Multi-input Laconic Function Evaluation

Bo Pang[1,2,3], Long Chen[3], Xiong Fan[4], and Qiang Tang[3(✉)]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
pangbo@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] New Jersey Institute of Technology, Newark, USA
{longchen,qiang}@njit.edu
[4] University of Maryland, College Park, USA
xfan@cs.umd.edu

Abstract. Recently, Quach, Wee and Wichs (FOCS 2018) proposed a new powerful cryptographic primitive called *laconic function evaluation* (LFE). Using an LFE scheme, 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. The laconic property requires that the size of the digest, the runtime of the encryption algorithm and the size of the ciphertext should be much smaller than the circuit-size of f. This new tool is motivated by an interesting application of "Bob-optimized" two-round secure two-party computation (2PC). In such a 2PC, Alice will get the final result thus the workload of Bob will be minimized.

In this paper, we consider a "client-optimized" two-round secure multiparty computation, in which multiple clients provide inputs and enable a server to obtain final outputs while protecting privacy of each individual input. More importantly, we would also minimize the cost of each client. For this purpose, we propose multi-input laconic function evaluation (MI-LFE), and give a systematic study of it.

It turns out that MI-LFE for general circuit is not easy. Specifically, we first show that the directly generalized version, i.e., the public-key MI-LFE implies virtual black-box obfuscation. Hence the public-key MI-LFE (for general circuits) is infeasible. This forces us to turn to secret key version of MI-LFE, in which encryption now needs to take a secret key. Next we show that secret-key MI-LFE also implies heavy cryptographic primitives including witness encryption for NP language and the indistinguishability obfuscation. On the positive side, we show that the secret-key MI-LFE can be constructed assuming indistinguishability obfuscation and learning with errors assumption. Our theoretical results suggest that we may have to explore relaxed versions of MI-LFE for meaningful new applications of "client-optimized" MPC and others.

Keywords: Multi-party computation · Laconic function evaluation · Indistinguishability obfuscation

J. K. Liu and H. Cui (Eds.): ACISP 2020, LNCS 12248, pp. 369–388, 2020.
https://doi.org/10.1007/978-3-030-55304-3_19

1 Introduction

In a recent paper [31], Quach, Wee and Wichs described an interesting secure two-round two-party computation (2PC) protocol which is "Bob-optimized". In such a protocol, Alice and Bob who have inputs x_A and x_B respectively want to jointly compute $f(x_A, x_B)$, and Alice initiates the first round message, and learns the output $f(x_A, x_B)$ in the second round. More interestingly, Alice does all the work while Bob's computation and communication during the protocol execution are both smaller than the size of the circuit f or even Alices input x_A, (concretely, the computational cost of Bob is only $(|x_B| + |f(x_A, x_B)|) \cdot \mathsf{poly}(d)$, where d is the depth of the circuit f). Such kind of "Bob-optimized" secure 2PC was considered more natural as it is Alice who obtains the output should do the work [31]. This is in contrast to prior solutions based on fully homomorphic encryption [12,13,21,23] which optimized the work of Alice.

To construct such kind of two-round "Bob-optimized" secure 2PC, a new cryptographic primitive *laconic function evaluation* (LFE) was formulated in [31]. In an LFE scheme, Alice can compress a potentially large circuit f into a small digest. Bob then can encrypt some data x under this digest s.t. Alice can recover $f(x)$ without learning anything else about the original data x. The size of the digest, the run-time of the encryption algorithm and the size of the ciphertext should all be much smaller than the size of circuit f, In this way, Bob's workload in the 2PC is minimized. In [31], they provided the first construction of LFE for general circuits under the learning with errors (LWE) assumption.

"Client-Optimized" 2-Round MPC. The "Bob-optimized" two-round 2PC is useful in many applications such as privacy preserving data analytics, especially when the client device (Bob in the above setting) is resource restrained (e.g., mobile devices), while the circuit representing the analytic function is substantially complex (e.g., some complicated data mining or machine learning algorithms). However, in many relevant scenarios, the data of "Bob" may not be generated all at once, or even come from multiple clients. Consider the following scenarios:

Privacy Preserving Data Analytic System. Many data analytic applications running in a server solicit input data via multiple data collectors. For instance, many surveillance cameras are now deployed on the roads to monitor traffic conditions by local governments, and the videos are collected and submitted to a central server to analyze the road condition.

It is not hard to see that those individual inputs may be sensitive, e.g., the videos could contain confidential geographic information regarding the cars traveling on the roads, thus they should not be directly submitted to the server in the plain. However, there are multiple clients (the cameras) to provide inputs, we need now to deploy a multiparty computation protocol. On the other hand, similar as before, those clients (cameras) are not powerful computing devices, thus the computation and communication cost on the cameras (as data collectors) have to be minimized. And it is not realistic to ask all the cameras to coordinate other than directly communicating with a server.

Privacy Preserving Call Records Surveillance. It is known that several major telephone companies were cooperating with the intelligent service to monitor the phone records of U.S. citizens, and store them in a large database known [1,2]. Though it is a constant debate that sometimes surveillance may be needed for law enforcement to do investigations, it is obviously unacceptable that personal call records are uploaded and stored in the clear. It would be necessary to design a privacy preserving call records surveillance system.

In such a setting, each client continuously uploads obfuscated call records, and the mobile phones are mostly not so powerful devices that would demand the optimization on the client side workload for a privacy preserving protocol. Moreover, since investigation may apply complex data mining algorithms on multiple pieces of call records from each targeted individual, similar as above, we would need to design a 2-round client-optimized multiparty (depending how many pieces of records needed) computation.[1]

"Client-Optimized" 2-Round MPC. To capture above two exemplary application scenarios, we need a 2-round "client-optimized" secure multi-party computation protocol in which there might be multiple inputs from multiple clients. [2] In particular, we wish to have a 2-round protocol, that the server initializes the protocol with a first round message, then each client (the same client in different time period sending a different input would be viewed as a different client) sends out a message and then the server obtains the final output $f(x_1, \ldots, x_n)$, where f is the analytic function and $x_1, \ldots, x_n$ are all the inputs. More importantly, the "client-optimized" property here refers to (1) the computation and communication of each client is as efficient as that of Bob in the "Bob-optimized" 2PC; and (2) there is no communication among each clients.

Insufficiencies of Existing Tools. There are several possible paths to proceed, unfortunately none of them reaches a satisfying solution. Let us analyze one by one. (1) The second property above disallowing communication among clients already excludes straightforward solutions such as the general multiparty computation protocol [4,6,30,32], let alone it is not clear how to ensure the low client costs. (2) Laconic functional evaluation [31] was also shown to be applicable to MPC with small online computation. However, in our setting, the similar idea letting the server first compress the function to obtain a digest, then all clients and the server run an MPC protocol for the encryption function of the laconic function evaluation scheme (with the server input of the digest) is not satisfying. Although there exist 2-round MPC protocol [20], the communications among clients could be potentially large, let alone in some of the scenarios,

[1] One may suggest to let the client wait and upload a bunch of call records all at once, however, each individual has no incentive to do so and this is not how call records stored nowadays.

[2] In this paper, we only consider semi-honest security for our MPC application, which is analogous to the Bob-optimized 2PC in the LFE paper. In this setting, we assume the server will only choose a proper function permitted by clients. Furthermore, a semi-honest security protocol can be easily upgraded to adaptively secure by standard techniques such as adding NIZK.

the clients may not be able to talk to each other. (3) Another related notion is functional encryption [10,18,26], especially multi-input functional encryption (MIFE) [9,24]. Straightforward application of MIFE requires an extra trusted party to generate the decryption key for the server (which is already a huge overload that is proportional to the size of f), while we cannot let the server to do this, otherwise no security of the inputs can be present. (4) Last but not least, similar as in [31], multi-key fully homomorphic encryption [15,30] cannot enable the server to learn the output in two rounds.

Formalizing Multi-input Laconic Function Evaluation. Motivated by the client-optimized 2-round MPC application and the deficiencies of existing tools, we generalize the notion of LFE to *multi-input laconic function evaluation* (MI-LFE). In a MI-LFE scheme, the server has a large circuit f defined over n inputs, which can be deterministically compressed into a short digest $\mathsf{digest}_f = \mathsf{Compress}(f)$. Then each client i can encrypt his input data x_i under this digest, resulting in a ciphertext $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{digest}_f, x_i)$, respectively. After receiving n ciphertexts, the server is then able to decrypt using her knowledge of f to recover the output $f(x_1, \ldots, x_n) = \mathsf{Dec}(f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$. Security ensures that the server does not learn anything else about the n inputs $x_1, \ldots, x_n$ beyond the output $f(x_1, \ldots, x_n)$, as formalized via the simulation paradigm. Similar to LFE, the laconic property of MI-LFE requires that each client's computation and communication complexity is small, and in particular, the size of the digest digest_f, the run-time of the encryption algorithm $\mathsf{Enc}(\mathsf{digest}_f, x_i)$ and the size of the ciphertext ct_i should be much smaller than the circuit-size of f.

With the new primitive at hand, the client-optimized 2-round MPC protocol can be easy: the server compresses the function and broadcasts the digest to all clients. Each client then uses the encryption algorithm to obfuscate his input and sends it to the server. The server then pools all ciphertext and evaluate. The laconic property guarantees that the workload of each client is small compared to the complex function f, and no communication is needed among clients.

The Difficulty of Constructing MI-LFE. We then systematically study the concept of MI-LFE. It turns out that such a notion is quite difficult to obtain. In particular, the most nature model for MI-LFE is in the public-key setting which generalizes LFE in [31] in a straightforward fashion. The only difference is that the function here is evaluated on multiple inputs, so the different inputs x_i for $i = 1, \ldots, n$ are encrypted into different ciphertexts ct_i for $i = 1, \ldots, n$, and the decryption procedure involves multiple ciphertexts. In Sect. 4, we show that public key MI-LFE actually implies virtual black-box obfuscation. Since VBB obfuscation is known to be impossible for general circuits [5], this yields us impossibility results for (general) MI-LFE in the public setting.

To circumvent such an impossibility but still enabling the client-optimized MPC, we turn to MI-LFE in the private-key setting. It follows the syntax of the version of the public-key setting, but an additional key generation procedure is involved and the encryption procedure will always take the private key as

input.[3] Restricting encryption procedure allows us to bypass the implication to VBB that essentially compresses a universal circuit and uses MI-LFE encryption algorithm to encrypt the function and input as different ciphertext to evaluate.

Then we find that even in the private key setting, constructing MI-LFE from standard assumptions is not an easy task. We show that the private-key MI-LFE with reasonable security definition implies the witness encryption (WE) for NP language [3,7,11,17,22] and the indistinguishability obfuscation (iO) for general circuits [14,16,28,29], respectively. Since these two advanced primitives have no constructions from the standard assumptions so far, leveraging some heavy tools in the construction of MI-LFE seems inevitable. As a byproduct, we notice that MI-LFE also implies MIFE, thus MI-LFE could be applicable in multiple advanced scenarios, if it ever exists. But the reverse implication is not straightforward, since MIFE do not have the compression property.

Constructing Private-Key MI-LFE. Next we show that the private-key MI-LFE can indeed be constructed from indistinguishability obfuscation and learning with errors (LWE) assumption. Our construction of MI-LFE is inspired by the techniques developed in the context of multi-input functional encryption [25] and laconic function evaluation [31]. Intuitively, the message x_i for i-th coordinate of circuit C in MI-LFE is encrypted using public key encryption and equipped with a proof showing that this encryption is done correctly. The other part of the i-th ciphertext is an indistinguishability obfuscation of a circuit that first checks the legitimacy of the input ciphertexts and then transforms public key encryptions of messages $\{x_i\}$ into an LFE ciphertext of the message $x_1, \ldots, x_n$. A decryptor in MI-LFE that has all ciphertexts of messages $\{x_i\}$ for a pre-compressed circuit C first obtains LFE encryption of messages for each coordinate of circuit C after evaluating the obfuscated program, and then the actual result $C(\{x_i\})$ by evaluating the decryption algorithm in LFE.

The laconic property of our scheme follows from the laconic property of LFE encryption algorithm. Only the LFE encryption procedure in the indistinguishable obfuscation is corresponding to the circuit C, and other parts in our encryption procedure is independent with the circuit C, we know that LFE encryption only scales with the circuit depth, much smaller than the circuit size. Even after been obfuscated, should still much smaller than the circuit size.[4]

The security definition of MI-LFE requires to simulate the challenge ciphertexts. Recall that there are two parts in a ciphertext: (1) two independent semantically secure public key encryption of the actual message and a proof showing the correctness and legitimacy of the encryption; (2) an obfuscation of a circuit

[3] We remark here that a secret key MI-LFE is enough for many of our applications of client-optimized 2-round MPC: if for one client generating inputs overtime, the client's device may have a pre-installed secret key by the device manufacturer; while for data analytics via multiple data collectors such as cameras, a secret key maybe installed by the government who deploy them.

[4] we consider an unbounded-size circuit class $\mathcal{C}$, meaning that there exists a large size circuit $C \in \mathcal{C}$. We refer to Lemma 3 for the details of laconic property of our construction.

that contains proofs verification and encryption transformation. Specifically, the simulation of first part relies on semantic security of the public-key encryption and special property of the proof system, i.e. witness indistinguishability. The circuit to be obfuscated can be changed according to changes happened in the first part of challenge ciphertext. We also would like to remark that in our construction, we simply rely on a common random string, rather than a common reference string used previously in MIFE.

1.1 Additional Related Works

Multi-input Functional Encryption. MI-LFE also appears to be related to multi-input functional encryption (MIFE) [9,24]. In MIFE, multiple ciphertexts are also encrypted independently by different parties, while the one holding the decryption key sk_f can only recover $f(x_1, \ldots, x_n)$ without learning anything else about $x_1, \ldots, x_n$. Despite of the similarities, a MIFE scheme involves a master authority whose duty is to derive a decryption key sk_f respect to a function f from the master key msk. More importantly, the complexity of the decryption key generation procedure may be proportion to the size of the function f. Note that in some application scenario like the one mention above, there are no parties that can be both trustworthy and capable to afford such workload.

Fully Homomorphic Encryption. FHE [12,13,21,23] can be viewed as the dual version of the LFE. In FHE, one party, say Alice, can encrypt different values x_i, resulting in a ciphertext $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, x_i)$, respectively. Then another party, say Bob, can homomorphically evaluate a function f on these ciphertexts. When Alice sees the evaluation, she can decrypt and recover the message $f(x_1, \ldots, x_n)$. In contrast with MI-LFE and LFE, here it is the encryptor to get the final result.

General MPC. Since our application is a special case of secure multi party computation, theoretically it can be realized in two round by the general MPC technique [4,6,32]. However, here we additionally require that the computation complexity for each surveillance to be cheap, and they only need to communicate with the data center but not with each other. Considering all these additional requirements, this application is hard to be directly achievable via general MPC. There exist 2-round MPC protocol [20,30], the communications cost among clients could be potentially large, i.e. an multiplicative overhead of the depth of circuit and size of input.

2 Preliminaries

In this section, we give background on two classical cryptographic primitives used in paper: non-interactive witness-indistinguishable proofs and perfectly binding commitments.

Non-interactive Proof Systems. Here, we recall the syntax and property of non-interactive witness-indistinguishable proofs.

Syntax. Let R be an efficiently computable binary relation. For pairs $(x, w) \in R$, we call x the statement and w the witness. Let L be the language consisting of statements in R. A non-interactive proof system [8,27] for a language L, consists of a CRS generation algorithm CRSGen, a proving algorithm Prove and a verification algorithm Verify, defined as follows:

- $\mathsf{CRSGen}(1^\lambda)$: On input the security parameter λ, it outputs a common reference string crs.
- $\mathsf{Prove}(\mathsf{crs}, x, w)$: On input the common reference string crs, and a statement x along with a witness w. If $(x, w) \in R$, it produces a proof string π, otherwise it outputs **fail**.
- $\mathsf{Verify}(\mathsf{crs}, x, \pi)$: On input the common reference string crs, and a statement x along with a proof string π, it outputs 1 if the proof is valid, and 0 otherwise.

Definition 1 (Non-interactive Proof System). *A non-interactive proof system for a language L with a* PPT *relation R is a tuple of algorithms* $(\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify})$ *such that the following properties hold:*

- ***Perfect Completeness.*** *A proof system is complete if an honest prover with a valid witness can convince an honest verifier. More formally, for all $x \in [L]$ and every w such that $(x, w) \in R$, it holds that*

$$\Pr[\mathsf{Verify}(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w)) = 1] = 1$$

  where $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^\lambda)$ *and the probability is taken over the coins of* CRSGen, Prove *and* Verify.
- ***Statistical Soundness.*** *A proof system is sound if it is infeasible to convince an honest verifier when the statement is false. More formally, for all adversary (even unbounded) $\mathcal{A}$, it holds that*

$$\Pr[\mathsf{Verify}(\mathsf{crs}, x.\pi) = 1 \wedge x \notin L | \mathsf{crs} \leftarrow \mathsf{CRSGen}(1^\lambda), (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs})] = \mathsf{negl}(\lambda)$$

Definition 2 (NIWI)**.** *A non-interactive proof system* $(\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify})$ *for a language L with a* PPT *relation R is witness indistinguishable if for any triplet (x, w_0, w_1) such that $(x, w_0) \in R$ and $(x, w_1) \in R$, the distributions* $\{\mathsf{crs}, \mathsf{Prove}(\mathsf{crs}, x, w_o)\}$ *and* $\{\mathsf{crs}, \mathsf{Prove}(\mathsf{crs}, x, w_1)\}$ *are computationally indistinguishable, where* $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^\lambda)$.

Non-interactive and Perfectly Binding Commitment Schemes. We let $\mathsf{Com}(\cdot;\cdot)$ denote the commitment function of a non-interactive commitment scheme. Com is a PPT algorithm that takes as input a string x and randomness r, and outputs $c \leftarrow \mathsf{Com}(x; r)$. A perfectly binding commitment scheme must satisfy the following properties:

- **Perfectly Binding.** This property states that the two different strings cannot have the same commitment. More formally, $\forall x_1 \neq x_2$, $\mathsf{Com}(x_1) \neq \mathsf{Com}(x_2)$.
- **Computationally Hiding.** For all strings x_0 and x_1 (of the same length), and all PPT adversaries $\mathcal{A}$, we have that:

$$|\Pr[\mathcal{A}(\mathsf{Com}(x_0)) = 1] - \Pr[\mathcal{A}(\mathsf{Com}(x_1)) = 1]| \leq \mathsf{negl}(\lambda)$$

3 Multi-input LFE: Syntax and Security Definition

In this section, we define the notion of multi-input laconic function evaluation (MI-LFE) for a class of n-ary circuits $\mathcal{C}$. We assume that every circuit $C \in \mathcal{C}$ is associated with some circuit parameters $C.\mathsf{params}$. By default we consider $\mathcal{C}$ to be the class of all circuits with $C.\mathsf{params} = (1^n, 1^k, 1^d)$, which take as input n bit-strings $(x_1, \ldots, x_n)$, where $x_i \in \{0,1\}^k$, and d denotes the circuit depth.

Definition 3 (MI-LFE). *A private-key multi-input laconic function evaluation for circuits class $\mathcal{C}$ consists of five algorithms* $(\mathsf{crsGen}, \mathsf{KeyGen}, \mathsf{Compress}, \mathsf{Enc}, \mathsf{Dec})$ *with details as follows:*

- $\mathsf{crsGen}(1^\lambda, C.\mathsf{params})$ *takes as input the security parameter 1^λ and circuit parameters $C.\mathsf{params}$, and outputs a uniformly random common random string* crs *of appropriate length.*
- $\mathsf{KeyGen}(1^\lambda, \mathsf{crs})$ *takes as input the security parameter 1^λ and the common random string* crs*, and outputs a private key* SK*.*
- $\mathsf{Compress}(\mathsf{crs}, C)$ *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*.*
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, i, x_i)$ *takes as input the common random string* crs*, a digest* digest_C*, a private key* SK*, an index i, and a message x_i, and outputs a ciphertext* ct_i*.*
- $\mathsf{Dec}(\mathsf{crs}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ *takes as input the common random string* crs*, a circuit $C \in \mathcal{C}$, and n ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$*, and outputs a message y.*

Correctness. For correctness, we require that for all λ and $C \in \mathcal{C}$ with $C.\mathsf{params}$, it holds that

$$\Pr\left[y = C(x_1, \ldots, x_n) \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{crsGen}(1^\lambda, C.\mathsf{params}), \\ \mathsf{SK} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{crs}), \\ \mathsf{digest}_C \leftarrow \mathsf{Compress}(\mathsf{crs}, C), \\ \mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, i, x_i), \\ y \leftarrow \mathsf{Dec}(\mathsf{crs}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_n), \end{array} \right] = 1$$

Definition 4 (SIM-Based Security). *For security, we say a private-key* MI-LFE *is (n,q)-*SIM*-secure, where n denotes the number of input strings for a circuit C, and q is number of challenge message tuples, if there exists a* PPT *simulator* SIM *such that for all stateful* PPT *adversary $\mathcal{A}$, it holds:*

$$\left| \Pr[\mathsf{Expt}^{\mathsf{Real}}_{\mathsf{MI\text{-}LFE}}(1^\lambda) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Ideal}}_{\mathsf{MI\text{-}LFE}}(1^\lambda) = 1] \right| \leq \mathsf{negl}(\lambda)$$

where the experiments $\mathsf{Expt}^{\mathsf{Real}}_{\mathsf{MI\text{-}LFE}}(1^\lambda)$ and $\mathsf{Expt}^{\mathsf{Ideal}}_{\mathsf{MI\text{-}LFE}}(1^\lambda)$ are defined in Fig. 1.

In Fig. 1, oracle $O(C, \cdot)$ denotes the trusted party. It accepts queries of the form $(j_1, \ldots, j_n)$, where $j_1, \ldots, j_n \in \{1, \ldots, q\}$. On input such a query, $O(C, \cdot)$ outputs the $C(x_1^{j_1}, \ldots, x_n^{j_n})$; otherwise outputs $\perp$. We refer to the above as adaptive security.

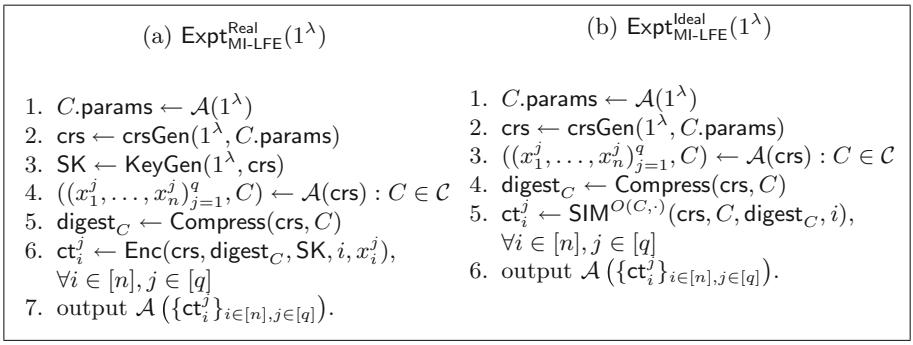

Fig. 1. SIM-based Security Experiments for MI-LFE

Definition 5 (IND-Based Security). *We say a private-key* MI-LFE *is* (n, q)-IND*-secure, where* n *denotes the number of inputs for a circuit* C*, and* q *is the number of* n*-ary challenge message tuples, if for any* PPT *adversary* $\mathcal{A}$*,*

$$\mathsf{Adv}^{\mathrm{IND}}_{\mathcal{A},\mathsf{MI\text{-}LFE}}(1^\lambda) = \left|\Pr[\mathsf{Expt}^{\mathrm{IND}}_{\mathcal{A},\mathsf{MI\text{-}LFE}}(1^\lambda) = 1] - \frac{1}{2}\right|$$

is $\mathsf{negl}(\lambda)$*, where the experiments* $\mathsf{Expt}^{\mathrm{IND}}_{\mathcal{A},\mathsf{MI\text{-}LFE}}(1^\lambda)$ *is defined in Fig. 2.*

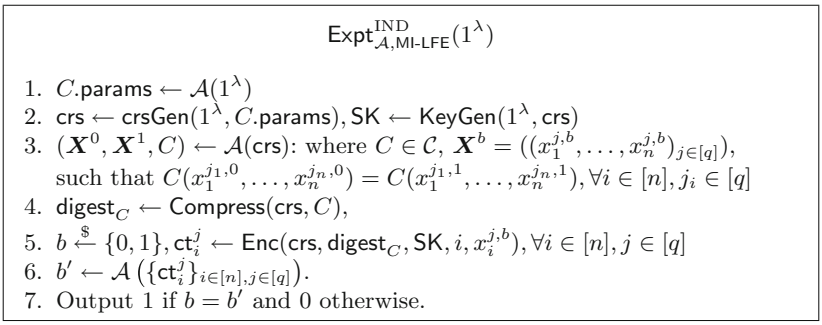

Fig. 2. IND-based Security Experiment For MI-LFE

Remark 1 (Selective Security). A weaker notion, called selective security, in both SIM-based and IND-based security, can be defines as: adversary $\mathcal{A}$ has to choose challenge plaintext at the very beginning of the experiments (as described in Fig. 1 and Fig. 2) before seeing crs.

Lemma 1. *If a MI-LFE scheme* Π *is SIM secure, then* Π *is also IND secure.*

Proof (Proof Sketch). We note that if a MI-LFE scheme Π satisfies SIM-based security, then we show that it also satisfies IND-based security. Now, for challenge message queries $(\boldsymbol{X}^0, \boldsymbol{X}^1)$, where $\boldsymbol{X}^b = \{(x_1^{j,b}, \ldots, x_n^{j,b})\}_{j\in[q]}$, the challenger in the IND-based security experiment chooses a random bit b, then invokes the simulator in SIM-based security game to compute

$$\mathsf{ct}_i^j \leftarrow \mathsf{SIM}^{O(C,\cdot)}(\mathsf{crs}, C, \mathsf{digest}_C, i), \quad \forall i \in [n], j \in [q]$$

where $O(C, \cdot)$ accepts the queries of the form $(j_1, \ldots, j_n)$ and outputs $C(x_1^{j_1,b}, \ldots, x_n^{j_n,b})$. Hence, by the SIM-based security, for all $i \in [n], j \in [q]$, each respond $\mathsf{ct}_i^j \leftarrow \mathsf{SIM}^{O(C,\cdot)}(\mathsf{crs}, C, \mathsf{digest}_C, i)$ is computationally indistinguishable from real execution $\mathsf{ct}_i^j \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, i, x_i^{j,b})$. And the bit b is chosen from random, this completes the IND-based security experiment for $\mathsf{MI\text{-}LFE}$. Since $(\boldsymbol{X}^0, \boldsymbol{X}^1)$ satisfies $C(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = C(x_1^{j_1,1}, \ldots, x_n^{j_n,1}), \forall i \in [n], j_i \in [q]$, we have that $\{ct_i^j\}$, output by the challenger, is independent with the bit b.

Laconic property. Same as LFE, we insist that the size of $(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, \mathsf{ct}_i)$ and the running time of Enc are at most sublinear of the size of circuit C.

4 Hardness of MI-LFE

In this section, we show the difficulty of MI-LFE schemes. In particular, we show public-key MI-LFE for general circuits is impossible by constructing a virtual black-box obfuscator from it. Moreover, even the private-key MI-LFE implies witness encryption and indistinguishability obfuscation.

Public-Key MI-LFE. We first discuss the syntax and security definition of public key MI-LFE. Then, we show that a MI-LFE scheme for all circuits implies virtual black-box obfuscation for all circuits, which is proved impossible by Barak et al. [5]. The main difference between MI-LFE in public-key setting and that in private-key setting (c.f. Sect. 3) is that algorithm $\mathsf{KeyGen}(1^\lambda, \mathsf{crs})$ does not exist in public-key setting, meaning that encryption algorithm can be performed by anyone who knows common reference string and the digest of circuit.

Security Definition. In this part, we discuss the intuition of simulation-based public key MI-LFE security. To illustrate the difference between public-key and private-key setting, it suffices to consider the case of 2-ary functions and *one* challenge message tuple (x_1, x_2). In this example, the simulation-based security in the private-key setting guarantees that for one function f, an adversary cannot learn anything more than $f(x_1, x_2)$ where (x_1, x_2) is the challenge message pair. However, its counterpart in the public-key setting cannot guarantee this property. The reason is that an adversary who knows the public key can create its own chiphertexts, thus can learn additional information $\{f(x_1, \cdot)\}$ and $\{f(\cdot, x_2)\}$ given ciphertexts for (x_1, x_2). This additional information must be taken into account for the simulator (adversary in ideal world) in the ideal world. We refer to the full version for the formal SIM-based security game.

VBB Obfuscation from Public-Key MI-LFE. Here we show that a virtual black-box obfuscator [5], can be derived from a two party public-key MI-LFE. The basic idea is to let the compressed function of the MI-LFE be a universal circuit U. The input of the first party is the function f which we wish to obfuscate, and the input of the second party is the input value x of the function f. So the obfuscator should output the first party's ciphertext $\mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_U, 1, f)$.

Specifically, for a universal circuit U, the obfuscator **VBB** works as follow:

1. **Obfuscation:** Run the crsGen and $\mathsf{Compress}$ algorithm to generate crs and digest_U, compute $\mathsf{ct}_1 \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_U, 1, f)$, and output the obfuscated circuit $\mathbf{VBB}(f) = (\mathsf{crs}, \mathsf{digest}_U, U, \mathsf{ct}_1)$.
2. **Evaluation:** To evaluate the obfuscated circuit $\mathbf{VBB}(f)$ on an input x, one just needs to compute $\mathsf{ct}_2 \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_U, 2, x)$ and run $\mathsf{Dec}(\mathsf{crs}, U, \mathsf{ct}_1, \mathsf{ct}_2)$.

According to the correctness of MI-LFE, the decryption result should be $U(f, x) = f(x)$. The virtual black-box property of this obfuscator follows from the simulation security of the MI-LFE, hence we have the following theorem.

Theorem 1. *A $(2, 1)$-SIM-secure MI-LFE in public-encryption setting for general 2-ary functions implies virtual black-box obfuscation for all circuits.*

Given an VBB adversary $\mathcal{A}$, we use A to construct an MI-LFE adversary $\mathcal{B}$, the full proof is given in full version.

Witness Encryption from Private-Key MI-LFE. Since public key MI-LFE for general circuits does not exist, we have to turn our attention to private key MI-LFE. We firstly introduce MI-ABLFE (a variant of private-key MI-LFE). Then, we construct witness encryption for NP language, for general circuits. Since MI-LFE trivially implies an MI-ABLFE, thus we conclude the implication of private key MI-LFE to witness encryption, introduced by Garg et al. [19].

MI-ABLFE. We start from LFE for a restricted class of functionalities, which call attribute-based LFE (AB-LFE) in analogy to attribute-based encryption [31]. We formalize the definition and security requirement of MI-ABLFE as follow:

Definition 6 (MI-ABLFE). *Let $C : (\{0,1\}^k)^n \to \{0,1\}$ be a circuit. We define the Conditional Disclosure Functionality (CDF) of C as the function*

$$\mathsf{CDF}[C]\left((x_1, u), \ldots, (x_n, u)\right) = \begin{cases} (x_1, \ldots, x_n, u) & \text{if } C(x_1, \ldots, x_n) = 1 \\ (x_1, \ldots, x_n, \perp) & \text{if } C(x_1, \ldots, x_n) = 0 \end{cases}$$

where $x_i \in \{0,1\}^k$, and $u \in \{0,1\}$.

A MI-ABLFE scheme for a circuit family $\mathcal{C}$ is a MI-LFE scheme that supports circuits $CDF[C]$ defined as above, for all $C \in \mathcal{C}$. We define $CDF[C].params = C.params = (1^n, 1^k, 1^d)$, where d is the depth of C.

Remark 2 (Security of MI-ABLFE). The IND-based security notion of MI-ABLFE can be defined similarly as MI-LFE, except for the difference that the payload u remains private if for any message queries (x_i, u), it holds that $C(x_1, \ldots, x_n) = 0$.

Witness Encryption from MI-ABLFE. Intuitively, the witness encryption [19] can use the MI-ABLFE in the following way: the general circuit C is used as the NP verifier such that the decryptor of MI-ABLFE can recover the message u if he has the witness w for the statement x satisfying $C(x, w) = 1$. More specificially, given an NP language L, the construction of $\Pi = (\mathsf{Enc}, \mathsf{Dec})$ for L is as follows:

- $\mathsf{Enc}(1^\lambda, x, u)$: On input a statement $x \in \{0,1\}^n$ (whose witness has length bounded by m), and a message $u \in \{0,1\}$, the executes the following:
  1. Set $C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ to be the NP verifier for language L that takes as the input $x \in \{0,1\}^n$, $w \in \{0,1\}^m$, and outputs 1 iff $(x, w) \in L$. Compute $\mathsf{MI\text{-}ABLFE.crsGen}(1^\lambda, C.\mathsf{params})$ to generate CRS string crs.
  2. Then it runs $\mathsf{MI\text{-}ABLFE.Compress}(\mathsf{crs}, C)$ to generate digest_C, and $\mathsf{MI\text{-}ABLFE.KeyGen}(\mathsf{crs})$ to get SK.
  3. For $i \in [n]$, compute $\mathsf{MI\text{-}ABLFE.Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, i, x_i, u)$, where x_i is the i-th bit of x.
  4. For $b \in \{0,1\}, j \in [m]$, compute $\mathsf{MI\text{-}ABLFE.Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, j, b, u), \forall b \in \{0,1\}, j \in [m]$.

  Output $\mathsf{ct} = \big(\mathsf{crs}, C, x, \{\mathsf{ct}_i\}_{i\in[n]}, \{\mathsf{ct}_{j,b}\}_{j\in[m],b\in\{0,1\}}\big)$.
- $\mathsf{Dec}(\mathsf{ct}, w)$: On input a witness $w \in \{0,1\}^m$ for the statement $x \in \{0,1\}^n$, and a ciphertext ct for x, the decryption algorithm computes and outputs

$$\mathsf{MI\text{-}ABLFE.Dec}\left(\mathsf{crs}, C, \{\mathsf{ct}_i\}_{i\in[n]}, \{\mathsf{ct}_{j,w_j}\}_{j\in[m]}\right)$$

  where w_j is the j-th bit of witness w.

The correctness of the witness encryption follows from the correctness of the MI-ABLFE.

Theorem 2. *Assuming the $(n + m, 2)$-IND-based selective security of MI-ABLFE scheme* $\mathsf{MI\text{-}ABLFE}$ *for general circuits, then the witness encryption scheme* Π *described above is secure.*

Given an witness encryption adversary $\mathcal{A}$, we describe an MI-LFE adversary $\mathcal{B}$ invoke A as a subroutine to attack the security of MI-LFE. The proof is completed in full version.

Indistinguishable Obfuscation from Private-Key MI-LFE. We can derive an indistinguishability obfuscator for all circuits [16], with k-bit inputs from a $(k+1)$-party MI-LFE in private-key setting. This, in particular, means that the use of indistinguishable obfuscation is inevitable for the private-key MI-LFE.

Now, we describe how to construct an indistinguishable obfuscator for a circuit class $\mathcal{C}$, where for every $C \in \mathcal{C}$, $C : \{0,1\}^k \to \{0,1\}^{k'}$ and $|C| = \ell$. Assuming there is a $(k + 1, 2)$-IND-secure MI-LFE scheme $\mathsf{MI\text{-}LFE} = (\mathsf{crsGen}, \mathsf{KeyGen}, \mathsf{Compress}, \mathsf{Enc}, \mathsf{Dec})$, where $k + 1$ denotes the number of party and 2 denotes the number of challenge message tuples. The intuition here is to let the actual function to be evaluated in $\mathsf{MI\text{-}LFE}$ to be a universal circuit U, defined as following:

$$U(x_1, \ldots, x_k, C) = C(x_1, \ldots, x_k), \quad \forall i \in [k], x_i \in \{0,1\}, C \in \{0,1\}^\ell$$

For each party $i \in [k]$, the input is random bit $b \in \{0,1\}$. And the input of the final party is the description of circuit C, the circuit to be obfuscated. The indistinguishable obfuscation of circuit C is MI-LFE encryption of all possible inputs bit plus description of circuit C.

Specifically, the construction of $i\mathcal{O}$ is as follows:

- **Obfuscation:** On input circuit C, Run the $\mathsf{crsGen}(U.\mathsf{params})$ to generate crs, $\mathsf{KeyGen}(\mathsf{crs})$ to generate SK and $\mathsf{Compress}(\mathsf{crs}, U)$ to generate digest_U. Then for $i \in [k], b \in \{0,1\}$, compute: $\mathsf{ct}_i^b \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_U, \mathsf{SK}, i, b)$ and next compute
$$\mathsf{ct}_{k+1} \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_U, \mathsf{SK}, k+1, C)$$
Finally, output the obfuscated circuit as $i\mathcal{O}(C) = \left(\{\mathsf{ct}_i^b\}_{i\in[k],b\in\{0,1\}}, \mathsf{ct}_{k+1}, \mathsf{crs}, U\right)$.
- **Evaluation:** On input $x \in \{0,1\}^k$, evaluate the obfuscated circuit $i\mathcal{O}(C)$ as computing $\mathsf{Dec}(\mathsf{crs}, U, \mathsf{ct}_1^{x_1}, \ldots, \mathsf{ct}_k^{x_k}, \mathsf{ct}_{k+1})$.

The correctness of our $i\mathcal{O}$ construction directly follows that of MI-LFE scheme. By the correctness of $\mathsf{MI\text{-}LFE}$, the decryption result should be $U(x_1, \ldots, x_k, C) = C(x_1, \ldots, x_k)$.

Theorem 3. *Assuming MI-LFE is $(k+1, 2)$-IND-secure (c.f. Definition 5), then the above construction is a secure indistinguishability obfuscator for all circuits.*

Given an $i\mathcal{O}$ adversary $\mathcal{A}$, we use it to construct an MI-LFE adversary $\mathcal{B}$, the full proof is completed in full version.

MIFE from MI-LFE. We have shown that private-key MI-LFE implies iO. Since Goldwasser et al. have proved that MIFE can be constructed from the indistinguishable obfuscation and one-way function, the detour inspires us that MI-LFE can imply MIFE. However, one can imagine a more directly reduction from $n+1$ inputs private-key MI-LFE to n inputs private-key MIFE. Here we present the intuition. One can fix the circuit of MI-LFE as a universal circuit U. Given the description of a function f, we should have $U(f, x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$. The master secret key and the encryption key of MIFE are both the secret key of the MI-LFE scheme. The decryption key for function f of the MIFE is a MI-LFE ciphertext c_0 of the description of the function f respect to digest_U. The encryption of MIFE of message $x_1, \ldots, x_n$ are the MI-LFE ciphertexts $c_1, \ldots, c_n$ of the same messages respect to digest_U. The decryption of the MIFE is just the MI-LFE evaluation on the ciphetexts c_0 and $c_1, \ldots, c_n$. Such an implication alone is trivial, but as a byproduct, MI-LFE could be used for all the applications of MIFE, which might provide a new route for the special cases.

5 Constructing Private-Key MI-LFE

The components we use in the construction include: (1) an indistinguishable obfuscator $i\mathcal{O}$ [16](of polynomial p of its input size), (2) a NIWI proof system $(\mathsf{NIWI.crsGen}, \mathsf{NIWI.Prove}, \mathsf{NIWI.Verify})$, (3) a perfectly binding commitment scheme Com, (4) a semantically secure public-key encryption scheme

$\mathsf{PKE} = (\mathsf{PKE.Setup}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ and (5) a secure laconic function evaluation $\mathsf{LFE} = (\mathsf{LFE.crsGen}, \mathsf{LFE.Compress}, \mathsf{LFE.Enc}, \mathsf{LFE.Dec})$ [31].

We denote the length of ciphertext in PKE by ℓ_{ct}. In particular, for a circuit $C : (\{0,1\}^k)^n \to \{0,1\}^\ell$ of circuit size $|C| = \omega(p(n,k,\lambda))$ and depth d, the description of our MI-LFE construction is as follows:

- $\mathsf{crsGen}(1^\lambda, C.\mathsf{params} = (1^n, 1^k, 1^d))$: The CRS generation algorithm first computes a common random string $\mathsf{crs}_1 \leftarrow \mathsf{NIWI.crsGen}(1^\lambda)$ for the NIWI proof system. Next, it computes a common random string $\mathsf{crs}_2 \leftarrow \mathsf{LFE.crsGen}(1^\lambda, (1^{n\times k}, 1^d))$. The algorithm outputs $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2)$.
- $\mathsf{KeyGen}(1^\lambda, \mathsf{crs})$: The key generation algorithm first computes two key pairs $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{PKE.Setup}(1^\lambda)$ and $(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathsf{PKE.Setup}$. Then it computes the following commitments:

$$z_{1,i}^j \leftarrow \mathsf{Com}(0^{2\ell_{\mathsf{ct}}}), \forall i \in [n], j \in [q], \quad z_2 \leftarrow \mathsf{Com}(0; r_0)$$

  It outputs $\mathsf{SK} = (\mathsf{pk}_1, \mathsf{pk}_2, \mathsf{sk}_1, \{z_{1,i}^j\}, z_2, r_0)$, where r_0 is the randomness used to compute the commitment z_2.
- $\mathsf{Compress}(\mathsf{crs}, C)$: The deterministic algorithm compression runs and outputs $\mathsf{digest}_C \leftarrow \mathsf{LFE.Compress}(\mathsf{crs}_2, C)$.
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{digest}_C, \mathsf{SK}, i, x_i)$: On input crs, digest digest_C, secret key SK, index i and input x_i, the encryption algorithm.
  1. Choose two random strings $r_{i,1}, r_{i,2}$, and compute $c_{i,1} = \mathsf{PKE.Enc}(\mathsf{pk}_1, x_i; r_{i,1})$ and $c_{i,2} = \mathsf{PKE.Enc}(\mathsf{pk}_2, x_i; r_{i,2})$.
  2. Generate proof $\pi \leftarrow \mathsf{NIWI.Prove}(\mathsf{crs}_1, y, w)$ for statement $y = (c_{i,1}, c_{i,2}, \mathsf{pk}_1, \mathsf{pk}_2, \{z_{1,i}^j\}_{i\in[n], j\in[q]}, z_2)$:
     - Either $c_{i,1}$ and $c_{i,2}$ are encryptions of the same message and z_2 is a commitment to 0;
     - Or there exists $j \in \{1, \ldots, q\}$, such that $z_{1,i}^j$ is a commitment to $c_{i,1} \parallel c_{i,2}$.

     A witness $\omega_{\mathsf{real}} = (m, r_{i,1}, r_{i,2}, r_0)$ for the first part of the statement, referred as the real witness, includes the message m, and the randomness $r_{i,1}$ and $r_{i,2}$ used to compute the ciphertexts $c_{i,1}$ and $c_{i,2}$, respectively, and the randomness r_0 used to compute z_2. A witness $\omega_{\mathsf{td}} = (j, r_{1,i}^j)$ for the second part of the statement, referred as the trapdoor witness, includes an index j and the randomness $r_{1,i}^j$ used to compute $z_{1,i}^j$.
  3. Compute $i\mathcal{O}(G_{\mathsf{digest}_C})$, defined in Fig. 3.

  Output ciphertext $\mathsf{CT}_i = (c_{i,1}, c_{i,2}, \pi_i, i\mathcal{O}(G_{\mathsf{digset}_C}))$.
- $\mathsf{Dec}(\mathsf{crs}, C, \mathsf{CT}_1, \ldots, \mathsf{CT}_n)$: The decryption algorithm first runs

$$\mathsf{ct} \leftarrow i\mathcal{O}(G_{\mathsf{digset}_C})((c_{1,1}, c_{1,2}, \pi_1), \ldots, (c_{n,1}, c_{n,2}, \pi_n))$$

  Then it computes and outputs $y = \mathsf{LFE.Dec}(\mathsf{crs}_2, C, \mathsf{ct}')$.

Lemma 2 (Correctness). *Assuming the correctness of the underlying semantically secure* PKE*, laconic function evaluation* LFE *and indistinguishability obfuscation* $i\mathcal{O}$*, the completeness property of* NIWI*, then the construction above is correct.*

$G_{\mathsf{digest}_C}[\mathsf{SK}, \mathsf{digest}_C]((c_{1,1}, c_{1,2}, \pi_1), \ldots, (c_{n,1}, c_{n,2}, \pi_n))$

Input: PKE ciphertexts and proof pairs $(c_{i,1}, c_{i,2}, \pi_i)$, for $i \in [n]$.
Hardcoded: secret key SK and digest digest_C.

(a) For $i = 1$ to n, let $y_i = (c_{i,1}, c_{i,2}, \mathsf{pk}_1, \mathsf{pk}_2, \{z^j_{1,i}\}, z_2)$ be the statement associate with the proof string π_i. If $\mathsf{NIWI.Verify}(\mathsf{crs}_1, y_i, \pi_i) = 0$, then stop and output $\perp$; Otherwise continue to $i + 1$.
(b) Compute $x_i = \mathsf{PKE.Dec}(\mathsf{sk}_1, c_{i,1})$.
(c) Outputs $\mathsf{ct} \leftarrow \mathsf{LFE.Enc}(\mathsf{crs}_2, \mathsf{digest}_C, (x_1, \ldots, x_n))$.

Fig. 3. Description of circuit G_{digest_C}

Proof. Now, by the perfect completeness of NIWI, the honest encryption algorithm can use the real witness $\omega_{\mathsf{real}} = (m, r_{i,1}, r_{i,2}, r_0)$ to generate the proof string π_i, such that $\mathsf{NIWI.Verify}(\mathsf{crs}_1, y_i, \pi_i) = 1$, for every $i \in [n]$. Then, by the property of $i\mathcal{O}$ and the correctness of the underlying PKE and LFE, we have:

$$\begin{aligned} y &= \mathsf{LFE.Dec}(\mathsf{crs}_2, C, \mathsf{ct}') \\ &= \mathsf{LFE.Dec}(\mathsf{crs}_2, C, i\mathcal{O}(G_{\mathsf{digset}_C})((c_{1,1}, c_{1,2}, \pi_1), \ldots, (c_{n,1}, c_{n,2}, \pi_n))) \\ &= \mathsf{LFE.Dec}(\mathsf{crs}_2, C, \mathsf{LFE.Enc}(\mathsf{crs}_2, \mathsf{digest}_C, x_1 \parallel \ldots \parallel x_n)) = C(x_1, \ldots, x_n) \end{aligned}$$

Lemma 3. (Laconic Property). *According to the efficiency of underlying* PKE, NIWI, *$i\mathcal{O}$ (assume $i\mathcal{O}$ is of polynomial p to its input size), and* LWE-based LFE *[31], our construction above is laconic for unbounded-size circuit class $\mathcal{C}$.*

Proof. Now, for a circuit $C \in \mathcal{C} : (\{0,1\}^k)^n \rightarrow \{0,1\}^\ell$ of circuit size $|C| = \omega(p(n, k, \lambda))$ and depth d, and security parameter λ, according to the parameters of LWE-based LFE, we analysis the parameters in our construction as follows:

- The crs consists of crs_1 for NIWI, and crs_2 of size $(n \times k) \cdot \mathsf{poly}(\lambda, d)$ for LWE-based LFE. Hence, the size of crs is much smaller than the circuit size of C.
- The digest is of size $\mathsf{poly}(\lambda)$ for LWE-based LFE. The SK consists of $(\mathsf{pk}_1, \mathsf{pk}_2, \mathsf{sk}_1)$ for PKE encryption scheme, commitments $\{z^j_{1,i}\}$ and z_2, and randomness r_0. Then, both the size of the digest and SK is independent with $|C|$.
- The encryption algorithm consists of generating two PKE encryptions, a NIWI proof string and an indistinguishable obfuscation for a circuit. The generation of two PKE encryptions and the corresponding proof string is independent with the circuit size of C. And, the main size of the circuit been obfuscated is the size of LWE-based $\mathsf{LFE.Enc}$, about $\tilde{O}(n \times k + \ell) \cdot \mathsf{poly}(\lambda, d)$. Then, the obfuscation of the circuit should be around $p(\lambda, n, k, \ell, d)$. Therefore, both the run-time of the encryption algorithm and the size of the ciphertext are much smaller than the circuit size $|C| = \omega(p(n, k, \lambda))$ of circuit C.

As the discussion above, we conclude that our construction is laconic.

Theorem 4. (Security Proof). *Let $q = q(\lambda)$ be such that $q^n = poly(\lambda)$, Then assume indistinguishability obfuscator for all polynomial-size computable*

circuits, one-way functions and selectively (adaptively) secure laconic function evaluation, the above construction is (n, q)-SIM *selectively (adaptively) secure.*

Proof. (*Proof Sketch).* To prove the above theorem, we first construct an ideal world simulator S.

Simulator S Recall the security definition in Fig. 1, the simulator is given the common reference string crs, circuit C, digest digest_C, and values $\{C(x_1^{j_1}, \ldots, x_n^{j_n})\}$ for $j_1, \ldots, j_n \in [q]$. The simulator S works as follows:

Simulate PKE Encryptions of Challenge Message:
- For all $i \in [n]$ and $j \in [q]$, S computes $c_{i,1}^j \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, 0)$ and $c_{i,2}^j \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_2, 0)$.
- S computes $z_2 \leftarrow \mathsf{Com}(1)$.

Simulate NIWI proof for PKE Encryptions of Challenge Message:
- For every $i \in [n]$, $j \in [q]$, S computes $z_{1,i}^j \leftarrow \mathsf{Com}(c_{i,1}^j \parallel c_{i,2}^j)$. Let $r_{1,i}^j$ denote the randomness used to compute $z_{1,i}^j$.
- Let $y_i^j = (c_{i,1}^j, c_{i,2}^j, \mathsf{pk}_1, \mathsf{pk}_2, \{z_{1,i}^j\}, z_2)$. S computes the proof string $\pi_i^j \leftarrow \mathsf{NIWI.Prove}(\mathsf{crs}_1, y_i^j, \omega_i^j)$, where the witness ω_i^j corresponds to the trapdoor witness $(j, r_{1,i}^j)$. That is, ω_i^j establishes that $z_{1,i}^j$ is a commitment to $c_{i,1}^j \parallel c_{i,2}^j$.

Simulate Indistinguishable Obfuscation: S computes indistinguishable obfuscation of the circuit $\mathsf{SIM}.G_{\mathsf{digest}_C}$, where $\mathsf{SIM}.G_{\mathsf{digest}_C}$ is defined in Fig. 4

Then we describe a sequence of hybrid experiments $\mathbf{H}_0, \ldots, \mathbf{H}_7$, where $\mathbf{H}_0$ corresponds to the real world experiment and $\mathbf{H}_7$ corresponds to the ideal world experiment. For every i, we will prove that the output of $\mathbf{H}_i$ is computationally indistinguishable from the output of $\mathbf{H}_{i+1}$.

Hyb $\mathbf{H}_0$: This is the real world experiment.

Hyb $\mathbf{H}_1$: This experiment is the same as $\mathbf{H}_0$ except that in every challenge ciphertext $\mathsf{CT}_i^j = (c_{i,1}^j, c_{i,2}^j, \pi_i^j, i\mathcal{O}(G_{\mathsf{digest}_C})$, the indistinguishable obfuscation of G_{digest_C} is replaced by the indistinguishable obfuscation of G', G' is defined in Fig. 5.
The indistinguishability between hybrids $\mathbf{H}_0$ and $\mathbf{H}_1$ follows from the property of indistinguishable obfuscator and security of laconic function evaluation. We refer to the full version for more details.

Hyb $\mathbf{H}_2$: This experiment is the same as $\mathbf{H}_1$ except that we start generating $z_{1,i}^j$ as a commitment to $c_{i,1}^j \parallel c_{i,2}^j$ rather than $0^{2\ell_{\mathsf{ct}}}$, for all $i \in [n], j \in [q]$.
The indistinguishability between hybrids $\mathbf{H}_1$ and $\mathbf{H}_2$ follows directly from the computational hiding property of the commitment scheme, since that there is nothing else corresponding to the commitments $\{z_{1,i}^j\}$ in these two experiments.

Hyb $\mathbf{H}_3$: This experiment is the same as $\mathbf{H}_2$ except that in every challenge ciphertext $\mathsf{CT}_i^j = (c_{i,1}^j, c_{i,2}^j, \pi_i^j, i\mathcal{O}(G'))$, the corresponding proof string π_i^j is computed using a trapdoor witness $(j, r_{1,i}^j)$, where $r_{1,i}^j$ be the randomness to generate $z_{1,i}^j \leftarrow \mathsf{Com}(c_{i,1}^j \parallel c_{i,2}^j)$, for all $i \in [n], j \in [q]$.

$\mathsf{SIM}.G_{\mathsf{digest}_C}[\{(c^j_{i,1}, c^j_{i,2})\}, \{z^j_{1,i}\}, z_2, \{C(x^{j_1}_1, \ldots, x^{j_n}_n)\}]((c_{1,1}, c_{1,2}, \pi_1), \ldots, (c_{n,1}, c_{n,2}, \pi_n))$

Input: PKE ciphertext and proof pairs $(c_{i,1}, c_{i,2}, \pi_i)$, for $i \in [n]$.

Hardcoded: statements of challenge ciphertext $(c^j_{i,1}, c^j_{i,2}, \mathsf{pk}_1, \mathsf{pk}_2, \{z^j_{1,i}\}, z_2)$ for $i \in [n], j \in [q]$, and values $\{C(x^{j_1}_1, \ldots, x^{j_n}_n)\}$ for $j_1, \ldots, j_n \in [q]$.

1. For every $i = 1, \ldots, n$, let $y_i = (c_{i,1}, c_{i,2}, \mathsf{pk}_1, \mathsf{pk}_2, \{z^j_{1,i}\}, z_2)$ be the statement corresponding to the proof string π_i. If $\mathsf{NIWI.Verify}(\mathsf{crs}_1, y_i, \pi_i) = 0$, then stop and output $\perp$; Otherwise continue to $i + 1$.
2. If $\exists(j_1, \ldots, j_n), s.t$ for every $i \in [n]$: $c_{i,1} = c^{j_i}_{i,1}$, and $c_{i,2} = c^{j_i}_{i,2}$, then stop and output $\mathsf{LFE.SIM}(\mathsf{crs}, \mathsf{digest}_C, C, C(x^{j_1}_1, \ldots, x^{j_n}_n))$; Otherwise output $\perp$.

Fig. 4. Description of the Circuit $\mathsf{SIM}.G_{\mathsf{digest}_C}$

The indistinguishability between hybrids $\mathbf{H}_2$ and $\mathbf{H}_3$ follows directly from the witness indistinguishable property of NIWI proof system.

Hyb $\mathbf{H}_4$: This experiment is the same as $\mathbf{H}_3$ except that we start generating z_2 as a commitment to 1 instead of 0.
The indistinguishability between hybrids $\mathbf{H}_3$ and $\mathbf{H}_4$ follows directly from the computational hiding property of the commitment scheme.

Hyb $\mathbf{H}_5$: This experiment is the same as $\mathbf{H}_4$ except that in the ciphertexts of PKE encryption for challenge message pairs, the second ciphertext $c^j_{i,2}$ is an encryption of zeros, i.e., $c^j_{i,2} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_2, 0)$.
The indistinguishability between hybrids $\mathbf{H}_4$ and $\mathbf{H}_5$ follows immediately from the semantic security of PKE encryption scheme.

Hyb $\mathbf{H}_6$: This experiment is the same as $\mathbf{H}_5$ except that in each challenge ciphertext $\mathsf{CT}^j_i = (c^j_{i,1}, c^j_{i,2}, \pi^j_i, i\mathcal{O}(G'))$, the indistinguishable obfuscation of G' is replaced by the indistinguishable obfuscation of $\mathsf{SIM}.G_{\mathsf{digest}_C}$.
The indistinguishability between hybrids $\mathbf{H}_5$ and $\mathbf{H}_6$ follows from the property of indistinguishable obfuscator, the perfectly binding property of the commitment scheme, and the statistical soundness property of NIWI proof system. We refer to the full version for formal proof.

Hyb $\mathbf{H}_7$: This experiment is the same as $\mathbf{H}_6$ except that in the ciphertexts of PKE encryption for challenge message pairs, the first ciphertext $c^j_{i,1}$ is an encryption of zeros, i.e., $c^j_{i,1} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, 0)$. Note that this is the ideal world experiment.
The indistinguishability between hybrids $\mathbf{H}_6$ and $\mathbf{H}_7$ follows immediately from the semantic security of PKE encryption scheme.

This completes the security proof of our construction.

$G'[\mathsf{SK}, \mathsf{digest}_C, \{(c^j_{i,1}, c^j_{i,2})\}, \{C(x^{j_1}_1, \ldots, x^{j_n}_n)\}]((c_{1,1}, c_{1,2}, \pi_1), \ldots, (c_{n,1}, c_{n,2}, \pi_n))$

Input: PKE ciphertext and proof pairs $(c_{i,1}, c_{i,2}, \pi_i)$, for $i \in [n]$.
Hardcoded: secret key SK, digest digest_C, PKE encryptions for challenge message $\{(c^j_{i,1}, c^j_{i,2})\}$ and values $\{C(x^{j_1}_1, \ldots, x^{j_n}_n)\}$.

1. For every $i = 1, \ldots, n$, let $y_i = (c_{i,1}, c_{i,2}, \mathsf{pk}_1, \mathsf{pk}_2, \{z^j_{1,i}\}, z_2)$ be the statement corresponding to the proof string π_i. If $\mathsf{NIWI.Verify}(\mathsf{crs}_1, y_i, \pi_i) = 0$, then stop and output $\perp$; Otherwise continue to $i + 1$.
2. If $\exists (j_1, \ldots, j_n), s.t$ for every $i \in [n]$: $c_{i,1} = c^{j_i}_{i,1}$, and $c_{i,2} = c^{j_i}_{i,2}$, then stop and output $\mathsf{LFE.SIM}(\mathsf{crs}, \mathsf{digest}_C, C, C(x^{j_1}_1, \ldots, x^{j_n}_n))$; Otherwise continue to the next step.
3. Compute $x_i = \mathsf{PKE.Dec}(\mathsf{sk}_1, c_{i,1})$.
4. Outputs $\mathsf{ct}' \leftarrow \mathsf{LFE.Enc}(\mathsf{crs}_2, \mathsf{digest}_C, (x_1, \ldots, x_n))$.

Fig. 5. Description of the circuit G'

6 Conclusion

The client-optimized MPC is the main motivation for this work, which yields the first study regarding multi-input laconic function evaluation. We propose definitions of variant multi-input laconic function evaluation and then explore construction and impossibility result of variants of it. Specifically, We show that public-key MI-LFE implies VBB obfuscation for all circuits, a primitive that is impossible to achieve. Then we build private-key MI-LFE from $i\mathcal{O}$. The use of $i\mathcal{O}$ is inevitable here as private-key MI-LFE can be used to construct witness encryption or $i\mathcal{O}$, which do not have constructions based on standard assumptions yet. Therefore, an interesting open problem is to explore MI-LFE for some special function families, such as inner product, or weaken the security requirement of MI-LFE to make it plausible to have a construction based on standard assumptions.

Acknowledgement. The authors thank anonymous reviewers for valuable comments. Qiang is supported by NSF CNS #1801492, and a Google Faculty Award. Bo Pang is supported by National Key R&D Program of China-2017YFB0802202 NSFC61772516.

References

1. Hemisphere project. https://en.wikipedia.org/wiki/Hemisphere_Project
2. Official words on surveillance. https://www.usatoday.com/news/washington/2006-05-10-nsa_x.htm
3. Ananth, P., Jain, A., Naor, M., Sahai, A., Yogev, E.: Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 491–520. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_17

4. Applebaum, B., Brakerski, Z., Tsabary, R.: Perfect secure computation in two rounds. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 152–174. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_6
5. Barak, B., et al.: On the (Im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1
6. Beaver, D.: Foundations of secure interactive computing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 377–391. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_31
7. Bellare, M., Hoang, V.T.: Adaptive witness encryption and asymmetric password-based cryptography. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 308–331. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_14
8. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th ACM STOC, pp. 103–112. ACM Press, May 1988
9. Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 563–594. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_19
10. Boneh, D., Sahai, A., Waters, B.: Functional encryption: definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19571-6_16
11. Brakerski, Z., Jain, A., Komargodski, I., Passelègue, A., Wichs, D.: Non-trivial witness encryption and null-io from standard assumptions. In: Catalano, D., De Prisco, R. (eds.) SCN 2018. LNCS, vol. 11035, pp. 425–441. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_23
12. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_29
13. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. In: Naor, M., (ed.) ITCS 2014, pp. 1–12. ACM, January 2014
14. Canetti, R., Holmgren, J., Jain, A., Vaikuntanathan, V.: Succinct garbling and indistinguishability obfuscation for RAM programs. In: Servedio, R.A., Rubinfeld, R., (eds.) 47th ACM STOC, pp, 429–437. ACM Press, June 2015
15. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_31
16. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS, pp. 40–49. IEEE Computer Society Press, October 2013
17. Garg, S., Gentry, C., Halevi, S., Wichs, D.: On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 518–535. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_29
18. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Functional encryption without obfuscation. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9563, pp. 480–511. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_18

19. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Boneh, D., Roughgarden, T., Feigenbaum, J., (eds.) 45th ACM STOC, pp. 467–476. ACM Press, June 2013
20. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 468–499. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_16
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC, pp. 169–178. ACM Press, May/June 2009
22. Gentry, C., Lewko, A., Waters, B.: Witness encryption from instance independent assumptions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 426–443. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_24
23. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
24. Goldwasser, S., et al.: Multi-input functional encryption. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 578–602. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_32
25. Goldwasser, S., Goyal, V., Jain, A., Sahai, A.: Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/727 (2013). http://eprint.iacr.org/2013/727
26. Goldwasser, S., Kalai, Y., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Boneh, D., Roughgarden, T., Feigenbaum, J., (eds.) 45th ACM STOC, pp. 555–564. ACM Press, June 2013
27. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21
28. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. In: Servedio, R.A., Rubinfeld, R., (eds.) 47th ACM STOC, pp. 419–428. ACM Press, June 2015
29. Lin, H., Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation with non-trivial efficiency. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016. LNCS, vol. 9615, pp. 447–462. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49387-8_17
30. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 735–763. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_26
31. Quach, W., Wee, H., Wichs, D.: Laconic function evaluation and applications. In: Thorup, M. (ed.) 59th FOCS, pp. 859–870. IEEE Computer Society Press, October (2018)
32. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th FOCS, pp. 162–167. IEEE Computer Society Press, October 1986