From FE Combiners to Secure MPC and Back

Prabhanjan Ananth[1], Saikrishna Badrinarayanan[2], Aayush Jain[2], Nathan Manohar[2], and Amit Sahai[2]

[1] UCSB
prabhanjan@cs.ucsb.edu
[2] UCLA
{saikrishna, aayushjain, nmanohar, sahai}@cs.ucla.edu

Abstract. Cryptographic combiners allow one to combine many candidates for a cryptographic primitive, possibly based on different computational assumptions, into another candidate with the guarantee that the resulting candidate is secure as long as at least one of the original candidates is secure. While the original motivation of cryptographic combiners was to reduce trust on existing candidates, in this work, we study a rather surprising implication of combiners to constructing secure multiparty computation protocols. Specifically, we initiate the study of functional encryption combiners and show its connection to secure multiparty computation.

Functional encryption (FE) has incredible applications towards computing on encrypted data. However, constructing the most general form of this primitive has remained elusive. Although some candidate constructions exist, they rely on nonstandard assumptions, and thus, their security has been questioned. An FE combiner attempts to make use of these candidates while minimizing the trust placed on any individual FE candidate. Informally, an FE combiner takes in a set of FE candidates and outputs a secure FE scheme if at least one of the candidates is secure.

Another fundamental area in cryptography is secure multi-party computation (MPC), which has been extensively studied for several decades. In this work, we initiate a formal study of the relationship between functional encryption (FE) combiners and secure multi-party computation (MPC). In particular, we show implications in both directions between these primitives. As a consequence of these implications, we obtain the following main results.

- A two-round semi-honest MPC protocol in the plain model secure against up to $n-1$ corruptions with communication complexity proportional only to the depth of the circuit being computed assuming learning with errors (LWE). Prior two round protocols based on standard assumptions that achieved this communication complexity required trust assumptions, namely, a common reference string.

- A functional encryption combiner based on pseudorandom generators (PRGs) in NC^1. This is a weak assumption as such PRGs are implied by many concrete intractability problems commonly used in cryptography, such as ones related to factoring, discrete logarithm, and lattice problems [11]. Previous constructions of FE combiners,

implicit in [7], were known only from LWE. Using this result, we build a universal construction of functional encryption: an explicit construction of functional encryption based only on the assumptions that functional encryption exists and PRGs in NC^1.

Keywords: Functional encryption · Cryptographic combiners · Multi-party computation.

1 Introduction

The foundations of several cryptographic primitives rely upon computational assumptions. The last few decades have seen the birth of many assumptions, such as factoring, quadratic residuosity, decisional Diffie-Hellman, learning with errors, and many more. Understanding the security of these assumptions is still very much an active research area. Despite years of research, very little is known in terms of how different cryptographic assumptions compare with each other. For instance, its not known whether decisional Diffie-Hellman is a weaker or a stronger assumption than learning with errors. This leads us to the following unsatisfactory scenario: suppose a cryptographic primitive (say, public key encryption) has many candidate constructions based on different assumptions, and we want to pick the most secure candidate. In this scenario, it is unclear which one we should pick.

Cryptographic Combiners. The notion of cryptographic combiners was introduced to resolve this dilemma. Given many candidates of a cryptographic primitive, possibly based on different assumptions, a cryptographic combiner churns these candidates into another candidate construction for the same primitive with the guarantee that the resulting construction is secure as long as at least *one* of the original candidates are secure. For instance, a combiner for public key encryption can be used to transform two candidates based on decisional Diffie-Hellman and learning with errors into a different public-key encryption candidate that is secure as long as *either* decisional Diffie-Hellman *or* learning with errors is secure.

While combiners were originally introduced to reduce trust on existing cryptographic constructions, in this work, we study a rather surprising implication from combiners to secure multi-party computation. Secure multi-party computation [79,50,19], one of the fundamental notions in cryptography, allows many parties, who don't necessarily trust each other, to come together and compute a function on their private inputs. We consider the primitive of functional encryption and study the implications of functional encryption combiners to secure multi-party computation. But first, we recall the notion of functional encryption.

Functional Encryption. Functional encryption (FE), introduced by [78,28,73], is one of the core primitives in the area of computing on encrypted data. This notion allows an authority to generate and distribute constrained keys associated with functions $f_1, \ldots, f_q$, called *functional keys*, which can be used to

learn the values $f_1(x), \ldots, f_q(x)$ given an encryption of x. Intuitively, the security notion states that the functional keys associated with $f_1, \ldots, f_q$ and an encryption of x reveal nothing beyond the values $f_1(x), \ldots, f_q(x)$. While this notion is interesting on its own, several works have studied its connections to other areas in cryptography and beyond, including reusable garbled circuits [51], indistinguishability obfuscation [8,23,9,69,68], adaptive garbling [57], verifiable random functions [54,21,14], deniable encryption [52], hardness o1f Nash equilibrium [45,46], and many more.

Currently, we know how to construct only restricted versions[3] of functional encryption from well studied cryptographic assumptions. However, constructing the most general form of functional encryption has been an active research area and has intensified over the past few years given its implication to indistinguishability obfuscation [8,23]. In fact, if we are willing to tolerate a subexponential security loss, then even secret-key FE is enough to imply indistinguishability obfuscation [22,63,62]. All the candidates [43,65,71,10,66,70] we know so far are either based on assumptions pertaining to the tool of graded encodings [29,42], or on other new and relatively unstudied assumptions [5,67,1]. Recent cryptanalytic attacks [35,60,39,36,38] on assumptions related to graded encodings have prompted scrutiny of the security of schemes that use this tool as the building block. Given this, we should hope to minimize the trust we place on any individual FE candidate. The notion of a functional encryption combiner achieves this purpose. Roughly speaking, a functional encryption combiner allows for combining many functional encryption candidates in such a way that the resulting FE candidate is secure as long as any one of the initial FE candidates is secure. In other words, a functional encryption combiner says that it suffices to place trust collectively on multiple FE candidates, instead of placing trust on any specific FE candidate.

Our Work. We initiate a systematic study of functional encryption combiners. In particular, we study implications from FE combiners to secure multi-party computation (and vice versa), and by doing so, we achieve interesting consequences that were previously unknown. We detail our contributions next.

1.1 Our Contributions

Our results can be classified into two parts. The first part shows how to translate constructions of functional encryption combiners into secure MPC protocols. The second part studies the other direction.

From combiners for single-key FE to secure MPC: Our first result shows how to construct a passively secure multi-party computation protocol that is both

[3] For instance, we can restrict the adversary to only ask for one functional key in the security experiment. A functional encryption scheme satisfying this property can be based on public key encryption schemes [77,53] (or one-way functions if one can settle for the secret key version).

round-optimal (two rounds) and communication efficient (depends only on circuit depth). Recall that in a passively secure MPC protocol, corrupted parties follow the instructions of the protocol, but try to learn about honest party inputs from their combined view of the protocol execution. Moreover, our resulting protocol is in the plain model and can tolerate all but one corruption[4]. Prior round-optimal passively-secure MPC protocols were either communication inefficient, that is communication complexity was proportional to circuit size [47,30,20,48], based on strong assumptions such as indistinguishability obfuscation [40] or were based on trust assumptions [37,72,74,33,32] (for instance, a common reference string). Subsequent to our work, [76] (FOCS'18) matched our result under the same assumption; they use the new primitive of laconic function evaluation to achieve their result instead of FE combiners. We prove the following theorem.

Theorem 1 (Informal). *Consider an n-party functionality f computable by a poly-sized circuit of depth d, for any $n \geq 2$. Assuming LWE, there is a construction of a passively secure (semi-honest) n-party computation protocol for f in the plain model secure against $n-1$ corruptions. The number of rounds in this protocol is 2, and the communication complexity is $\mathsf{poly}(\lambda, n, d, L_{in}, L_{out})$, where L_{in} is the input length of this circuit computing f, L_{out} is its output length and λ is the security parameter.*

We summarize the state of art in the Table 1.1.

	Communication Complexity	Assumptions	Model
[37,72,74,33]	$\mathsf{poly}(\lambda, n, d, L_{in}, L_{out})$	LWE	CRS
[40]	$\mathsf{poly}(\lambda, n, d, L_{in}, L_{out})$	piO and lossy encryption	Plain
[47]	$\mathsf{poly}(\lambda, n, \lvert f \rvert)$	Bilinear maps	Plain
[20,48]	$\mathsf{poly}(\lambda, n, \lvert f \rvert)$	Two-round OT	Plain
Our Work, [76]	$\mathsf{poly}(\lambda, n, d, L_{in}, L_{out})$	LWE	Plain

Fig. 1. State of the art in terms of communication complexity of two-round passively secure n-party protocols in the all-but-one corruption model. We denote by $|f|$ and d the size and depth of the circuit representing the MPC functionality f, respectively. Moreover, L_{in} and L_{out}, respectively, denote the input and output lengths of the circuit. CRS stands for common reference string and piO stands for probabilistic indistinguishability obfuscation [34].

Central to proving the above theorem is a transformation from a functional encryption combiner to passively secure MPC. We only require a combiner for

[4] Unless otherwise specified, we only consider MPC protocols tolerating all but one corruption.

functional encryption schemes where the adversary only receives one functional key. We require the functional encryption combiner to have some structural properties. Namely, the functional key for f associated with the combined candidate needs to be of the form $(f, sk_f^1, \ldots, sk_f^n)$, where (i) decomposability: sk_f^i is produced by the i^{th} FE candidate and, (ii) succinctness: the length of sk_f^i is $\mathsf{poly}(\lambda, d, L_{out})$, where d is the depth of the circuit computing f and L_{out} is its output length. As part of the succinctness property, we also require that the encryption complexity is $\mathsf{poly}(\lambda, d, L_{in})$, where L_{in} is the length of the message to be encrypted. We show how to construct such an FE combiner assuming LWE.

An intermediate tool we use in this implication is a communication **in**efficient passively secure MPC protocol. By communication inefficient, we mean that the communication complexity is proportional to the size of the circuit representing f. We note that such protocols [47,20,48] exist in the literature[5] based on just the assumption of round-optimal passively secure oblivious transfer.

Lemma 1 (Informal). *Consider a n-party functionality f, for any $n \geq 2$. There is a passively secure n-party computation protocol for f in two rounds with communication complexity $\mathsf{poly}(\lambda, n, d, L_{in}, L_{out})$ secure against $n-1$ corruptions, where d is the depth of circuit computing f, L_{in} is the input length of the circuit and L_{out} is its output length. Moreover, we assume (i) a decomposable and succinct functional encryption combiner and (ii) a communication inefficient (as defined above) two-round secure n-party computation protocol secure against $n-1$ corruptions.*

By plugging in the recent round-optimal secure MPC protocols [47,20,48] that can be based on two-round oblivious transfer, which in turn can be based on learning with errors [75], and our new decomposable and succinct FE combiner from LWE, we get Theorem 1. We note that MPC with malicious security requires at least 4 rounds [44,4,32,15,55], and thus, we do not consider MPC with malicious security in this work.

From secure MPC to combiners for unbounded-key FE: In the other direction, we show how to transform existing secure multi-party computation protocols into constructions of functional encryption combiners. However, we note that the FE combiners we construct from MPC here do not satisfy decomposability or succinctness. In particular, we show how to transform specific constant round passively secure MPC protocols based on low degree randomized encodings [17] into functional encryption combiners. By instantiating low degree randomized encodings from pseudorandom generators in NC^1, we get the following result.

[5] These protocols are inherently communication inefficient. The reason is that they present a compiler that turns any arbitrary interactive MPC protocol into a two-round MPC protocol. The communication complexity in the resulting two-round MPC protocol is at least the computational complexity of the original MPC protocol. However, the computational complexity of the resulting protocol has to be proportional to the size of the circuit representing the functionality f.

Theorem 2 (Informal). *Assuming pseudorandom generators in* NC^1*, there is a construction of a combiner for unbounded-key functional encryption.*

By unbounded-key functional encryption, we mean that there is no a priori bound on the number of functional keys the adversary can request in the security experiment. We note that such pseudorandom generators in NC^1 are implied by most concrete intractability assumptions commonly used in cryptography, such as ones related to factoring, discrete logarithm, and lattice problems [11]. Furthermore, such PRGs are also implied by the existence of one-way permutations in NC^1 or one-way functions in NC^1 with efficiently computable entropy [11].

Next, we present a generic reduction that can transform two-round passively secure MPC protocols into functional encryption combiners. For this transformation to hold, the MPC protocol must satisfy two properties: (i) delayed function-dependence: the first round of the MPC protocol should be independent of the functionality being securely computed and (ii) reusability: the first round can be reused by the parties to securely compute many functionalities (but on the same inputs fixed by the first round).

Theorem 3 (Informal). *Assuming a delayed function-dependent and reusable round-optimal secure MPC protocol, there is a construction of an unbounded-key decomposable functional encryption combiner.*

We then observe that existing two-round secure MPC protocols [72,74,33], based on learning with errors, already satisfy delayed function-dependence and reusability. We note that it is not necessary for the round-optimal protocols to be in the plain model (indeed, the protocols [72,74,33] are in the common reference string (CRS) model).

Prior to this work, the only polynomial hardness assumption known to imply an FE combiner was the learning with errors assumption [7][6]. While Theorem 2 already gives a construction of a functional encryption combiner from learning with errors (pseudorandom generators in NC^1 can be based on learning with errors [16]), the functional encryption combiner constructed in Theorem 3 arguably provides a more efficient transformation. In particular, the efficiency of the functional keys in the combined scheme from Theorem 3 is linear in the efficiency of the functional keys in the FE candidates. However, the efficiency in the combined scheme from Theorem 2 degrades polynomially in the efficiency of the original FE candidates. Furthermore, the FE combiner from Theorem 3 is *decomposable*, a property needed by an FE combiner as a building block in the proof of Theorem 1. On the other hand, the FE combiner from Theorem 2 is inherently not decomposable, since it is based on an "onion-layered" approach – this means that the keys generated with respect to one FE candidate make oracle calls to other FE candidates (see [56] for a related discussion on black-box combiners). Furthermore, the FE combiner from Theorem 3 makes only black-box use of the underlying FE candidates, whereas the FE combiner from Theorem 2 is inherently non-black-box.

[6] Note that [7] required sub-exponential hardness only for constructing iO combiners, not for constructing FE combiners.

In terms of techniques, we introduce mechanisms to emulate a MPC protocol using functional encryption candidates. This is remiscient of "MPC-in-the-head" paradigm introduced by Ishai et al. [61] and more relevant to the context of FE is the work of Gorbunov et al. [53] who used information-theoretic MPC protocols to construct single-key FE. However, we encounter new challenges to implement the "MPC-in-the-head" paradigm in our context.

Universal Functional Encryption: We strengthen our constructions of FE combiners by showing how to transform them into combiners that also work when the insecure candidates don't necessarily satisfy correctness (of course, we still require that the secure candidate is correct). Such combiners are called *robust combiners*. To do this, we present correctness amplification theorems based on previous works on indistinguishability obfuscation [24,7] and, in particular, our correctness amplification assumes only one-way functions (unlike [24,25]). Robust combiners have been useful in universal constructions [6,7]. Roughly speaking, a universal construction of FE is a concrete construction of FE that is secure as long as any secure and correct construction exists. We show how to build universal functional encryption from robust FE combiners.

Theorem 4 (Universal Functional Encryption). *Assuming pseudorandom generators in* NC^1*, there is a universal unbounded-key functional encryption scheme.*

Our construction will be parameterized by T, where T is an upper bound on the running time of all the algorithms associated with the secure candidate. This was a feature even in the universal iO construction of [6].

Related Work: The notion of combiners has been studied in the context of many cryptographic primitives. Asmuth and Blakely [13] studied combiners for encryption schemes. Levin proposed a universal construction of one-way functions [64]. Later, a systematic study of combiners and their relation to universal constructions was proposed by Harnik et al. [56] (also relevant are the constructions in [58,59]). Recently, Ananth et al. [6] designed universal constructions of indistinguishability obfuscation (iO). Concurrently, Fischlin et al. also proposed combiners in the context of program obfuscation [41]. Ananth et al. [7] then proposed the concept of transforming combiners that transforms many candidates of a primitive X, with at least one of them being secure and, into a secure candidate of primitive Y. In particular, they construct iO-to-functional encryption transforming combiners.

Subsequent to our work, [76] (FOCS'18) matched our result by also achieving a two-round semi-honest MPC protocol in the plain model with depth-proportional communication complexity assuming LWE, using laconic function evaluation instead of FE combiners. [76]'s protocol consists of pre-processing, online, and post-processing phases. Additionally, they note that the computation complexity of the online phase is also independent of the size of the function

being computed. After seeing their work, we observe that our protocol also satisfies this property. In particular, in the construction in Section 5, steps $1-3$ in round 1 can be made the preprocessing phase. The resulting protocol will now have online computation complexity independent of the size of the function being computed.

1.2 Technical Overview

We begin by tackling the problem of constructing secure multi-party computation with depth-proportional communication complexity, i.e, proportional only to the depth of the circuit being securely computed, starting from a functional encryption combiner.

Round-Optimal MPC with Depth-Proportional Communication: Let's start by recalling prior known two-round secure MPC protocols [72,74,33] with depth-proportional communication in the CRS model. The basic template is as follows: in the first round, the i^{th} party broadcasts an encryption of its input x_i. These ciphertexts are computed with respect to public keys that are derived from the CRS. All the n parties then homomorphically compute on the encryptions of $(x_1, \ldots, x_n)$ to obtain a ciphertext of $f(x_1, \ldots, x_n)$, where f is the function they wish to securely compute. The resulting ciphertext is then partially decrypted, and every party broadcasts its partially decrypted value in the second round. These values can be combined to recover the output of the functionality.

One could imagine getting rid of the CRS in the above protocol using the recent round-optimal MPC protocols in the plain model [47,20]. If this were possible, then it would yield a round-optimal MPC in the plain model that has depth-proportional communication complexity. However, the issue is that the messages in the first round of [72,74,33] are computed as functions of the CRS and thus, such an approach would inherently require three rounds.

To overcome this, we introduce a mechanism to parallelize the evaluation and the encryption processes. The output of the evaluation in our approach is the output of the functionality and not a partially decrypted value, as was the case in [72,74,33], and thus, we save one round. To implement this high level idea, we use a functional encryption combiner. Before we describe the high level template, we require that the underlying functional encryption combiner satisfies the *decomposability property*: Suppose we have FE candidates $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$. Then, a functional key for a circuit C in the combined scheme is just a concatenation of the functional keys for C, $(sk_1^C, \ldots, sk_n^C)$, where sk_i^C is computed with respect to the i^{th} FE candidate.

The template of our depth-proportional communication secure MPC construction from an FE combiner satisfying this decomposability property is in Figure 1.2. As an intermediate tool, we use a size-proportional communication secure MPC protocol (henceforth, also referred to as a communication inefficient protocol). By this, we mean that the communication complexity of the secure

MPC protocol grows polynomially with the size of the circuit being securely computed.

Our Approach

Goal: t-round depth-proportional communication secure MPC from t-round size-proportional communication secure MPC using decomposable FE combiners.

- Suppose the input of the i^{th} party is x_i and f is the function to be securely computed. All the parties execute the t-round (communication-inefficient) MPC protocol to obtain an encryption of $(x_1, \ldots, x_n)$ with respect to the combined FE scheme.
- Simultaneously, the i^{th} party computes the functional key of f with respect to the i^{th} candidate and sends it to everyone.

Fig. 2. Our approach to construct round-optimal depth-proportional communication secure MPC from decomposable functional encryption combiners.

At the end of second round, every party has an encryption of $(x_1, \ldots, x_n)$ with respect to the combined candidate and functional keys for f with respect to every candidate. From the decomposability property, this is equivalent to generating a functional key for f with respect to the combined candidate. Each party can separately execute the FE combiner decryption algorithm to obtain $f(x_1, \ldots, x_n)$, as desired. Here, we crucially rely on the fact that all the FE candidates are correct. This completes the high level description of the template.

In the first bullet in Figure 1.2, we instantiate the secure MPC protocol with size-proportional communication with [47,20,48]. The works of [47,20,48] are two-round protocols in the plain model and by suitably instantiating the FE combiners (described later), our approach yields a two-round MPC protocol with depth-proportional communication.

To argue security of our MPC protocol, the idea is to start with the assumption of a secure FE scheme and instantiate all the candidates using the same FE scheme. If the adversary corrupts all but the j^{th} party, this means that he can obtain all the master secret keys of the FE scheme except the j^{th} one. This is effectively the same as all except the j^{th} candidate being broken. At this point, we can use the security of the j^{th} FE scheme to argue the security of the MPC protocol. This shows that the above template yields a secure two-round MPC protocol assuming a secure FE scheme.

Note that we also assume a two-round (communication-inefficient) MPC protocol. Without showing that our protocol has depth-proportional communica-

tion, the above protocol doesn't achieve anything new. Indeed, it is unclear why our protocol should have depth-proportional communication. There are two sources of concern: (i) we are still using a communication *in*efficient MPC protocol and, (ii) the functional key of f could be proportional to the size of the circuit computing f. Suppose we had a secure (magical) FE scheme satisfying the following two properties: (1) the encryption complexity of this FE scheme is proportional only to the depth of f, and (2) the functional key of f is of the form (f, aux), where $|\mathsf{aux}|$ only depends on the depth of the circuit computing f. We claim that this would immediately show that our protocol has communication complexity proportional only to the depth. Concern (i) is addressed by the fact the communication-inefficient MPC protocol is used only to evaluate the encryption circuit of the underlying FE scheme. Since the underlying FE scheme is succinct, the size of the encryption circuit only depends on the depth of the functionality f. Therefore, the communication complexity of the communication-inefficient MPC protocol does not affect our construction. Concern (ii) is handled by the fact that the parties only have to send the "aux" part of the function keys to the other parties, which is only proportional to the depth of f.

We next observe that the functional encryption scheme of Goldwasser et al. [51] can be used to satisfy both properties (1) and (2). We recall the functional encryption construction of Goldwasser et al.: the building blocks in this construction are attribute based encryption (ABE) for circuits, fully homomorphic encryption (FHE), and garbling schemes.

- To encrypt a message x, first encrypt x using a (leveled) FHE scheme. Suppose the maximum output length of the functions for which we generate functional keys is L_{out}. Generate $\mathsf{poly}(L_{out})$ ABE encryptions of the FHE ciphertext, for some fixed polynomial poly, along with wire keys of a garbled circuit. The garbled circuit is associated with the FHE decryption circuit.
- A functional key of f consists of $\mathsf{poly}(L_{out})$ ABE keys associated with the circuit that computes the FHE evaluation of f.

If we instantiate the ABE scheme with the scheme of Boneh et al. [27] and the leveled FHE scheme with any of the schemes proposed in [49,31], we achieve both properties (1) and (2) described above. The schemes of [27] and [49,31] have encryption complexity proportional only to the depth of the circuit. In terms of the structure of the functional key, we note that the ABE scheme of [27] satisfies this nice property: you can express the ABE key of a function f as (f, aux), where $|\mathsf{aux}|$ is a polynomial in depth and L_{out}. This can be used to argue that the above FE scheme satisfies property (2).

Thus starting from an FE combiner, we have constructed a communication-efficient two-round MPC. We note that the FE combiner is required to satisfy simulation security in order to prove that the resulting MPC is simulation secure. The security proof of the resulting MPC directly follows from the simulation security of the FE combiner and the simulation security of the underlying communication inefficient MPC.

Next, we show how to construct such an FE combiner.

Constructing the FE combiner: As in the works of [6,7], we view the FE candidates as analogous to parties in a secure MPC protocol. Suppose we want to construct an FE combiner for n candidates. We start with a two-round (semi-honest) secure n-party MPC protocol in the plain model. To encrypt a message x, first additively secret share x into shares $(x_1, \ldots, x_n)$. Compute the first round messages of all the parties, where the i^{th} party's input is x_i. Finally, for every $i \in [n]$, encrypt the first round messages of all the parties along with the local state of the i^{th} party using i^{th} FE candidate. All the n encryptions will form the ciphertext corresponding to the FE combiner scheme.

To generate a functional key for f, we generate n functional keys with each key associated with an FE candidate. The i^{th} functional key computes the next message function of the i^{th} party. In this context, we define the next message function to be a deterministic algorithm that takes as input the state of the party along with the messages received so far and produces the next message. Moreover, the MPC functionality associated with the next message function is as follows: it takes as input n shares of x, reconstructs x, and computes $f(x)$. The functional key of f corresponding to the FE combiner is the collection of all these n functional keys.

The decryption in the FE combiner scheme proceeds by recovering the first and second round messages of all the parties. The reconstruction algorithm of the secure MPC protocol is then executed to recover the output of the functionality. An issue here is that the reconstruction part need not be publicly computable. Meaning that it might not be possible to recover the output of the functionality from the transcript of the protocol alone. This can be resolved by revealing the local state of one of the parties to the FE evaluator who can then use this to recover the output. We implement this by considering an $(n + 1)$-party MPC protocol with the FE evaluator corresponding to one of the parties in the MPC protocol.

Without restricting ourselves to a specific type of two-round secure MPC protocols, the above template could be ill defined for two reasons:

- *Function-Dependence*: The first round messages of the MPC protocol we start off with could depend on the functionality being securely computed. This means that the FE encryptor needs to be aware of the function f when it is encrypting the message x. Hence, we need to enforce a delayed function-dependence property on the underlying MPC protocol. Roughly, this property states that the first round messages of the MPC protocol are independent of the functionality being securely computed.
- *Reusability*: Suppose we wish to construct a *collusion-resistant* FE combiner, meaning that the FE combiner is secure even if the adversary obtains multiple functional keys during the security experiment. Even if one of the candidates is secure in the collusion-resistant setting, the above template doesn't necessarily yield a collusion-resistant FE combiner. This is because the first

round MPC messages are "reused" across different FE evaluations. The security of MPC, as is, doesn't necessarily guarantee any security if the first round messages are reused for secure computation of multiple functionalities. Hence, we need to enforce a corresponding reusability property on the underlying MPC protocol to make it work in the collusion resistant setting.

Once we start with a delayed function-dependent and reusable secure MPC protocol, we can implement an FE combiner using the above template. We observe that the schemes of [72,74,33] are both delayed function-dependent and reusable. As a corollary, we obtain an FE combiner based on learning with errors.

We note that this would give an FE combiner that satisfies indistinguishability security. This is inherent since collusion-resistant FE that is also simulation secure was shown to be impossible [2]. Thus, for our application of communication efficient MPC, we construct a simulation secure FE combiner in the single-key setting (i.e., the adversary can only submit one function query) starting from a threshold fully homomorphic encryption scheme.

FE Combiner from Weaker Assumptions: The above constructions and previous constructions of FE combiners [7] relied on the learning with errors assumption. However, it would be interesting to try to construct an FE combiner from weaker assumptions. Our first observation is that there is a simple construction of an FE combiner for two FE candidates. In this case, one can simply "nest" the two candidates. That is, if the candidates are denoted FE_1 and FE_2, we encrypt a message x by first encrypting x under FE_1 and then encrypting the resulting ciphertext under FE_2. To construct a function key for f, we first construct the function key SK_1 for f using FE_1 and then construct the function key SK_2 for the decryption circuit of FE_1, with SK_1 hardcoded as the function key, using FE_2. SK_2 is then the function key for f in the nested scheme. In fact, this nested approach works to combine any constant d number of candidates. However, this approach does not scale polynomially in the number of candidates, and therefore, does not give us an FE combiner for a polynomial number of candidates.

Using the above observation, we note that we can evaluate circuits over a constant number of inputs. In particular, we can evaluate constant-sized products. If we could compute the sum of various constant-sized products, then we could compute constant-degree polynomials, which would allow us to apply known bootstrapping techniques to go from FE for constant degree polynomials to FE for arbitrary functions via randomized encodings. Such randomized encodings can be constructed assuming a PRG in NC^1 [11]. But how do we go about computing the sums of constant degree polynomials? To reason about this, we will view this as an MPC problem, where each FE candidate is associated with a party. Given an input x, we bitwise secret share x amongst all the parties. This effectively gives us an MPC problem where each party/candidate has a secret input (their share of x). For simplicity, let's consider the case where each candidate is given a single bit (the ith candidate is given the bit x_i). As an example,

suppose we wished to evaluate the polynomial

$$x_1^2 + x_1x_2 + x_1x_3 + x_2x_3.$$

Using the simple nested combiner for two candidates, we could evaluate each monomial and then sum the resulting monomial evaluations to compute the polynomial. However, this approach is flawed, since it will leak the values of each of the monomials, whereas functional encryption requires *only* the value of the polynomial to be computable and nothing else. We resolve this issue by masking each of the monomial evaluations by secret shares of 0 such that summing all these values gives the correct polynomial evaluation, but the individual computed monomial evaluations hide the true values of the monomials. To illustrate this, for the above polynomial, candidate 1 has its secret input in 3 monomials x_1^2, x_1x_2, and x_1x_3. We secret share 0 across 3 shares. Let $\mathsf{Share}_{1,1}, \mathsf{Share}_{1,2}, \mathsf{Share}_{1,3}$ denote these values, where

$$\mathsf{Share}_{1,1} + \mathsf{Share}_{1,2} + \mathsf{Share}_{1,3} = 0.$$

Similarly, candidates 2 and 3 have their secret inputs in 2 monomials: x_1x_2, x_2x_3 for candidate 2 and x_1x_3, x_2x_3 for candidate 3. We secret share 0 across 2 shares for each of these candidates. These shares are denoted $\mathsf{Share}_{2,1}, \mathsf{Share}_{2,2}$ for candidate 2 and $\mathsf{Share}_{3,1}, \mathsf{Share}_{3,2}$ for candidate 3. We then place a total ordering on the monomials of the polynomial in order to assign the shares to the monomials. Suppose our ordering was

$$x_1^2 < x_1x_2 < x_1x_3 < x_2x_3.$$

Then, we would see that x_1^2 was the first monomial containing x_1 and assign $\mathsf{Share}_{1,1}$ to this monomial. For x_1x_2, we see that it is the second monomial containing x_1 and the first monomial containing x_2. Therefore, we assign the shares $\mathsf{Share}_{1,2}$ and $\mathsf{Share}_{2,1}$ to the monomial x_1x_2. In a similar manner, we assign the shares $\mathsf{Share}_{1,3}, \mathsf{Share}_{3,1}$ to x_1x_3 and the shares $\mathsf{Share}_{2,2}, \mathsf{Share}_{3,2}$ to x_2x_3. When generating the function key to evaluate the monomial x_1^2, we actually give out a function key that evaluates $x_1^2 + \mathsf{Share}_{1,1}$. Similarly, when generating a function key to evaluate the monomial x_1x_2, we actually give out a function key that evaluates $x_1x_2 + \mathsf{Share}_{1,2} + \mathsf{Share}_{2,1}$.

By proceeding in this manner, we have made it so that each monomial evaluation hides the actual monomial value, but the sum of the monomial evaluations gives the polynomial value. However, this approach still raises several concerns: (i) how can we ensure that our secret sharing procedure hides intermediate sums of monomials, and (ii) how can we coordinate the randomness needed to generate the secret shares amongst the various monomials. To illustrate the first issue, suppose that the polynomial to evaluate was $x_1 + x_2$. In this instance, we would not add any secret shares, which would reveal x_1 and x_2. Fortunately, the first issue is not an issue at all, since such problematic polynomials will not occur. This is because we begin by secret sharing the bits of the input x amongst the candidates. Therefore, every monomial will be broken into the sum of new

monomials, such that each candidate contains a private bit in one of these new monomials. Since one of the candidates is secure, the secret sharing amongst the monomials with bits corresponding to the secure candidate ensures that nothing except the actual polynomial evaluation can be learned. To solve issue (ii), we utilize a PRF and generate a random PRF key for each candidate. This PRF key is then used to generate the secret shares of 0 associated with that candidate.

Organization: We begin by defining the notion of functional encryption and secure multi-party computation in Section 2. In Section 3, we define the notion of a functional encryption combiner. In Section 4, we show how to build a decomposable FE combiner that will be used as a building block in the construction of our round-optimal and communication efficient MPC protocol and how to instantiate it from [51]. In Section 5, we give the construction of our round-optimal and communication efficient MPC protocol. In Section 6, we show how to build an FE combiner assuming the existence of a PRG in NC^1. In Section 7, we demonstrate how to convert a delayed function-dependent and reusable round-optimal secure MPC protocol into an FE combiner. Finally, in Section 8, we show how to convert an FE combiner into a robust FE combiner and build a universal functional encryption scheme.

2 Preliminaries

We denote the security parameter by λ. For an integer $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We use $\mathcal{D}_0 \cong_c \mathcal{D}_1$ to denote that two distributions $\mathcal{D}_0, \mathcal{D}_1$ are computationally indistinguishable. We use $\mathsf{negl}(\lambda)$ to denote a function that is negligible in λ. We use $x \leftarrow \mathcal{A}$ to denote that x is the output of a randomized algorithm $\mathcal{A}$, where the randomness of $\mathcal{A}$ is sampled from the uniform distribution.

2.1 Functional Encryption

We define the notion of a (secret key) functional encryption candidate and a (secret key) functional encryption scheme. A functional encryption candidate is associated with the correctness requirement, while a secure functional encryption scheme is associated with both correctness and security.

Syntax of a Functional Encryption Candidate/Scheme. A functional encryption (FE) candidate/scheme FE for a class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ consists of four polynomial time algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ defined as follows. Let $\mathcal{X}_\lambda$ be the input space of the circuit class $\mathcal{C}_\lambda$ and let $\mathcal{Y}_\lambda$ be the output space of $\mathcal{C}_\lambda$. We refer to $\mathcal{X}_\lambda$ and $\mathcal{Y}_\lambda$ as the input and output space of the candidate/scheme, respectively.

- **Setup,** $\mathsf{MSK} \leftarrow \mathsf{FE.Setup}(1^\lambda)$: It takes as input the security parameter λ and outputs the master secret key MSK.
- **Encryption,** $\mathsf{CT} \leftarrow \mathsf{FE.Enc}(\mathsf{MSK}, m)$: It takes as input the master secret key MSK and a message $m \in \mathcal{X}_\lambda$ and outputs CT, an encryption of m.
- **Key Generation,** $\mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C)$: It takes as input the master secret key MSK and a circuit $C \in \mathcal{C}_\lambda$ and outputs a function key SK_C.
- **Decryption,** $y \leftarrow \mathsf{FE.Dec}(\mathsf{SK}_C, \mathsf{CT})$: It takes as input a function secret key SK_C, a ciphertext CT and outputs a value $y \in \mathcal{Y}_\lambda$.

Throughout this work, we will only be concerned with *uniform* algorithms. That is, $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ can be represented as Turing machines (or equivalently uniform circuits).

We describe the properties associated with the above candidate.

Approximate Correctness.

Definition 1 (Approximate Correctness). *A functional encryption candidate* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* α*-correct if it satisfies the following property: for every* $C : \mathcal{X}_\lambda \to \mathcal{Y}_\lambda \in \mathcal{C}_\lambda, m \in \mathcal{X}_\lambda$ *it holds that:*

$$\Pr\left[\begin{array}{c} \mathsf{MSK} \leftarrow \mathsf{FE.Setup}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{FE.Enc}(\mathsf{MSK}, m) \\ \mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C) \\ C(m) \leftarrow \mathsf{FE.Dec}(\mathsf{SK}_C, \mathsf{CT}) \end{array}\right] \geq \alpha,$$

where the probability is taken over the coins of the algorithms.

We refer to FE candidates that satisfy the above definition of correctness with $\alpha = 1 - \mathsf{negl}(\lambda)$ *for a negligible function* $\mathsf{negl}(\cdot)$ *as (almost) correct candidates.*

Except for Section 8, we will only deal with correct candidates. Unless explicitly stated otherwise, all FE candidates throughout this paper satisfy (almost) correctness.

IND-Security. We recall indistinguishability-based selective security for FE. This security notion is modeled as a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ where the adversary can request functional keys and ciphertexts from $\mathcal{C}$. Specifically, $\mathcal{A}$ can submit function queries C and $\mathcal{C}$ responds with the corresponding functional keys. $\mathcal{A}$ can also submit message queries of the form (x_0, x_1) and receives an encryption of messages x_b for some bit $b \in \{0, 1\}$. The adversary $\mathcal{A}$ wins the game if she can guess b with probability significantly more than $1/2$ and if for all function queries C and message queries (x_0, x_1), $C(x_0) = C(x_1)$. That is to say, any function evaluation that is computable by $\mathcal{A}$ gives the same value regardless of b. It is required that the adversary must declare the challenge messages at the beginning of the game.

Definition 2 (IND-secure FE). *A secret-key FE scheme* FE *for a class of circuits* $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in[\mathbb{N}]}$ *and message space* $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in[\mathbb{N}]}$ *is selectively secure if for any PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mu(\cdot)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is*

$$\mathsf{Adv}^{\mathsf{FE}}_{\mathcal{A}} = \left|\mathsf{Pr}[\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \mathsf{Pr}[\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ *and* $\lambda \in \mathbb{N}$*, the experiment* $\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, b)$ *is defined below:*

1. ***Challenge message queries****:* $\mathcal{A}$ *submits message queries,*

$$\left\{(x_0^i, x_1^i)\right\}$$

   with $x_0^i, x_1^i \in \mathcal{X}_\lambda$ *to the challenger* $\mathcal{C}$*.*
2. $\mathcal{C}$ *computes* $\mathsf{MSK} \leftarrow \mathsf{FE.Setup}(1^\lambda)$ *and then computes* $\mathsf{CT}_i \leftarrow \mathsf{FE.Enc}(\mathsf{MSK}, x_b^i)$ *for all* i*. The challenger* $\mathcal{C}$ *then sends* $\{\mathsf{CT}_i\}$ *to the adversary* $\mathcal{A}$*.*
3. ***Function queries****: The following is repeated an at most polynomial number of times:* $\mathcal{A}$ *submits a function query* $C \in \mathcal{C}_\lambda$ *to* $\mathcal{C}$*. The challenger* $\mathcal{C}$ *computes* $\mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C)$ *and sends it to* $\mathcal{A}$*.*
4. *If there exists a function query* C *and challenge message queries* (x_0^i, x_1^i) *such that* $C(x_0^i) \neq C(x_1^i)$*, then the output of the experiment is set to* $\perp$*. Otherwise, the output of the experiment is set to* b'*, where* b' *is the output of* $\mathcal{A}$*.*

Adaptive Security. The above security notion is referred to as selective security in the literature. One can consider a stronger notion of security, called *adaptive security*, where the adversary can interleave the challenge messages and the function queries in any arbitrary order. Analogous to Definition 2, we can define an adaptively secure FE scheme. In this paper, we only deal with selectively secure FE schemes. However, the security of these schemes can be upgraded to adaptive with no additional cost [3].

Simulation Security. We can also consider a different notion of security, called (single-key) simulation security.

Definition 3. *(SIM-Security) Let* FE *denote a functional encryption scheme for a circuit class* $\mathcal{C}$*. For every PPT adversary* $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ *and a PPT simulator* Sim*, consider the following two experiments:*

$\mathsf{Exp}^{\mathsf{real}}_{\mathsf{FE},\mathsf{A}}(1^\lambda)$	$\mathsf{Exp}^{\mathsf{ideal}}_{\mathsf{FE},\mathsf{A},\mathsf{Sim}}(1^\lambda)$
$\{\mathsf{FE.Setup}(1^\lambda) \to \mathsf{MSK}\}$	$\{\mathsf{FE.Setup}(1^\lambda) \to \mathsf{MSK}\}$
$\mathsf{A}_1 \to (C, \mathsf{state}_{\mathsf{A}_1})$	$\mathsf{A}_1 \to (C, \mathsf{state}_{\mathsf{A}_1})$
$\{\mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C)\}$	$\{\mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C)\}$

$\mathsf{A}_2(\mathsf{state}_{\mathsf{A}_1}, \mathsf{SK}_C) \to (m, \mathsf{state}_{\mathsf{A}_2})$	$\mathsf{A}_2(\mathsf{state}_{\mathsf{A}_1}, \mathsf{SK}_C) \to (m, \mathsf{state}_{\mathsf{A}_2})$
$\mathsf{FE.Enc}(\mathsf{MSK}, m) \to \mathsf{CT}$	$\mathsf{Sim}(\mathsf{MSK}, C, \mathsf{SK}_C, C(m)) \to \mathsf{CT}$
Output $(\mathsf{CT}, \mathsf{state}_{\mathsf{A}_2})$	*Output* $(\tilde{\mathsf{CT}}, \mathsf{state}_{\mathsf{A}_2})$

The scheme is said to be (single-key) SIM-secure if there exists a PPT simulator Sim *such that for all PPT adversaries* $(\mathsf{A}_1, \mathsf{A}_2)$*, the outcomes of the two experiments are computationally indistinguishable:*

$$\{\mathsf{Exp}^{\mathsf{real}}_{\mathsf{FE},\mathsf{A}}(1^\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{\mathsf{Exp}^{\mathsf{ideal}}_{\mathsf{FE},\mathsf{A},\mathsf{Sim}}(1^\lambda)\}_{\lambda\in\mathbb{N}}$$

Collusions. We can parameterize the FE candidate by the number of function secret key queries that the adversary can make in the security experiment. If the adversary can only submit an a priori upper bounded q secret key queries, we say that the scheme is q-key or q-collusion secure. We say that the functional encryption scheme unbounded-key or unbounded-collusion secure if the adversary can make an unbounded (polynomial) number of function secret key queries. In this work, unless otherwise stated, we will allow the adversary to make an arbitrary polynomial number of function secret key queries.

Succinctness.

Definition 4 (Succinctness). *A functional encryption candidate* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *for a circuit class* $\mathcal{C}$ *containing circuits that take inputs of length* ℓ_{in}*, outputs strings of length* ℓ_{out} *bits and are of depth at most* d *is said to be succinct if the following holds: For any circuit* $C \in \mathcal{C}$*,*

- *Let* $\mathsf{MSK} \leftarrow \mathsf{FE.Setup}(1^\lambda)$*. The size of the circuit* $\mathsf{FE.Enc}(\mathsf{MSK}, \cdot) < \mathsf{poly}(\lambda, d, \ell_{\mathsf{in}}, \ell_{\mathsf{out}})$ *for some polynomial* poly*.*
- *The function key* $\mathsf{SK}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{MSK}, C)$ *is of the form* (C, aux) *where* $|\mathsf{aux}| \leq \mathsf{poly}(\lambda, d, \ell_{\mathsf{out}})$ *for some polynomial* poly*.*

In general, an FE candidate/scheme need not satisfy succinctness. However, we will need to utilize succinct FE candidates when constructing depth-proportional communication MPC (Section 4 and Section 5). In such cases, we will explicitly state that the FE candidates are succinct.

FE Candidates vs. FE Schemes. As defined above, an FE scheme must satisfy *both* correctness and security, while an FE candidate is simply the set of algorithms. Unless otherwise specified, we will be dealing with FE candidates that satisfy correctness. We will only refer to FE constructions as FE schemes if it is known that the construction satisfies both correctness and security.

2.2 Secure Multi-Party Computation

The syntax and security definitions for secure multi-party computation can be found in the full version. Since we are dealing throughout this paper with the efficiency of MPC protocols, we give the definition of a succinct MPC protocol below.

Definition 5 (Succinct MPC protocol). *Consider an n-party semi-honest secure MPC protocol Π for a functionality f, represented by a polynomial-sized circuit C. We define the communication complexity of Π to be the total length of all the messages exchanged in the protocol.*

We define Π to be succinct if the communication complexity of Π is $\mathsf{poly}(\lambda, d, n)$*, where λ is the security parameter and d is the depth of the circuit C.*

2.3 Additional Preliminaries

In this work, we will also make occasional use of threshold leveled fully homomorphic encryption [12,72,26] and garbling schemes [79,18]. Formal definitions of these primitives can be found in the full version.

3 FE Combiners: Definition

In this section, we give a formal definition of an FE combiner. Intuitively, an FE combiner FEComb takes n FE candidates, $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$ and compiles them into a new FE candidate with the property that FEComb is a secure FE scheme provided that at least one of the n FE candidates is a secure FE scheme.

Syntax of a Functional Encryption Combiner. A functional encryption combiner FEComb for a class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ consists of four polynomial time algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ defined as follows. Let $\mathcal{X}_\lambda$ be the input space of the circuit class $\mathcal{C}_\lambda$ and let $\mathcal{Y}_\lambda$ be the output space of $\mathcal{C}_\lambda$. We refer to $\mathcal{X}_\lambda$ and $\mathcal{Y}_\lambda$ as the input and output space of the combiner, respectively. Furthermore, let $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$ denote the descriptions of n FE candidates.

- **Setup,** $\mathsf{FEComb.Setup}(1^\lambda, \{\mathsf{FE}_i\}_{i\in[n]})$: It takes as input the security parameter λ and the descriptions of n FE candidates $\{\mathsf{FE}_i\}_{i\in[n]}$ and outputs the master secret key MSK.
- **Encryption,** $\mathsf{FEComb.Enc}(\mathsf{MSK}, \{\mathsf{FE}_i\}_{i\in[n]}, m)$: It takes as input the master secret key MSK, the descriptions of n FE candidates $\{\mathsf{FE}_i\}_{i\in[n]}$, and a message $m \in \mathcal{X}_\lambda$ and outputs CT, an encryption of m.
- **Key Generation,** $\mathsf{FEComb.Keygen}\left(\mathsf{MSK}, \{\mathsf{FE}_i\}_{i\in[n]}, C\right)$: It takes as input the master secret key MSK, the descriptions of n FE candidates $\{\mathsf{FE}_i\}_{i\in[n]}$, and a circuit $C \in \mathcal{C}_\lambda$ and outputs a function key SK_C.

– **Decryption,** $\mathsf{FEComb.Dec}\left(\{\mathsf{FE}_i\}_{i\in[n]}, \mathsf{SK}_C, \mathsf{CT}\right)$: It is a deterministic algorithm that takes as input the descriptions of n FE candidates $\{\mathsf{FE}_i\}_{i\in[n]}$, a function secret key SK_C, and a ciphertext CT and outputs a value $y \in \mathcal{Y}_\lambda$.

Remark 1. In the formal definition above, we have included $\{\mathsf{FE}_i\}_{i\in[n]}$, the descriptions of the FE candidates, as input to all the algorithms of FEComb. For notational simplicity, we will often forgo these inputs and assume that they are implicit.

We now define the properties associated with an FE combiner. The three properties are correctness, polynomial slowdown, and security. Correctness is analogous to that of an FE candidate, provided that the n input FE candidates are all valid FE candidates. Polynomial slowdown says that the running times of all the algorithms of FEComb are polynomial in λ and n. Finally, security intuitively says that if at least one of the FE candidates is also secure, then FEComb is a secure FE scheme. We provide the formal definitions below.

Correctness.

Definition 6 (Correctness). *Suppose* $\{\mathsf{FE}_i\}_{i\in[n]}$ *are correct FE candidates. We say that an FE combiner is correct if for every circuit* $C : \mathcal{X}_\lambda \to \mathcal{Y}_\lambda \in \mathcal{C}_\lambda$*, and message* $m \in \mathcal{X}_\lambda$ *it holds that:*

$$\Pr\left[\begin{array}{c} \mathsf{MSK} \leftarrow \mathsf{FEComb.Setup}(1^\lambda, \{\mathsf{FE}_i\}_{i\in[n]}) \\ \mathsf{CT} \leftarrow \mathsf{FEComb.Enc}(\mathsf{MSK}, \{\mathsf{FE}_i\}_{i\in[n]}, m) \\ \mathsf{SK}_C \leftarrow \mathsf{FEComb.Keygen}(\mathsf{MSK}, \{\mathsf{FE}_i\}_{i\in[n]}, C) \\ C(m) \leftarrow \mathsf{FEComb.Dec}(\{\mathsf{FE}_i\}_{i\in[n]}, \mathsf{SK}_C, \mathsf{CT}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda),$$

where the probability is taken over the coins of the algorithms and $\mathsf{negl}(\lambda)$ *is a negligible function in* λ*.*

Polynomial Slowdown.

Definition 7 (Polynomial Slowdown). *An FE combiner* FEComb *satisfies polynomial slowdown if on all inputs, the running times of* $\mathsf{FEComb.Setup}$, $\mathsf{FEComb.Enc}$, $\mathsf{FEComb.Keygen}$, *and* $\mathsf{FEComb.Dec}$ *are at most* $\mathsf{poly}(\lambda, n)$*, where* n *is the number of FE candidates that are being combined.*

IND-Security.

Definition 8 (IND-Secure FE Combiner). *An FE combiner* FEComb *is selectively secure if for any set* $\{\mathsf{FE}_i\}_{i\in[n]}$ *of correct FE candidates, it satisfies Definition 2, where the descriptions of* $\{\mathsf{FE}_i\}_{i\in[n]}$ *are public and implicit in all invocations of the algorithms of* FEComb*,* if at least one *of the FE candidates* $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$ *also satisfies Definition 2.*

Note that *Definition 2* is the IND-security definition for FE. Unless otherwise specified, when we say a *secure* FE combiner, we refer to one that satisfies IND-security.

Simulation Security. Similarly to FE candidates, we can also consider a different notion of security called (single-key) simulation security.

Definition 9. *An FE combiner* FEComb *is single-key simulation secure if for any set* $\{\mathsf{FE}_i\}_{i\in[n]}$ *of correct FE candidates, it satisfies Definition 3, where the descriptions of* $\{\mathsf{FE}_i\}_{i\in[n]}$ *are public and implicit in all invocations of the algorithms of* FEComb, if at least one *of the FE candidates* $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$ *also satisfies Definition 3.*

Note that *Definition 3* is the simulation security definition for FE.

Succinctness. Similarly to FE candidates, we can also define the notion of a succinct FE combiner. An FE combiner is not required to satisfy succinctness, but we will utilize a succinct FE combiner when construction low communication MPC (Section 4 and Section 5).

Definition 10. *An FE combiner* $\mathsf{FEComb} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *for a circuit class* $\mathcal{C}$ *containing circuits that take inputs of length* ℓ_{in}*, outputs strings of length* ℓ_{out} *bits and are of depth at most* d *is succinct if for every set of succinct FE candidates* $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$*, the following holds: For any circuit* $C \in \mathcal{C}$*,*

- *Let* $\mathsf{MSK} \leftarrow \mathsf{FEComb.Setup}(1^\lambda, \{\mathsf{FE}_i\}_{i\in[n]})$*. The size of the circuit* $\mathsf{FEComb.Enc}(\mathsf{MSK}, \cdot) \leq \mathsf{poly}(\lambda, d, \ell_{\mathsf{in}}, \ell_{\mathsf{out}}, n)$ *for some polynomial* poly*.*
- *The function key* $\mathsf{SK}_C \leftarrow \mathsf{FEComb.KeyGen}(\mathsf{MSK}, C)$ *is of the form* (C, aux) *where* $|\mathsf{aux}| \leq \mathsf{poly}(\lambda, d, \ell_{\mathsf{out}}, n)$ *for some polynomial* poly*.*

Robust FE Combiners and Universal FE.

Remark 2. We also define the notion of a robust FE combiner. An FE combiner FEComb is robust if it is an FE combiner that satisfies the three properties (correctness, polynomial slowdown, and security) associated with an FE combiner when given any set of FE candidates $\{\mathsf{FE}_i\}_{i\in[n]}$, provided that one is a correct and secure FE candidate. No restriction is placed on the other FE candidates. In particular, they need not satisfy correctness at all.

Robust FE combiners can be used to build a universal functional encryption scheme defined below.

Definition 11 (T-Universal Functional Encryption). *We say that an explicit Turing machine* $\Pi_{\mathsf{univ}} = (\Pi_{\mathsf{univ}}.\mathsf{Setup}, \Pi_{\mathsf{univ}}.\mathsf{Enc}, \Pi_{\mathsf{univ}}.\mathsf{KeyGen}, \Pi_{\mathsf{univ}}.\mathsf{Dec})$ *is a universal functional encryption scheme parametrized by* T *if* Π_{univ} *is a correct and secure FE scheme assuming the existence a correct and secure FE scheme with runtime* $< T$*.*

4 Succinct Single-Key Simulation Secure Decomposable FE Combiner

In this section, we define and construct a succinct single-key simulation secure decomposable FE combiner ($\mathsf{DFEComb}$ for short) that will be useful later for our communication-efficient MPC result. This section can be found in the full version.

5 Round Optimal MPC with Depth-Proportional Communication from an FE Combiner

In this section, using any succinct single-key simulation secure decomposable FE combiner (see Section 4), we show how to compile any two round semi-honest secure MPC protocol into one where the communication complexity is proportional only to the depth of the circuit being evaluated.

Let $\mathsf{Comm.Compl}(\pi)$ denote the communication complexity of any protocol π. Let λ denote the security parameter, n denote the number of parties, and ℓ denote the size of the input to each party. Formally, we show the following theorem:

Theorem 5. *Assuming the existence of*

- *A succinct single-key single-ciphertext simulation secure decomposable FE combiner (AND)*
- *Succinct FE candidates (AND)*
- *A two round semi-honest MPC in the plain model (that may not be communication efficient) that is secure against up to all but one corruption,*

there exists a two round semi-honest MPC protocol π in the plain model that is secure against up to all but one corruption for any boolean circuit C, where the communication complexity of the protocol π is independent of the size of the circuit. That is, $\mathsf{Comm.Compl}(\pi) = \mathsf{poly}(\mathsf{Depth}(C), n, \ell, \lambda)$.

We know how to construct a succinct single-key simulation secure decomposable FE combiner based on the learning with errors (LWE) assumption (see Section 4). Further, in Section 4, we saw that the construction in [51] is a succinct FE candidate. Also, two round semi-honest MPC protocols secure against up to all but one corruption can be based on the LWE assumption [48,20,75][7]. Instantiating the primitives in the above theorem, we get the following corollary:

Corollary 1. *Assuming LWE, there exists a two round semi-honest MPC protocol π in the plain model that is secure against up to all but one corruption for any boolean circuit C with* $\mathsf{Comm.Compl}(\pi) = \mathsf{poly}(\mathsf{Depth}(C), n, \ell, \lambda)$.

[7] [48,20] showed how to construct two round semi-honest MPC in the plain model from any two round semi-honest OT in the plain model and [75] show that the latter can be constructed from LWE.

Furthermore, if we allow our protocol to have a preprocessing phase, we can obtain a two round semi-honest MPC protocol with depth-proportional communication complexity and with the computational complexity of each party in the online phase independent of the size of the circuit, matching the result of [76]. By simply making steps $1-3$ in round 1 of our construction the preprocessing phase, we arrive at the following corollary:

Corollary 2. *Assuming LWE, there exists a two round semi-honest MPC protocol π in the plain model that is secure against up to all but one corruption for any boolean circuit C with* $\mathsf{Comm.Compl}(\pi) = \mathsf{poly}(\mathsf{Depth}(C), n, \ell, \lambda)$ *and with the computational complexity of the online phase* $\mathsf{poly}(\mathsf{Depth}(C), n, \ell, \lambda)$.

5.1 Construction

Notation:

- Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ with inputs $x_1, \ldots, x_n$, respectively, who wish to evaluate a boolean circuit C on their joint inputs. Let λ denote the security parameter and without loss of generality, let's assume $|x_i| = \lambda$ for all $i \in [n]$. Also, let's denote the randomness of each party P_i as $r_i = (r_i^{\mathsf{Setup}}, r_i^{\mathsf{Enc}}, r_i^{\mathsf{SH}}, r_i^{\mathsf{KeyGen}})$.
- Let $\mathsf{DFEComb} = (\mathsf{DFEComb.Setup}, \mathsf{DFEComb.Enc}, \mathsf{DFEComb.Keygen}, \mathsf{DFEComb.Dec}, \mathsf{DFEComb.Partition})$ be a succinct single-key simulation secure decomposable FE combiner (see Section 4) for n FE candidates $\mathsf{FE}_1, \ldots, \mathsf{FE}_n$.
- Let π^{SH} be a two round semi-honest secure MPC protocol (not necessarily communication efficient). Let $(\pi^{\mathsf{SH}}.\mathsf{Round}_1, \pi^{\mathsf{SH}}.\mathsf{Round}_2)$ denote the algorithms used by any party to compute the messages in each of the two rounds and $\pi^{\mathsf{SH}}.\mathsf{Out}$ denote the algorithm to compute the final output. Further, let $\pi^{\mathsf{SH}}.\mathsf{Sim} = (\pi^{\mathsf{SH}}.\mathsf{Sim}_1, \pi^{\mathsf{SH}}.\mathsf{Sim}_2)$ denote the simulator for this protocol - that is, $\pi^{\mathsf{SH}}.\mathsf{Sim}_i$ is the simulator's algorithm to compute the i^{th} round's messages.

Protocol. We now describe the construction of our protocol π with depth-proportional communication complexity.

- **Round 1:** Each party P_i does the following:
  1. Generate $\mathsf{MSK}_i \leftarrow \mathsf{FE}_i.\mathsf{Setup}(1^\lambda)$ using randomness r_i^{Setup}.
  2. Compute $(C_1, \ldots, C_n) \leftarrow \mathsf{DFEComb.Partition}(1^\lambda, C)$.
  3. Compute $\mathsf{SK}_i = \mathsf{FE}_i.\mathsf{KeyGen}(\mathsf{MSK}_i, C_i)$ using randomness r_i^{KeyGen}.
  4. Participate in an execution of protocol π^{SH} with the remaining $(n-1)$ parties using input $y_i = (x_i, \mathsf{MSK}_i, r_i^{\mathsf{Enc}})$ and randomness r_i^{SH} to compute the deterministic circuit C_{CT} defined in Figure 3. That is, compute the first round message $\mathsf{msg}_{1,i} \leftarrow \pi^{\mathsf{SH}}.\mathsf{Round}_1(y_i; r_i^{\mathsf{SH}})$.

5. Output $(\mathsf{msg}_{1,i}, \mathsf{SK}_i)$.

– **Round 2:** Each party P_i does the following:
1. Let τ_1 denote the transcript of protocol π^{SH} after round 1.
2. Compute the second round message $\mathsf{msg}_{2,i} \leftarrow \pi^{\mathsf{SH}}.\mathsf{Round}_2(y_i, \tau_1; r_i^{\mathsf{SH}})$ where $y_i = (x_i, \mathsf{MSK}_i, r_i^{\mathsf{Enc}})$.
3. Output $(\mathsf{msg}_{2,i})$.

– **Output Computation:** Each party P_i does the following:
1. Let τ_2 denote the transcript of protocol π^{SH} after round 2.
2. Compute the output of π^{SH} as $\mathsf{CT} \leftarrow \pi^{\mathsf{SH}}.\mathsf{Out}(y_i, \tau_2; r_i^{\mathsf{SH}})$.
3. Let $\mathsf{SK}_C = (\mathsf{SK}_1, \ldots, \mathsf{SK}_n)$.
4. Output $\mathsf{DFEComb.Dec}(\mathsf{SK}_C, \mathsf{CT})$.

Input: $\{(x_i, \mathsf{MSK}_i, r_i^{\mathsf{Enc}})\}_{i=1}^n$

- Let $\mathsf{MSK} = (\mathsf{MSK}_1, \ldots, \mathsf{MSK}_n)$, $x = (x_1, \ldots, x_n)$ and $r = (r_1^{\mathsf{Enc}}, \ldots, r_n^{\mathsf{Enc}})$.
- Output $\mathsf{DFEComb.Enc}(\mathsf{MSK}, x)$ using randomness r.

Fig. 3. Circuit C_{CT}

Correctness and Efficiency: Correctness follows immediately from the construction. In particular, at the end of the protocol, each party possesses CT, an encryption of $x = (x_1, \ldots, x_n)$ under the FE combiner, and SK_C, the function key for C. This ciphertext can then be decrypted using SK_C to yield $C(x)$, as desired.

Now, let's analyze the communication complexity of the protocol. First, observe that each circuit C that is of depth d and outputs a single bit is partitioned into n circuits $C_1, \ldots, C_n$ by running the $\mathsf{DFEComb.Partition}$ algorithm. The circuit C_i just computes a partial decryption of $\mathsf{TFHE.Eval}(C, \cdot)$. Now, even though C is a boolean circuit, the output length of C_i might not be 1. However, this is not an issue for us. Indeed, observe that from the compactness of the TFHE scheme, the length of the partial decryption is just $\mathsf{poly}(\lambda, d)$ for some fixed polynomial poly for all circuits C with depth d and output length 1. Thus, the size of the output length of C_i for all $i \in [n]$ is at most $\mathsf{poly}(\lambda, d)$ bits. Thus, from Section 4, we know that $|\mathsf{SK}_i| = \mathsf{poly}(d, n, \lambda)$ and $|\mathsf{CT}| = \mathsf{poly}(d, n, \lambda)$. Recall that CT is the ciphertext that is the output of the protocol π^{SH} (computed during decryption). In fact, from Section 4, we also know that the size of the circuit computing the ciphertext CT is also bounded by $\mathsf{poly}(d, n, \lambda)$. Then, for the protocol π^{SH} recall that the input is $y_i = (x_i, \mathsf{MSK}_i, r_i^{\mathsf{Enc}})$ and so $|y_i| = \mathsf{poly}(\lambda, d)$

for some polynomial. Therefore, for each party P_i, $|\mathsf{msg}_{1,i}| = \mathsf{poly}(d, n, \lambda)$ and $|\mathsf{msg}_{2,i}| = \mathsf{poly}(d, n, \lambda)$.

Therefore, in our two round protocol π, in each round the size of the message sent by any party is $\mathsf{poly}(n, d, \lambda)$. Thus, $\mathsf{Comm.Compl}(\pi) = \mathsf{poly}(n, d, \lambda)$.

The above analysis was for circuits with boolean output. For circuits that output multi-bit strings, the communication complexity of our MPC protocol π is bounded by $\mathsf{poly}(n, d, \lambda) \cdot \ell_{\mathsf{out}}$, where ℓ_{out} is the output length of the circuit. This follows immediately by viewing the multi-bit output circuit as ℓ_{out} different boolean circuits and running in parallel.

5.2 Security Proof

The security proof can be found in the full version.

6 Construction of an FE Combiner from Weaker Assumptions

In this section, we employ a tool extensively used in the secure multi-party computation literature, namely, randomized encodings to construct an FE combiner. Roughly speaking, a randomized encoding is a mechanism to "efficiently" encode a function f and an input x such that the encoding reveals $f(x)$ and nothing more. A randomized encoding scheme is said to be low degree if the encoding algorithm can be represented as a low degree polynomial. Low degree randomized encodings have been used to achieve constant-round secure multi-party computation [17]. We show how to use this tool to obtain functional encryption combiners. The underlying assumption used to instantiate the low degree randomized encoding is the existence of a PRG in NC^1. Formally, we show the following theorem.

Theorem 6. *Assuming the existence of a PRG in* NC^1*, there exists an unbounded-key FE combiner for polynomial-sized circuits.*

The rest of this section can be found in the full version.

7 From MPC to FE Combiners

In this section, we show how to build an FE combiner from any semi-honest MPC protocol π that satisfies a property called delayed function-dependence. This section can be found in the full version.

8 From an FE Combiner to a Robust FE Combiner

The FE combiners constructed previously are not *robust*. By this, we mean that the constructions provide no guarantee of correctness or security if any of the

underlying FE candidates do not satisfy correctness. However, determining the correctness of FE candidates may be difficult since a candidate FE may be correct with overwhelming probability on certain message, circuit pairs (m, C) and not others. With no worst-case guarantees, it can be challenging to reason about the correctness of an FE candidate especially if the function space $\mathcal{C}$ is say all poly-sized circuits, where sampling uniformly over the space is difficult.

We can mitigate this issue by making our FE combiners robust. A *robust FE combiner* is an FE combiner that satisfies correctness and security provided that at least one FE candidate, FE_i, satisfies both correctness and security. No restrictions are placed on the other FE candidates. In particular, they may satisfy neither correctness nor security. In this section, we show how to transform any FE combiner into a robust FE combiner. Formally, we show the following.

Theorem 7. *If there exists an FE combiner, then there exists a robust FE combiner.*

Combining Theorem 7 with Theorem 6, we obtain the following corollary.

Corollary 3. *Assuming the existence of a PRG in* NC^1*, there exists an unbounded-key robust FE combiner.*

This is done, at a high level, via the following steps.

1. Transform each FE candidate FE_i into a new FE candidate FE'_i such that
   (a) If FE_i is correct and secure, then FE'_i is also correct and secure.
   (b) If FE'_i is correct for any fixed message, circuit pair (m, C) with probability α, then it is at least α'-correct for all other message, circuit pairs (m', C') where $\alpha' = \alpha - \mathsf{negl}(\lambda)$.
2. Fix a message m and a circuit C and test each candidate repeatedly on (m, C) to determine if each candidate is α-correct for $\alpha \geq 1 - \frac{1}{\lambda}$. Discard those that are not.
3. Using standard techniques of BPP correctness amplification, transform the α-correct candidates into (almost) correct candidates.
4. Instantiate constructions of FE combiners from previous sections with these (almost) correct candidates.

We defer the construction and proof of Theorem 7 to the full version.

Universal Functional Encryption: Robust FE combiners are closely related to the notion of universal functional encryption. Universal functional encryption is a construction of functional encryption satisfying the following simple guarantee. If there exists a Turing Machine with running time bounded by some $T(n) = \mathsf{poly}(n)$ that implements a correct and secure FE scheme, then the universal functional encryption construction is itself a correct and secure FE scheme. Using the existence of a robust FE combiner (Theorem 7) and the results of [6], we observe the following.

Theorem 8. *Assuming the existence of a robust FE combiner, there exists a universal functional encryption scheme.*

Using the above theorem and Corollary 3, we arrive at the following corollary.

Corollary 4. *Assuming the existence of a PRG in* NC^1*, there exists a universal unbounded-key functional encryption scheme.*

9 Acknowledgements

We thank the anonymous TCC reviewers for their helpful comments.

Saikrishna Badrinarayanan, Aayush Jain, Nathan Manohar and Amit Sahai were supported in part from a DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, and NSF grant 1619348, BSF grant 2012378, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. Saikrishna Badrinarayanan is also supported by an IBM PhD fellowship. Aayush Jain is also supported by a Google PhD fellowship award in Privacy and Security. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C- 0205. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, the U.S. Government, IBM, or Google.

References

1. Agrawal, S.: New Methods for Indistinguishability Obfuscation: Bootstrapping and Instantiation. Cryptology ePrint Archive, Report 2018/633 (2018), https://eprint.iacr.org/2018/633
2. Agrawal, S., Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional Encryption: New Perspectives and Lower Bounds. In: CRYPTO (2013). https://doi.org/10.1007/978-3-642-40084-1$_2$8
3. Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From Selective to Adaptive Security in Functional Encryption. In: CRYPTO (2015)
4. Ananth, P., Choudhuri, A.R., Jain, A.: A new approach to round-optimal secure multiparty computation. In: Annual International Cryptology Conference. pp. 468–499. Springer (2017)
5. Ananth, P., Jain, A., Khurana, D., Sahai, A.: Indistinguishability Obfuscation Without Multilinear Maps: iO from LWE, Bilinear Maps, and Weak Pseudorandomness. Cryptology ePrint Archive, Report 2018/615 (2018), https://eprint.iacr.org/2018/615
6. Ananth, P., Jain, A., Naor, M., Sahai, A., Yogev, E.: Universal Constructions and Robust Combiners for Indistinguishability Obfuscation and Witness Encryption. In: CRYPTO (2016)
7. Ananth, P., Jain, A., Sahai, A.: Robust Transforming Combiners from Indistinguishability Obfuscation to Functional Encryption. In: EUROCRYPT (2017)

8. Ananth, P., Jain, A.: Indistinguishability Obfuscation from Compact Functional Encryption. In: CRYPTO (2015). https://doi.org/10.1007/978-3-662-47989-6_15
9. Ananth, P., Jain, A., Sahai, A.: Indistinguishability Obfuscation from Functional Encryption for Simple Functions. Cryptology ePrint Archive, Report 2015/730 (2015)
10. Ananth, P., Sahai, A.: Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In: EUROCRYPT (2017)
11. Applebaum, B., Ishai, Y., Kushilevitz, E.: Computationally private randomizing polynomials and their applications (extended abstract). In: CCC (June 2005)
12. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE. In: EUROCRYPT (2012)
13. Asmuth, C., Blakley, G.: An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems. Computers and Mathematics with Applications (1981)
14. Badrinarayanan, S., Goyal, V., Jain, A., Sahai, A.: A note on VRFs from Verifiable Functional Encryption. IACR Cryptology ePrint Archive **2017**, 51 (2017)
15. Badrinarayanan, S., Goyal, V., Jain, A., Kalai, Y.T., Khurana, D., Sahai, A.: Promise Zero Knowledge and Its Applications to Round Optimal MPC. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 459–487. Springer International Publishing, Cham (2018)
16. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: EUROCRYPT (2012)
17. Beaver, D., Micali, S., Rogaway, P.: The round complexity of secure protocols. In: STOC (1990)
18. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of Garbled Circuits. In: CCS (2012)
19. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC (1988)
20. Benhamouda, F., Lin, H.: k-round MPC from k-round OT via garbled interactive circuits. EUROCRYPT (2018)
21. Bitansky, N.: Verifiable random functions from non-interactive witness-indistinguishable proofs. In: TCC (2017)
22. Bitansky, N., Nishimaki, R., Passelègue, A., Wichs, D.: From cryptomania to obfustopia through secret-key functional encryption. In: TCC Part II (2016)
23. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: FOCS (2015)
24. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation: from approximate to exact. In: Theory of Cryptography Conference. pp. 67–95. Springer (2016)
25. Bitansky, N., Vaikuntanathan, V.: A note on perfect correctness by derandomization. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 592–606. Springer (2017)
26. Boneh, D., Gennaro, R., Goldfeder, S., Jain, A., Kim, S., Rasmussen, P.M., Sahai, A.: Threshold Cryptosystems From Threshold Fully Homomorphic Encryption. IACR Cryptology ePrint Archive **2017** (2017)
27. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: EUROCRYPT (2014). https://doi.org/10.1007/978-3-642-55220-5_30
28. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Theory of Cryptography, pp. 253–273. Springer (2011)

29. Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemporary Mathematics **324**(1), 71–90 (2003)
30. Boyle, E., Gilboa, N., Ishai, Y., Lin, H., Tessaro, S.: Foundations of homomorphic secret sharing. In: LIPIcs-Leibniz International Proceedings in Informatics. vol. 94. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018)
31. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) Fully Homomorphic Encryption Without Bootstrapping. ACM Trans. Comput. Theory **6**(3), 13:1–13:36 (Jul 2014)
32. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four Round Secure Computation Without Setup. In: Theory of Cryptography (2017)
33. Brakerski, Z., Perlman, R.: Lattice-based fully dynamic multi-key FHE with short ciphertexts. In: CRYPTO. pp. 190–213. Springer (2016)
34. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Theory of Cryptography Conference. pp. 468–497. Springer (2015)
35. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the Multilinear Map over the Integers. In: EUROCRYPT (2015)
36. Cheon, J.H., Jeong, J., Lee, C.: An Algorithm for CSPR Problems and Cryptanalysis of the GGH Multilinear Map without an encoding of zero. Tech. rep., Cryptology ePrint Archive, Report 2016/139 (2016)
37. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: CRYPTO (2015)
38. Coron, J.S., Gentry, C., Halevi, S., Lepoint, T., Maji, H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In: CRYPTO (2015)
39. Coron, J.S., Lee, M.S., Lepoint, T., Tibouchi, M.: Cryptanalysis of GGH15 Multilinear Maps. Cryptology ePrint Archive, Report 2015/1037 (2015)
40. Dodis, Y., Halevi, S., Rothblum, R.D., Wichs, D.: Spooky encryption and its applications. In: Annual Cryptology Conference. pp. 93–122. Springer (2016)
41. Fischlin, M., Herzberg, A., Noon, H.B., Shulman, H.: Obfuscation Combiners. In: CRYPTO (2016)
42. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT. pp. 1–17 (2013). https://doi.org/10.1007/978-3-642-38348-9$_1$
43. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Fully secure functional encryption without obfuscation. IACR Cryptology ePrint Archive **2014**, 666 (2014)
44. Garg, S., Mukherjee, P., Pandey, O., Polychroniadou, A.: The exact round complexity of secure computation. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 448–476. Springer (2016)
45. Garg, S., Pandey, O., Srinivasan, A.: Revisiting the cryptographic hardness of finding a nash equilibrium. In: CRYPTO (2016)
46. Garg, S., Pandey, O., Srinivasan, A., Zhandry, M.: Breaking the sub-exponential barrier in obfustopia. In: EUROCRYPT (2017)
47. Garg, S., Srinivasan, A.: Garbled protocols and two-round MPC from bilinear maps. FOCS (2017)
48. Garg, S., Srinivasan, A.: Two-Round Multiparty Secure Computation from Minimal Assumptions. EUROCRYPT (2018)
49. Gentry, C., Sahai, A., Waters, B.: Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In: CRYPTO (2013)
50. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: STOC. pp. 218–229. ACM (1987)

51. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: STOC (2013). https://doi.org/10.1145/2488608.2488678
52. Goldwasser, S., Klein, S., Wichs, D.: The Edited Truth. In: TCC. pp. 305–340. Springer (2017)
53. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions via multi-party computation. In: CRYPTO (2012)
54. Goyal, R., Hohenberger, S., Koppula, V., Waters, B.: A generic approach to constructing and proving verifiable random functions. In: TCC. pp. 537–566. Springer (2017)
55. Halevi, S., Hazay, C., Polychroniadou, A., Venkitasubramaniam, M.: Round-Optimal Secure Multi-Party Computation. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 488–520. Springer International Publishing, Cham (2018)
56. Harnik, D., Kilian, J., Naor, M., Reingold, O., Rosen, A.: On Robust Combiners for Oblivious Transfer and Other Primitives. In: EUROCRYPT (2005)
57. Hemenway, B., Jafargholi, Z., Ostrovsky, R., Scafuro, A., Wichs, D.: Adaptively secure garbled circuits from one-way functions. In: CRYPTO (2016)
58. Herzberg, A.: On Tolerant Cryptographic Constructions. In: CT-RSA (2005)
59. Herzberg, A.: Folklore, practice and theory of robust combiners. Journal of Computer Security **17**(2), 159–189 (2009)
60. Hu, Y., Jia, H.: Cryptanalysis of GGH Map. IACR Cryptology ePrint Archive **2015**, 301 (2015)
61. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing. pp. 21–30. ACM (2007)
62. Kitagawa, F., Nishimaki, R., Tanaka, K.: Obfustopia built on secret-key functional encryption. In: EUROCRYPT. pp. 603–648 (2018)
63. Komargodski, I., Segev, G.: From minicrypt to obfustopia via private-key functional encryption. In: EUROCRYPT (2017)
64. Levin, L.A.: One-way functions and pseudorandom generators. Combinatorica (1987)
65. Lin, H.: Indistinguishability obfuscation from constant-degree graded encoding schemes. In: EUROCRYPT (2016)
66. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: CRYPTO (2017)
67. Lin, H., Matt, C.: Pseudo Flawed-Smudging Generators and Their Application to Indistinguishability Obfuscation. Cryptology ePrint Archive, Report 2018/646 (2018), https://eprint.iacr.org/2018/646
68. Lin, H., Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation with non-trivial efficiency. In: PKC (2016)
69. Lin, H., Pass, R., Seth, K., Telang, S.: Output-compressing randomized encodings and applications. In: TCC (2016)
70. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In: CRYPTO (2017)
71. Lin, H., Vaikuntanathan, V.: Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In: FOCS (2016)
72. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: EUROCRYPT (2016)
73. O'Neill, A.: Definitional issues in functional encryption. IACR Cryptology ePrint Archive **2010**, 556 (2010)

74. Peikert, C., Shiehian, S.: Multi-key FHE from LWE, revisited. In: TCC (2016)
75. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: CRYPTO (2008)
76. Quach, W., Wee, H., Wichs, D.: Laconic Function Evaluation and Applications. Cryptology ePrint Archive, Report 2018/409 (2018), https://eprint.iacr.org/2018/409
77. Sahai, A., Seyalioglu, H.: Worry-free encryption: functional encryption with public keys. In: CCS. pp. 463–472. ACM (2010)
78. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: EUROCRYPT (2005). https://doi.org/10.1007/11426639_27
79. Yao, A.C.C.: How to Generate and Exchange Secrets (Extended Abstract). In: FOCS. pp. 162–167 (1986)