N]} r_i).$$

  We think of $\mathsf{crs}, \mathsf{digest}_f$ as part of the public function description. Each party i has input (x_i, r_i), where r_i is chosen uniformly at random, and gets ct as output.
- *Post-Processing:* Each party computes $y = \mathsf{LFE.Dec}(\mathsf{crs}, f, \mathsf{ct})$.

b) The Protocol π_f^{SH}: A a semi-honest secure protocol in the plain model.

- *Pre-Processing:* Party 1 samples $\mathsf{crs} \leftarrow \mathsf{LFE.crsGen}(1^\lambda, f.\mathsf{params})$ and computes $\mathsf{digest}_f = \mathsf{LFE.Compress}(\mathsf{crs}, f)$.
- *Online Phase:* The parties run a generic MPC protocol ψ for the function

$$\mathsf{ct} = \mathsf{LFE.Enc}(\mathsf{crs}, \mathsf{digest}_f, (x_1, \ldots, x_N)\,; \bigoplus_{i \in [N]} r_i).$$

  The input of party 1 consists of $(\mathsf{crs}, \mathsf{digest}_f, x_1, r_1)$ and the input of each party $i > 1$ consists of (x_i, r_i). The values r_i are chosen uniformly at random. Each party get crs, ct as output.
- *Post-Processing:* Each party computes $y = \mathsf{LFE.Dec}(\mathsf{crs}, f, \mathsf{ct})$.

Theorem 10. *Assuming that the LFE scheme is selectively secure and that ψ is a semi-honest secure MPC, the protocol π_f^{SH} is a semi-honest secure MPC for f in the plain model. Assuming that the LFE is adaptively secure and that ψ is a malicious secure MPC, the protocol π_f^M is a malicious secure MPC for f in the CRS model. In both cases we assume static corruptions.*

We defer the proof of the theorem to the full version of the paper [QWW18].

c) Instantiation and Parameters: We can instantiate the protocol ψ with (e.g.,) the [GMW86] protocol, which provides malicious security, using oblivious transfer based on LWE. By plugging in the parameters for LFE based on LWE described in the full version, we get the following parameters. For a circuit $f : (\{0,1\}^k)^N \to \{0,1\}^\ell$ with depth d we get a semi-honest MPC protocol in the plain model and a malicious-secure MPC in the CRS model with the following efficiency:

- In the CRS model, the size of the CRS is $k \cdot N \cdot \mathsf{poly}(\lambda, d)$.
- The communication complexity is $(k + \ell) \cdot \mathsf{poly}(\lambda, d, N)$.

- Each party's computation in the online phase is $(k + \ell) \cdot \mathsf{poly}(\lambda, d, N)$.
- Each party's computation in the pre-processing/post-processing phase is $|f| \cdot \mathsf{poly}(\lambda, d)$.

Semi-honest security holds under the LWE assumption. Malicious security in the CRS model holds under the adaptive LWE (ALWE) assumption with optimistic parameters. Under the ALWE assumption with provable parameters, which follows from the sub-exponential security of LWE, we get a malicious secure protocol in the CRS model with decreased efficiency where all of the $\mathsf{poly}(\lambda, d)$ and $\mathsf{poly}(\lambda, d, N)$ factors become $\mathsf{poly}(\lambda, d, k, N)$.

d) Improving 2-round 2PC: Alternatively, we can instantiate the protocol ψ with a 2-round semi-honest protocol in the plain model such as the one in [BL18], [GS18] based only on OT which follows from LWE. In that case, we get semi-honest MPC with the same efficiency as above but with only 2 rounds of interaction in the plain model. Note that the total computational complexity of each party, including pre-processing and post-processing, is $|f| \cdot \mathsf{poly}(\lambda, d) + (k+\ell) \cdot \mathsf{poly}(\lambda, d, N)$. Despite many works on 2-round 2-PC [MW16], [PS16], [BP16], [GS17], [GS18], [BL18], in all prior constructions the per-party computation is at least $|f| \cdot N^2$ even in the semi-honest setting and even in the CRS model. Therefore the above gives an asymptotic improvement on the computation complexity of 2-round 2-PC even if we do not distinguish between online and offline computation.

e) Necessity of Pre-Processing and Post-Processing: We argue that both the pre-processing and post-processing steps are necessary, at least in the circuit model where the function f is given to the parties as a circuit.

Assume that we were able to remove the pre-processing step. Since the online computation is small, most bits of the description of f were never accessed by any party during the protocol until the post-processing step. But that means that, during the post-processing step, party 1 can modify a random bit of f (in its head) to get f' and with good probability is able to learn $f'(x_1, \ldots, x_N)$. This violates security. (Note that this argument relies on f having a large description. It remains an interesting problem if we can get rid of pre-processing if f is a Turing Machine with short description.)

Assume we were able to remove the post-processing step. This means that, even if we didn't care about security, we now have a method to pre-process an arbitrary circuit f and then evaluate $f(x)$ on arbitrary inputs x at a lower cost than computing f. But that means that there is an altogether smaller circuit g that computes f. Since not all circuits are compressible [Sha49] this cannot be done in general.

ACKNOWLEDGMENTS

We thank Yuval Ishai for insightful discussions on succinct secure computation. We thank Alex Lombardi for pointing out to us that LFE implies succinct FE.

The second author is supported by ERC Project aSCEND (H2020 639554). The third author is supported by NSF grants CNS-1314722, CNS-1413964 and the Alfred P. Sloan Research Fellowship.

REFERENCES

[ABJ+18] Prabhanjan Ananth, Saikrishna Badrinarayanan, Aayush Jain, Nathan Manohar, and Amit Sahai. From fe combiners to secure mpc and back. Cryptology ePrint Archive, Report 2018/457, 2018. https://eprint.iacr.org/2018/457.

[Agr17] Shweta Agrawal. Stronger security for reusable garbled circuits, general definitions and attacks. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 3–35. Springer, Heidelberg, August 2017.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501. Springer, Heidelberg, April 2012.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 111–120. ACM Press, June 2013.

[BCI+13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 315–333. Springer, Heidelberg, March 2013.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556. Springer, Heidelberg, May 2014.

[BL18] Fabrice Benhamouda and Huijia Lin. k-round MPC from k-round OT via garbled interactive circuits. In *EUROCRYPT*, 2018.

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous ibe, leakage resilience and circular security from new assumption. In *EUROCRYPT*, 2018.

[BP16] Zvika Brakerski and Renen Perlman. Lattice-based fully dynamic multi-key FHE with short ciphertexts. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 190–213. Springer, Heidelberg, August 2016.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, Heidelberg, March 2011.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 97–106. IEEE Computer Society Press, October 2011.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In Moni Naor, editor, *ITCS 2014: 5th Innovations in Theoretical Computer Science*, pages 1–12. Association for Computing Machinery, January 2014.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, Heidelberg, February 2007.

[CDG+17] Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65. Springer, Heidelberg, August 2017.

[DG17] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569. Springer, Heidelberg, August 2017.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178. ACM Press, May / June 2009.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, Heidelberg, May 2013.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 555–564. ACM Press, June 2013.

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 174–187. IEEE Computer Society Press, October 1986.

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, Heidelberg, December 2010.

[GS17] Sanjam Garg and Akshayaram Srinivasan. Garbled protocols and two-round MPC from bilinear maps. In *58th Annual Symposium on Foundations of Computer Science*, pages 588–599. IEEE Computer Society Press, 2017.

[GS18] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In *EUROCRYPT*, 2018.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, Heidelberg, August 2013.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 162–179. Springer, Heidelberg, August 2012.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 545–554. ACM Press, June 2013.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 503–523. Springer, Heidelberg, August 2015.

[HW15] Pavel Hubacek and Daniel Wichs. On the communication complexity of secure function evaluation with long output. In Tim Roughgarden, editor, *ITCS 2015: 6th Innovations in Theoretical Computer Science*, pages 163–172. Association for Computing Machinery, January 2015.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, Heidelberg, April 2008.

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th Annual Symposium on Foundations of Computer Science*, pages 436–453. IEEE Computer Society Press, November 1994.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, Heidelberg, April 2012.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 735–763. Springer, Heidelberg, May 2016.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 333–342. ACM Press, May / June 2009.

[PS16] Chris Peikert and Sina Shiehian. Multi-key FHE from LWE, revisited. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 217–238. Springer, Heidelberg, October / November 2016.

[QWW18] Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. Cryptology ePrint Archive, Report 2018/409, 2018. https://eprint.iacr.org/2018/409.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93. ACM Press, May 2005.

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10: 17th Conference on Computer and Communications Security*, pages 463–472. ACM Press, October 2010.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, Heidelberg, May 2005.