Feasibility and Infeasibility of Adaptively Secure Fully Homomorphic Encryption

Jonathan Katz[⋆], Aishwarya Thiruvengadam, and Hong-Sheng Zhou[⋆⋆]

Department of Computer Science, University of Maryland
{jkatz,aish,hszhou}@cs.umd.edu

Abstract. Fully homomorphic encryption (FHE) is a form of public-key encryption that enables arbitrary computation over encrypted data. The past few years have seen several realizations of FHE under different assumptions, and FHE has been used as a building block in many cryptographic applications.

Adaptive security for public-key encryption schemes is an important security notion proposed by Canetti et al. It is intended to ensure security when encryption is used within an interactive protocol and parties may be *adaptively* corrupted by an adversary during the course of the protocol execution. Due to the extensive applications of FHE to protocol design, it is natural to understand whether adaptively secure FHE is achievable.

In this paper we show two contrasting results in this direction. First, we show that adaptive security is *impossible* for FHE satisfying the (standard) *compactness* requirement. On the other hand, we show a construction of adaptively secure FHE that is not compact, but that does achieve circuit privacy.

1 Introduction

1.1 Fully Homomorphic Encryption

Fully homomorphic encryption (FHE) [18,11] is a form of public-key encryption that enables a third party (who does not know the associated secret key) to perform computations over encrypted data. That is, given a public key pk and a ciphertext $c = \mathsf{Enc}_{pk}(m)$ that is the encryption of some (unknown) plaintext message m, anyone can compute a ciphertext c' whose decryption is $f(m)$ for any

[⋆] Research sponsored in part by the US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained here are those of the authors and should not be interpreted as representing the official policies, expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[⋆⋆] Supported by an NSF CI postdoctoral fellowship.

desired function f. The actual definition is even more general (see Section 2.1): given pk and ciphertexts

$$c_1 = \mathsf{Enc}_{pk}(m_1), \ldots, c_\ell = \mathsf{Enc}_{pk}(m_\ell),$$

it is possible to compute an encryption of $f(m_1, \ldots, m_\ell)$.

FHE has several applications. As one example, FHE can be used to construct simple protocols for secure computation. We restrict ourselves to the two-party setting with honest-but-curious parties. (In this setting two parties with inputs x and y, respectively, wish to compute a function $f(x, y)$ over their inputs without revealing to each other anything more than the result; in the honest-but-curious setting, the parties are again assumed to follow the protocol though privacy of their inputs must still be maintained.) Here, a party with input x can generate a public/private key pair (pk, sk) for an FHE scheme and send $pk, \mathsf{Enc}_{pk}(x)$ to the other party. This second party can then compute an encryption of the desired result $f(x, y)$ and send the resulting ciphertext back to the first party. The first party can then decrypt this ciphertext to recover $f(x, y)$.

FHE with the functionality described above can be realized trivially by the construction in which we define $f, \mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell)$ to be a valid encryption of $f(m_1, \ldots, m_\ell)$. This notion, however, does not suffice for most applications of FHE; in particular, it does not suffice for the application described above. Thus, some "non-triviality" requirement must be added to the definition of FHE in order to make the notion meaningful. Various requirements can be considered, and we consider two here: (1) *compactness*, which requires that ciphertexts have bounded length, and (2) *circuit privacy*, which informally requires that the encryption of $f(m_1, \ldots, m_\ell)$ should not reveal f. Note that the trivial scheme described earlier does not satisfy either of these conditions.

1.2 Adaptive Security

In a separate line of work, Canetti et al. [6] proposed the notion of *adaptive security* for (standard) public-key encryption schemes. Their motivation was to guarantee security when encryption schemes are used to encrypt messages sent during an interactive protocol, and parties running the protocol can be *adaptively* corrupted during the course of the protocol execution [4]. (The adaptive-corruption model stands in contrast to the *static*-corruption model where the attacker is assumed to corrupt parties only *before* the protocol begins.)

The primary challenge with regard to adaptively secure encryption is that the protocol *simulator* (used to prove security of the protocol) must simulate the ciphertexts being sent by the various parties without knowing the underlying plaintext. At some later point in time, the adversary may request to corrupt a party and the simulator must then simulate for the adversary any secret keys held by that party. (If secure erasure is not assumed, the simulator will also have to simulate for the adversary any random coins used by the sender. This only makes the problem harder.) These secret keys must be such that they correctly decrypt any ciphertexts previously sent to that party. Most natural public-key encryption schemes will not be suitable here, in particular because a given public key typically has a unique secret key associated with it, which implies that any (correctly generated) ciphertext can be "opened" later to at most one plaintext.

Canetti et al. [6] show how to construct adaptively secure encryption schemes from general assumptions. Subsequent research [3,10,14,15,7,9] has shown more efficient constructions based on specific number-theoretic assumptions, or satisfying weaker (but still meaningful) notions of adaptive security.

1.3 Adaptively Secure FHE?

Because of the applications of FHE to protocol design, it is natural to ask whether adaptive security can be realized for FHE. We focus on *receiver* corruption; equivalently, we assume secure erasure and so senders can erase the random coins they use immediately before sending a ciphertext. (But the receiver cannot erase its secret key until it receives and decrypts the ciphertext.)

We show two results in this regard. First, we show unconditionally that adaptive security is *impossible* for FHE schemes satisfying compactness.[1] On the other hand, we show that adaptive security *is* possible for FHE schemes satisfying circuit privacy. Our results are interesting in their own right, but also show a separation between two notions of non-triviality (namely, compactness and circuit privacy) that have been considered in the literature.

2 Definitions

Throughout, we let k denote the security parameter.

2.1 Fully Homomorphic Encryption

We begin by formally defining the notion of fully homomorphic encryption.

Definition 1 (Fully homomorphic encryption). *Fix a function $\ell = \ell(k)$. An ℓ-*homomorphic encryption scheme *$\mathcal{HE}$ for a class of circuits $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ consists of four polynomial-time algorithms* Gen, Enc, Dec, *and* Eval *such that*

- Gen, *the* key-generation algorithm, *is a randomized algorithm that takes the security parameter 1^k as input and outputs a public key pk and secret key sk.*
- Enc, *the* encryption algorithm, *is a randomized algorithm that takes a public key pk and a message $m \in \{0,1\}$ as input, and outputs a ciphertext c.*
- Dec, *the* decryption algorithm, *is a deterministic algorithm that takes the secret key sk and a ciphertext c as input, and outputs a message $m \in \{0,1\}$.*

[1] The impossibility result of Nielsen [17] does not apply to our setting, since we are willing to place an *a priori* upper bound on the length of plaintext(s) that are encrypted under a single public key. This makes sense when encryption is used to encrypt messages sent within an interactive protocol, where the length of the messages to be encrypted is bounded in advance (and a fresh public key can be generated for each independent protocol execution).

– Eval, *the* homomorphic evaluation algorithm, *takes as input a public key* pk, *a circuit* $C \in \mathcal{C}_k$, *and ciphertexts*[2] $c_1, \cdots, c_{\ell(k)}$; *it outputs a ciphertext* c^*.

The following properties are required to hold:

1. *For any* k, *any* (pk, sk) *output by* $\mathsf{Gen}(1^k)$, *and any* $m \in \{0,1\}$ *we have* $m = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m))$.
2. *For any* k, *any* (pk, sk) *output by* $\mathsf{Gen}(1^k)$, *any* $m_1, \ldots, m_{\ell(k)} \in \{0,1\}$, *and any* $C \in \mathcal{C}_k$, *we have*

$$C(m_1, \ldots, m_\ell) = \mathsf{Dec}_{sk}(\mathsf{Eval}_{pk}(C, \mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell)))$$

We use the standard notion of security against chosen-plaintext attacks. (Although stronger notions of security could be considered, the question of adaptive security is tangential to these considerations.)

Definition 2. *A homomorphic encryption scheme* $\mathcal{HE}$ *is* CPA-secure *if for any polynomial-time adversary* $\mathcal{A}$ *the following is negligible in* k*:*

$$|\Pr[\mathcal{A}(pk, \mathsf{Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, \mathsf{Enc}_{pk}(1)) = 1]|,$$

where $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$.

As noted earlier, Definitions 1 and 2 are not enough to capture a meaningful notion of fully homomorphic encryption because they can be satisfied by a "trivial" construction starting from any CPA-secure (standard) public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}')$ by defining Enc, Eval, and Dec as follows:

– $\mathsf{Enc}_{pk}(m) = (0, \mathsf{Enc}'_{pk}(m))$.
– $\mathsf{Eval}_{pk}(C, c_1, \ldots, c_\ell)$ outputs $(1, C, c_1, \ldots, c_\ell)$.
– $\mathsf{Dec}_{sk}(c)$ does as follows: if $c = (0, c')$, then output $\mathsf{Dec}'_{sk}(c')$ (i.e., decrypt as in Π). If $c = (1, C, c_1, \ldots, c_\ell)$, then output

$$C(\mathsf{Dec}'_{sk}(c_1), \ldots, \mathsf{Dec}'_{sk}(c_\ell))$$

(i.e., decrypt and then apply C to the results).

There are various ways one could imagine ruling out trivial schemes like the above. The first approach (following previous work in the literature) is to require that ciphertexts cannot grow arbitrarily large; this is known as *compactness*.

Definition 3 (Compactness). *An* ℓ*-homomorphic encryption scheme* $\mathcal{HE}$ *for a class of circuits* $\{\mathcal{C}_k\}_{k \in \mathbb{N}}$ *is* compact *if there exists a polynomial* $\alpha = \alpha(k)$ *such that ciphertexts output by* Eval *have length at most* α. *(For this to be non-trivial it should be the case that, for all* k, *we have* $\alpha(k) \leq |C|$ *for some* $C \in \mathcal{C}_k$.*)*

[2] We assume for simplicity that all circuits in $\mathcal{C}_k$ take exactly $\ell = \ell(k)$ input bits.

We say an ℓ-homomorphic encryption scheme is *ℓ-fully homomorphic* if it is homomorphic for all boolean circuits, CPA-secure, and compact.

An alternate non-triviality condition that has been considered is to require that the output of $\mathsf{Eval}_{pk}(C, c_1, \ldots, c_\ell)$ reveal nothing about C, even to the holder of the secret key sk. This notion is called *circuit privacy.* There are different ways of formalizing such a notion. The definition we use is weaker than the one introduced by Gentry [12], but similar to the notion considered in [13]. We also note that we allow (an upper bound on) the *size* of C to be revealed.

Definition 4 (Circuit privacy). *An ℓ-homomorphic encryption scheme $\mathcal{HE}$ for a class of circuits $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ is* circuit private *if there exists an efficient simulator $\mathcal{S}$ such that for every (pk, sk) generated by* Gen*, every $C \in \mathcal{C}_k$, and every $m_1, \ldots, m_\ell$, the following two distributions are computationally indistinguishable (even given $pk, sk, C, m_1, \ldots, m_\ell$):*

$$\left\{\forall i : c_i \leftarrow \mathsf{Enc}_{pk}(m_i) : \left(\mathsf{Eval}_{pk}(C, c_1, \cdots, c_\ell), c_1, \ldots, c_\ell\right)\right\}$$

$$\left\{\forall i : c_i \leftarrow \mathsf{Enc}_{pk}(m_i) : \mathcal{S}(1^k, pk, |C|, C(m_1, \ldots, m_\ell), c_1, \cdots, c_\ell)\right\}.$$

2.2 Adaptively Secure Fully Homomorphic Encryption

We consider here a security notion for FHE that captures adaptive corruption of the *receiver.* (Alternately, we assume secure erasure and thus the sender can erase the randomness it uses for encryption immediately after encryption is complete.) Here, a simulator is required to commit to (a bounded number of) simulated ciphertexts $c_1, \ldots, c_\ell$; the adversary then outputs messages $m_1, \ldots, m_\ell \in \{0, 1\}$, and the simulator must give the adversary a (single) secret key sk that "explains" (i.e., decrypts) each ciphertext c_i as m_i.

Definition 5 (Adaptively secure FHE). *An ℓ-homomorphic encryption scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is* adaptively secure *if there exists a non-uniform, polynomial-time algorithm $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all non-uniform, polynomial-time algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ we have*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

```
$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k)}$
    $(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$;
    $(m_1, \ldots, m_\ell, \tau) \leftarrow \mathcal{A}_1(1^k, pk)$;
    $sk \leftarrow \mathcal{S}_2(s, m_1, \ldots, m_\ell)$;
    $b \leftarrow \mathcal{A}_2(\tau, pk, c_1, \ldots, c_\ell, sk)$;
    return $b$.

$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
    $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$;
    $(m_1, \ldots, m_\ell, \tau) \leftarrow \mathcal{A}_1(1^k, pk)$;
    $c_1 \leftarrow \mathsf{Enc}_{pk}(m_1); \ldots;$
        $c_\ell \leftarrow \mathsf{Enc}_{pk}(m_\ell)$;
    $b \leftarrow \mathcal{A}_2(\tau, pk, c_1, \ldots, c_\ell, sk)$;
    return $b$.
```

□

3 Impossibility Result

In this section, we show that adaptively secure ℓ-fully homomorphic encryption is *impossible.* We first give some intuition. Say adaptively secure FHE were possible, so there is a simulator as in Definition 5. This gives an alternate way of computing any function f, described by a circuit $C_f : \{0,1\}^\ell \to \{0,1\}$, as follows:

1. Run $\mathcal{S}_1$ to obtain pk, $c_1, \ldots, c_\ell$, and state s.
2. Compute $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$.
3. Given input $x \in \{0,1\}^\ell$, run $\mathcal{S}_2(s, x_1, \ldots, x_\ell)$ to obtain a secret key sk.
4. Compute $\mathsf{Dec}_{sk}(c')$ to obtain $f(x)$.

Note that steps 1 and 2 can be computed in advance of receiving the input x. Thus, we can hard-code s, c', and randomness (if any) for $\mathcal{S}_2$ into a circuit that, upon receiving input $x = (x_1, \ldots, x_\ell)$, computes $sk = \mathcal{S}_2(s, x)$ and then outputs $\mathsf{Dec}_{sk}(c')$. Adaptive security implies that this output must be correct for most inputs x. (More precisely, it guarantees that there *exist* values of s, c', and randomness for $\mathcal{S}_2$ for which the circuit is correct for most inputs x.) But because Dec and $\mathcal{S}_2$ are algorithms of some fixed complexity, and c' is of some bounded size (here is where we use the compactness property), we have some polynomial upper-bound t on the size of the circuit that we get above. Taking f to be a function that cannot be approximated by circuits of size t, but that can be computed by a circuit of some polynomial size $T \gg t$, we obtain a contradiction.

Theorem 1. *Let $\ell = \omega(\log k)$. Then, adaptively secure fully ℓ-homomorphic encryption does not exist.*

Proof. Assume, toward a contradiction, that such a scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ exists. This implies the existence of a non-uniform family of circuits $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ satisfying Definition 5. Let $t(k)$ denote an upper bound on the size of the circuit for $\mathcal{S}_2$ plus the size of a circuit computing Dec for any ciphertext c' output by Eval. Using compactness (which says that the size of any such c' is bounded by some fixed polynomial) and the fact that Dec runs in polynomial time, we see that $t(k) = \mathsf{poly}(k)$.

Let $\{f_k : \{0,1\}^{\ell(k)} \to \{0,1\}\}_k$ be a function family that can be computed by polynomial-size circuits. Fix some particular k in the discussion that follows, and let C_f be the circuit computing $f = f_k$. Define a circuit $C^*_{s,c',\omega}$ as follows. First, compute $(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$. Then compute $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$. Choose random coins ω for $\mathcal{S}_2$, and define $C^*_{s,c',\omega}$ as:

– On input $x \in \{0,1\}^\ell$, run $\mathcal{S}_2(s, x_1, \ldots, x_\ell)$ (using random coins ω) to obtain sk. Then output $\mathsf{Dec}_{sk}(c')$.

We stress that s, c', and ω are hard-coded into the above circuit. Thus, the size of $C^*_{s,c',\omega}$ is at most $t(k)$.

A circuit $C : \{0,1\}^\ell \to \{0,1\}$ is an *ϵ-approximation of f* if

$$\Pr_{x \leftarrow \{0,1\}^\ell}[C(x) = f(x)] \geq \epsilon.$$

The theorem follows from the next two lemmas.

Lemma 1. *There exist s, c', ω such that the circuit $C^*_{s,c',\omega}$ constructed above is a 3/4-approximation of f.*

Proof. Consider the following non-uniform, polynomial-size adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$: adversary $\mathcal{A}_1$, on input pk, outputs random $x_1, \ldots, x_\ell \in \{0,1\}$. On input $c_1, \ldots, c_\ell$ and sk, adversary $\mathcal{A}_2$ computes $c' \leftarrow \mathsf{Eval}_{pk}(C_f, c_1, \ldots, c_\ell)$ followed by $y = \mathsf{Dec}_{sk}(c')$. (Non-uniformity is used to hard-wire into $\mathcal{A}_2$ a description of the circuit C_f.) Finally, $\mathcal{A}_2$ outputs 1 if and only if $y = f(x_1, \ldots, x_\ell)$.

Correctness of the FHE scheme implies that in $\mathsf{Real}_{\mathcal{A}}(k)$ the adversary always outputs 1. Adaptive security thus implies that the adversary outputs 1 with all but negligible probability in $\mathsf{Ideal}_{\mathcal{A}}(k)$. But this means that

$$\Pr_{x,s,c',\omega}[C^*_{s,c',\omega}(x) \neq f(x)] < \mathsf{negl}(k),$$

where $x \in \{0,1\}^\ell$ is chosen uniformly and s, c', ω are generated as in the construction of $C^*_{s,c',\omega}$ described earlier. But then there exist s, c', ω for which

$$\Pr_x[C^*_{s,c',\omega}(x) \neq f(x)] < \mathsf{negl}(k),$$

where the probability is now only over the uniform choice of $x \in \{0,1\}^\ell$. This circuit $C^*_{s,c',\omega}$ is thus a 3/4-approximation of f.

The contradiction is given by the fact that there exist functions f that can be computed by circuits of polynomial size T but cannot be 3/4-approximated by circuits of size t.

Lemma 2. *For any $t(k) = \mathsf{poly}(k)$ and $\ell(k) = \omega(\log k)$, there exists a function family $\{f_k : \{0,1\}^{\ell(k)} \to \{0,1\}\}_k$ that can be computed by circuits of polynomial size $T(k)$ but cannot be 3/4-approximated by any circuit of size $t(k)$.*

Proof. A proof follows via suitable modification of the proof of the standard hierarchy theorem for non-uniform computation [1]. Pick a random function f, and consider the probability that a fixed circuit C correctly computes f on at least 3/4 of its inputs. Using Chernoff bounds, we can show that this probability is at most $e^{-2^\ell/16}$. Since there are at most $2^{2S\log S+5S}$ circuits of size S, we have that if $S = 2^{\ell/2}/16$ (and hence $2S\log S + 5S < 2^\ell/16$) there exists a function that is hard to 3/4-approximate for *all* circuits of size S.

Any function $g : \{0,1\}^n \to \{0,1\}$ is computable by a circuit of size $2^n 10n$. Let k be such that $\ell > 2.2 \log t(k)$ (here we use the fact that $\ell = \omega(\log k)$). If we set $n = 2.2 \log t(k)$ and let $f : \{0,1\}^\ell \to \{0,1\}$ be the function that applies g to the first n bits of its input, then f can be computed by a circuit of size $O(t(k)^3)$, but cannot be 3/4-approximated by circuits of size $t(k)$.

This concludes the proof of the theorem.

We note that our impossibility result also holds for a weaker definition of adaptive security where the adversary has to output the messages (whose encryptions the simulator has to explain) before getting the public key.

4 Feasibility Result

In this section, we show that adaptive security is possible for *circuit-private* fully homomorphic encryption. The main idea is to modify the constructions in [12,13,2] to use adaptively secure building blocks. Specifically, our construction is based on *(i)* a two-move semi-honest oblivious transfer (OT) protocol with receiver adaptive security, *(ii)* a projective garbling scheme leaking only the circuit size [19,5], and *(iii)* multiple-message, receiver-non-committing public-key encryption (which is a stronger version of single-message receiver non-committing encryption introduced in [14]).

We recall the high-level idea of [12,13,2], and explain how to upgrade the building blocks to achieve our goal. The idea is to use the key generation algorithm of a public-key encryption scheme to generate the public and secret keys. To perform encryption, encode the input to be sent and in addition, encrypt the randomness used for encoding. The evaluation algorithm forwards the encryption of the randomness in addition to an encoded value of the result of the evaluation of the circuit on the input sent. For decryption, using the secret key, we can decrypt to get the randomness used. This randomness can then be used to recover the result. This idea can be constructed using a regular public key encryption scheme, oblivious transfer protocol and Yao's garbling technique [19].

To achieve adaptive security, we need to upgrade the building blocks to adaptively secure ones. A semi-honest two-move OT protocol will suffice in the above construction, but to achieve adaptive security of the circuit-private homomorphic encryption we need semi-honest two-move OT but with *adaptive receiver security.* The semi-honest OT protocol in [8] is sufficient for our goal. We also need to replace the regular public key encryption with a *multi-message receiver non-commiting encryption* (a formal definition can be found below), a strengthened version of receiver non-committing encryption introduced in [14].

Before describing our construction, we recall the definitions of garbling schemes, semi-honest OT with adaptive receiver security, and then define the multi-message receiver non-committing encryption. Then, we present our construction of a circuit-private homomorphic scheme which achieves correctness, adaptive security, and circuit privacy, but not compactness.

4.1 Building Blocks

Garbling Schemes. Here, we define garbling schemes and introduce the security notion we consider for such schemes in this work. Our notation follows the recent work by Bellare, Hoang, and Rogaway [5].

A *garbling scheme* is a five-tuple of algorithms $\mathcal{G} = (\mathsf{Gb}, \mathsf{En}, \mathsf{De}, \mathsf{Ev}, \mathsf{ev})$. A string f, the original function, describes the function $\mathsf{ev}(f, .) : \{0,1\}^\ell \to \{0,1\}^n$ that we want to garble. On input f and a security parameter k, the probabilistic algorithm Gb returns a triple of strings $(F, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, f)$. String e describes an encoding function, $\mathsf{En}(\mathsf{e}, .)$, that maps an initial input $m \in \{0,1\}^\ell$ to a garbled input $X = \mathsf{En}(\mathsf{e}, m)$. String F describes a garbled function $\mathsf{ev}(F, .)$, that maps each garbled input X to a garbled output $Y = \mathsf{ev}(F, X)$. String d describes a decoding function, $\mathsf{De}(\mathsf{d}, .)$, that maps a garbled output Y to a final output $y = \mathsf{De}(\mathsf{d}, Y)$.

We consider only projective garbling schemes in this work. A *projective garbling scheme* as described in [5] is one where e encodes a list of tokens, one pair for each bit in $m \in \{0,1\}^\ell$. Encoding function $\mathsf{En}(\mathsf{e}, .)$ uses the bits of $m = m_1 \cdots m_\ell$ to select from $\mathsf{e} = X_1^0, X_1^1, \cdots, X_\ell^0, X_\ell^1$ the subvector $X = (X_1^{m_1}, \cdots, X_\ell^{m_\ell})$. A garbling scheme $\mathcal{G} = (\mathsf{Gb}, \mathsf{En}, \mathsf{De}, \mathsf{Ev}, \mathsf{ev})$ is *projective* if for all $f, m, m' \in \{0,1\}^\ell$, $k \in \mathbb{N}$, and $i \in [\ell]$, when $(F, \mathsf{e}, \mathsf{d}) \in [\mathsf{Gb}(1^k, f)]$, $X = \mathsf{En}(\mathsf{e}, m)$ and $X' = \mathsf{En}(\mathsf{e}, m')$, then $X = (X_1 \cdots X_\ell)$ and $X' = (X'_1 \cdots X'_\ell)$ are ℓ vectors, $|X_i| = |X'_i|$, and $X_i = X'_i$ if m and m' have the same ith bit.

For the privacy notion considered, we allow that certain information about the function f can be revealed and this is captured by the *side information function* $\Phi(f)$. Specifically, for this work, the side information function is the size of the circuit.

For the security notion, we describe only the simulation-based notion of privacy in [5]. We present the definition of the simulation-based security notion of privacy of a garbling scheme.

Definition 6. *Consider the following game* $\mathrm{PrvSim}_{\mathcal{G},\Phi,\mathcal{S}}$ *associated with a garbling scheme* $\mathcal{G}$*, side information function* $\Phi(f)$ *and a simulator* $\mathcal{S}$*. The adversary* $\mathcal{A}$ *is run on input* 1^k *and makes exactly one* GARBLE *query. The* GARBLE *procedure is described as follows.*

```
GARBLE($f, m$)
    $b \leftarrow \{0,1\}$
    if $m \notin \{0,1\}^\ell$ return $\perp$
    if $b = 1$ then $(F, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, f)$;   $X \leftarrow \mathsf{En}(\mathsf{e}, m)$
    else $y \leftarrow \mathsf{ev}(f, m)$;   $(F, X, \mathsf{d}) \leftarrow \mathcal{S}(1^k, y, \Phi(f))$
    return $(F, X, \mathsf{d})$
```

The adversary after getting the answer to the query must output a bit b'*. The adversary's advantage is given by:*

$$\mathbf{Adv}_{\mathcal{G}}^{\Phi,\mathcal{S}}(\mathcal{A}, k) = \left| \mathbf{Pr}\left[b' = b\right] - \frac{1}{2} \right|$$

The protocol $\mathcal{G}$ is secure over Φ if for every polynomial-time adversary $\mathcal{A}$ there is a polynomial-time simulator $\mathcal{S}$ such that $\mathbf{Adv}_{\mathcal{G}}^{\Phi,\mathcal{S}}(\mathcal{A}, k)$ is negligible.

Semi-honest OT with Adaptive Receiver Security. 1-out-of-2 Oblivious Transfer (OT) allows a receiver to obtain exactly one of two messages from a sender where the receiver remains oblivious to the other message, and the sender is oblivious to which value was received. Please refer to Figure 1 for 2-move OT. We next define secure 2-move OT scheme with adaptive receiver security.

Definition 7. *A k-bit 2-move oblivious-transfer scheme* $\mathcal{OT} = (\mathsf{OT1}, \mathsf{OT2}, \mathsf{OT3})$ *is* secure with adaptive receiver security *if the following properties hold:*

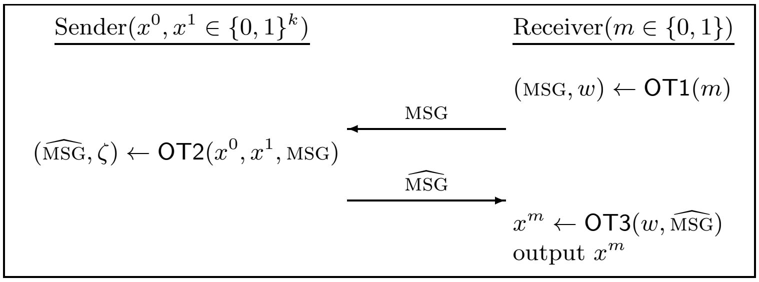
Sender$(x^0, x^1 \in \{0,1\}^k)$
Receiver$(m \in \{0,1\})$
$(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$
$\textsc{msg}$
$(\widehat{\textsc{msg}}, \zeta) \leftarrow \mathsf{OT2}(x^0, x^1, \textsc{msg})$
$\widehat{\textsc{msg}}$
$x^m \leftarrow \mathsf{OT3}(w, \widehat{\textsc{msg}})$
output x^m

Fig. 1. A 2-move OT protocol

Correctness. *For all $m \in \{0,1\}$, and $x^0, x^1 \in \{0,1\}^k$:*

$$\Pr\left[\begin{array}{l}(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m);\\ \widehat{\textsc{msg}} \leftarrow \mathsf{OT2}(x^0, x^1, \textsc{msg})\end{array} : x^m = \mathsf{OT3}(w, \widehat{\textsc{msg}})\right] \geq 1 - \mathsf{negl}(k)$$

Adaptive Receiver Security. *There exists a non-uniform, polynomial-time algorithm $\mathcal{S}^{\mathrm{recv}} = (\mathcal{S}_1^{\mathrm{recv}}, \mathcal{S}_2^{\mathrm{recv}})$ such that for all non-uniform, polynomial-time algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$:*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\mathrm{recv}}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

```
$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\mathrm{recv}}}(k)$
    $(m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
    $(\textsc{msg}, s) \leftarrow \mathcal{S}_1^{\mathrm{recv}}(1^k)$;
    $w \leftarrow \mathcal{S}_2^{\mathrm{recv}}(s, m)$;
    $b \leftarrow \mathcal{A}_2(\tau, w, \textsc{msg})$;
    return $b$
```

```
$\mathsf{Real}_{\mathcal{A}}(k)$
    $(m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
    $(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$;
    $b \leftarrow \mathcal{A}_2(\tau, w, \textsc{msg})$;
    return $b$
```

Sender Security. *There exists a non-uniform, polynomial-time algorithm $\mathcal{S}^{\mathrm{send}}$ such that for all non-uniform, polynomial-time algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$:*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\mathrm{send}}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

```
$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}^{\mathrm{send}}}(k)$
    $(x^0, x^1, m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
    $(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$;
    $\widehat{\textsc{msg}} \leftarrow \mathcal{S}^{\mathrm{send}}(1^k, \textsc{msg}, x^m, m)$;
    $b \leftarrow \mathcal{A}_2(\tau, \textsc{msg}, w, \widehat{\textsc{msg}})$;
    return $b$
```

```
$\mathsf{Real}_{\mathcal{A}}(k)$
    $(x^0, x^1, m, \tau) \leftarrow \mathcal{A}_1(1^k)$;
    $(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$;
    $\widehat{\textsc{msg}} \leftarrow \mathsf{OT2}(x^0, x^1, \textsc{msg})$;
    $b \leftarrow \mathcal{A}_2(\tau, \textsc{msg}, w, \widehat{\textsc{msg}})$;
    return $b$
```

The construction from [8] satisfies the above properties.

Multi-message, Receiver Non-committing, Public-Key Encryption. Receiver non-committing encryption (RNCE) was introduced in [14] and further studied in [7]. Here, we strengthen their notion to deal with multiple messages with an a priori bound $\alpha = \mathsf{poly}(k)$ on the number of messages; we call this α-message RNCE, and formally define it below.

- The randomized key-generation algorithm $\mathtt{gen}$ takes as input the security parameter and outputs a key-pair. This is denoted by: $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$. The public key pk defines the message space $\mathcal{M}$.
- The randomized encryption algorithm $\mathtt{enc}$ takes a public key pk and a message $m \in \mathcal{M}$. It returns a ciphertext $c \leftarrow \mathtt{enc}_{\mathrm{pk}}(m)$.
- The decryption algorithm $\mathtt{dec}$ takes as input a secret key sk and a ciphertext c, and returns a message $m \leftarrow \mathtt{dec}_{\mathrm{sk}}(c)$, where $m \in \mathcal{M} \cup \perp$.
- The randomized key-faking algorithm $\widetilde{\mathtt{gen}}$ takes as input the security parameter and outputs a public key as well as some auxiliary information. This is denoted by: $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$.
- The fake encryption algorithm $\widetilde{\mathtt{enc}}$ takes as input a tuple (pk, z) as output by $\widetilde{\mathtt{gen}}$, and outputs a tuple of fake ciphertexts and some auxiliary information: $(c_1, \ldots, c_\alpha, z') \leftarrow \widetilde{\mathtt{enc}}(\mathrm{pk}, z)$.
- The reveal algorithm $\widetilde{\mathtt{rev}}$ takes as input a tuple $(c_1, \ldots, c_\alpha, z')$ as output by $\widetilde{\mathtt{enc}}$, and a tuple of messages $m_1, \ldots, m_\alpha \in \mathcal{M}$. It outputs a fake secret key $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$. (Intuitively, sk is a valid-looking secret key for which c_i decrypts to m_i for all $i \in [\alpha]$.)

Definition 8. $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ *is a secure receiver non-committing encryption scheme for bounded* $\alpha = \mathsf{poly}(k)$*, if there exist non-uniform, polynomial-time algorithms* $\mathcal{S} = (\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$ *such that for all non-uniform, polynomial-time algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *we have*

$$|\Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k) = 1] - \Pr[\mathsf{Real}_{\mathcal{A}}(k) = 1]| \leq \mathsf{negl}(k)$$

where:

```
$\underline{\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(k)}$
    $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$;
    $(m_1, \ldots, m_\alpha, \tau) \leftarrow \mathcal{A}_1(1^k, \mathrm{pk})$;
    $(c_1, \ldots, c_\alpha, z') \leftarrow \widetilde{\mathtt{enc}}(\mathrm{pk}, z)$;
    $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$;
    $b \leftarrow \mathcal{A}_2(\tau, \mathrm{pk}, \mathrm{sk}, c_1, \ldots, c_\alpha)$;
    return $b$.
```

```
$\underline{\mathsf{Real}_{\mathcal{A}}(k)}$
    $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$;
    $(m_1, \ldots, m_\alpha, \tau) \leftarrow \mathcal{A}_1(1^k, \mathrm{pk})$;
    $c_1 \leftarrow \mathtt{enc}_{\mathrm{pk}}(m_1); \ldots;$
      $c_\alpha \leftarrow \mathtt{enc}_{\mathrm{pk}}(m_\alpha)$;
    $b \leftarrow \mathcal{A}_2(\tau, \mathrm{pk}, \mathrm{sk}, c_1, \ldots, c_\alpha)$;
    return $b$.
```

Following [14], we present a construction of a multiple-message, receiver non-committing scheme based on the DDH assumption. As in [14], we can only directly encrypt logarithmic-length messages since computation of a discrete logarithm is required by the receiver. In our case, however, we can then encrypt messages of length k by breaking the message into logarithmic-length blocks, encrypting block-by-block, and adjusting the parameter α accordingly.

- The randomized key-generation algorithm: $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$:
  Generate a group $\mathbb{G}$ of prime order q with generators $g_0, g_1, \ldots, g_\alpha$. Output the group parameters $(\mathbb{G}, q, g_0, g_1, \ldots, g_\alpha)$. Choose $x_0, \ldots, x_\alpha \leftarrow \mathbb{Z}_q$ and output the public key $\mathrm{pk} = \prod_{i=0}^{\alpha} g_i^{x_i}$ and secret key $\mathrm{sk} = (x_0, \ldots, x_\alpha)$.

– The randomized encryption algorithm $c \leftarrow \mathtt{enc}_{\mathrm{pk}}(m)$:
  View m as an element of $\mathbb{Z}_q$. Choose $r \in \mathbb{Z}_q$ at random and output the ciphertext $(g_0^r, \ldots, g_\alpha^r, P^r \cdot g_0^m)$.
– The decryption algorithm $m \leftarrow \mathtt{dec}_{\mathrm{sk}}(c)$:
  Parse $c = (A_0, \ldots, A_\alpha, B)$, and compute $C = B / \prod_i A_i^{x_i}$. Set $m = \log_{g_0} C$. (We assume m is small enough so that this computation can be done efficiently.)
– The randomized key-faking algorithm $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$:
  Generate a group $\mathbb{G}$ of prime order q with generators $g_0, g_1, \ldots, g_\alpha$ where $g_i = {g_0}^{\sigma_i}$ and σ_i is known. Output the group parameters $(\mathbb{G}, q, g_0, g_1, \ldots, g_\alpha)$. Choose $x_0, \ldots, x_\alpha \leftarrow \mathbb{Z}_q$ and output the public key $\mathrm{pk} = \prod_{i=0}^{\alpha} g_i^{x_i}$.
– The fake encryption algorithm $\widetilde{\mathtt{enc}}$:
  For $i \in \{1, \ldots, \alpha\}$, choose $r_{i,0}, \ldots, r_{i,\alpha}, r'_i \leftarrow \mathbb{Z}_q$, and output the ciphertext $(g_0^{r_{i,0}}, \ldots, g_\alpha^{r_{i,\alpha}}, g_0^{r'_i})$.
– The reveal algorithm $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', c_1, \ldots, c_\alpha, m_1, \ldots, m_\alpha)$:
  Output $x'_0, \ldots, x'_\alpha$ satisfying

$$x_0 - x'_0 + \sum_{i=1}^{\alpha} (x_i - x'_i)\sigma_i = 0$$

$$\forall j \in \{1, \ldots, \alpha\} : \ m_j + x'_0 r_{j,0} + \sum_{i=1}^{\alpha} x'_i r_{j,i} \sigma_i = r'_j.$$

  Note that this is a system of $\alpha + 1$ equations in $\alpha + 1$ unknowns, and has a solution with all but negligible probability.

4.2 Adaptively Secure Circuit-Private ℓ-Homomorphic Encryption

Our construction of a circuit-private ℓ-homomorphic encryption scheme $\mathcal{HE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$. is based on (i) a projective garbling scheme (Gb, En, De, Ev, ev) leaking only the circuit size [19,16,5], (ii) a receiver adaptively secure semi-honest OT protocol $(\mathsf{OT1}, \mathsf{OT2}, \mathsf{OT3})$, and (iii) an ℓ-message receiver non-committing public-key encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$.

– The key-generation algorithm $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$:
  Compute $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathtt{gen}(1^k)$, and set $pk := \mathrm{pk}$ and $sk := \mathrm{sk}$.
– The encryption algorithm $c \leftarrow \mathsf{Enc}_{pk}(m)$:
  Upon input $m \in \{0, 1\}$, compute $(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$. Compute $e \leftarrow \mathtt{enc}_{\mathrm{pk}}(w)$ and output $c := (\textsc{msg}, e)$.
– The decryption algorithm $m \leftarrow \mathsf{Dec}_{sk}(c)$:
  - If $c = (\textsc{msg}, e)$, compute $w \leftarrow \mathtt{dec}_{\mathrm{sk}}(e)$. Then run $\mathsf{OT1}$ with inputs 0 and 1, and randomness w, and output m for which $(\textsc{msg}, w) \leftarrow \mathsf{OT1}(m)$.
  - If $c = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ then for all $i \in [\ell]$, further parse $\hat{c}_i$ into $(\widehat{\textsc{msg}}_i, e_i)$, compute $w_i \leftarrow \mathtt{dec}_{\mathrm{sk}}(e_i)$, and compute $X_i^{m_i} \leftarrow \mathsf{OT3}(w_i, \widehat{\textsc{msg}}_i)$. This gives the garbled input $X = (X_1^{m_1}, \ldots, X_\ell^{m_\ell})$. Output $\mathsf{De}(\mathsf{d}, \mathsf{Ev}(\hat{C}, X))$ as message m.

– The evaluation algorithm $c^* \leftarrow \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$:
  - Let $\mathsf{Gb}(1^k, C) \rightarrow (\hat{C}, \mathsf{e}, \mathsf{d})$. Parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$.
  - Parse c_i into $(\textsc{msg}_i, e_i)$ for $i \in [\ell]$. Compute $\widehat{\textsc{msg}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \textsc{msg}_i)$, and set $\hat{c}_i := (\widehat{\textsc{msg}}_i, e_i)$, for all $i \in [\ell]$.
  - Finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$.

Theorem 2. *Construction $\mathcal{HE}$ presented above is an adaptively secure circuit-private ℓ-homomorphic encryption.*

Proof. We show below that the construction $\mathcal{HE}$ is a secure ℓ-homomorphic encryption that satisfies correctness, circuit privacy, and adaptive security.

Correctness. It is easy to verify the correctness. To compute $\mathsf{Dec}_{sk}(c)$ for $c = \mathsf{Enc}_{pk}(m)$ where $c = (\textsc{msg}, e)$, we first compute $w' \leftarrow \mathtt{dec}_{\mathrm{sk}}(e)$. Then run $\mathsf{OT1}$ twice with both inputs 0 and 1, and randomness w'. Output m' where $(\textsc{msg}, w') \leftarrow \mathsf{OT1}(m)$. Given the fact that $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ is correct, it holds that $w = \mathtt{dec}_{\mathrm{sk}}(\mathtt{enc}_{\mathrm{pk}}(w))$, i.e., $w = w'$; furthermore, given the fact that the OT scheme is correct, it holds that $(\textsc{msg}, w) = \mathsf{OT1}(m)$, i.e., $m = m'$. Therefore, we have $\mathsf{Dec}_{sk}(c) = m$ for $c = \mathsf{Enc}_{pk}(m)$.

To compute $\mathsf{Dec}_{sk}(c)$ for $c = \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ where $c = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$, parse $\hat{c}_i$ as $\hat{c}_i = (\widehat{\textsc{msg}}_i, e_i)$ for all $i \in [\ell]$. We first compute $w_i' \leftarrow \mathtt{dec}_{\mathrm{sk}}(e_i)$, $\hat{X}_i \leftarrow \mathsf{OT3}(w_i', \widehat{\textsc{msg}}_i)$. Then we use the input key strings $X = (\hat{X}_1, \ldots, \hat{X}_\ell)$ to evaluate the garbled circuit $\hat{C}$ as $\mathsf{Ev}(\hat{C}, X)$ and obtain $C(\hat{m}_1, \ldots, \hat{m}_\ell)$. Given the fact that $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ is correct, we have $w_i = w_i'$; furthermore, given the fact that the OT is correct, $\hat{X}_i = X_i^{m_i}$; also, given the fact that the garbling scheme is correct, we have $C(m_1, \ldots, m_\ell) = \mathsf{De}(\mathsf{d}, \mathsf{Ev}(\hat{C}, X))$. Therefore, we have $\mathsf{Dec}_{sk}(c) = m$ for $c = \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ and $m = C(m_1, \ldots, m_\ell)$.

Circuit privacy. The property of circuit privacy follows from the security of the garbling scheme and the sender security of the OT. By the security of the garbling scheme, we have that there exists a simulator $\mathcal{S}^{\mathcal{G}}$ on input the security parameter, output of the function and the side information function Φ outputs (F, X, d) except with negligible probability. As mentioned before, $\Phi(C) = |C|$ in our construction.

Let us construct a simulator $\mathcal{S}$ as follows:

– Upon receiving $(1^k, pk, |C|, C(m_1, \ldots, m_\ell), c_1, \ldots, c_\ell)$ where $c_i = \mathsf{Enc}_{pk}(m_i)$ for $i \in [\ell]$, the simulator $\mathcal{S}$ runs the simulator $\mathcal{S}^{\mathcal{G}}$ of the garbling scheme to obtain
$(F, X, \mathsf{d}) \leftarrow \mathcal{S}^{\mathcal{G}}(1^k, C(m_1, \ldots, m_\ell), |C|)$.
– Set $\hat{C} = F$. Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$.
– To compute the ciphertext $\hat{c}_i$, parse c_i into $(\textsc{msg}_i, e_i)$ as in the construction. Then compute $\widehat{\textsc{msg}}_i \leftarrow \mathcal{S}^{\mathrm{send}}(1^k, \textsc{msg}_i, \hat{X}_i)$ using the OT-simulator $\mathcal{S}^{\mathrm{send}}$ for sender security, and set $\hat{c}_i := (\widehat{\textsc{msg}}_i, e_i)$, for all $i \in [\ell]$.

– Finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$.

Next, we develop a sequence of hybrids to show that Definition 4 is satisfied.

Hybrid $\mathbf{H}_0$: As in the real scheme, we run the evaluation algorithm Eval to compute c^*, i.e, $c^* \leftarrow \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)$ where $c^* = (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$. Concretely, we use the projective garbling scheme, on input C, compute $(\hat{C}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, C)$; then we parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$, and parse c_i into $(\textsc{msg}_i, e_i)$ for $i \in [\ell]$; after that, we compute $\widehat{\textsc{msg}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \textsc{msg}_i)$, and set $\hat{c}_i := (\widehat{\textsc{msg}}_i, e_i)$, for all $i \in [\ell]$; finally, set $c^* := (\hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$. The adversary $\mathcal{A}$ is given c^* as well as the input m, circuit C and the secret key sk.

Hybrid $\mathbf{H}_{1,j}$: For $j \in [0 \ldots \ell]$, the hybrid $\mathbf{H}_{1,j}$ is the same as the Hybrid $\mathbf{H}_0$ except the following:

For all $i \in [j]$, parse c_i into $(\textsc{msg}_i, e_i)$. Compute $\widehat{\textsc{msg}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \textsc{msg}_i, X_i^{m_i})$, using the OT simulator for sender security and set $\hat{c}_i := (\widehat{\textsc{msg}}_i, e_i)$.

We argue that for $j \in [1, \ldots, \ell]$, the hybrids $\mathbf{H}_{1,j}$ and $\mathbf{H}_{1,j+1}$ are computationally indistinguishable under the assumption that the OT satisfies sender security. If there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible probability, then we can construct an adversary $\mathcal{B}$ who breaks the sender security of the OT as follows.

The adversary $\mathcal{B}$ acts as follows:

– run $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, i.e., run $(\text{pk}, \text{sk}) \leftarrow \mathtt{gen}(1^k)$ and set $pk := \text{pk}$, $sk := \text{sk}$.
– Choose $m = (m_1, \ldots, m_\ell)$. For all $i \in [\ell]$, compute $c_i \leftarrow \mathsf{Enc}_{pk}(m_i)$, i.e., compute $(\textsc{msg}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$, and $e_i \leftarrow \mathtt{enc}_{\text{pk}}(w_i)$. Set $c_i := (\textsc{msg}_i, e_i)$;
– Using the projective garbling scheme on a circuit C, let $(\hat{C}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^k, C)$. Parse the encoding function represented by string e as $(X_1^0, X_1^1, \ldots, X_\ell^0, X_\ell^1)$;
– for all $i \in [j-1]$, parse c_i into $(\textsc{msg}_i, e_i)$. Then compute $\widehat{\textsc{msg}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \textsc{msg}_i, X_i^{m_i})$;
– for $i = j$, output (m_i, X_i^0, X_i^1). Parse c_i into $(\textsc{msg}_i, e_i)$ and obtain $\widehat{\textsc{msg}}_i$, where it is either generated by $\mathcal{S}^{\text{send}}$, i.e., $\widehat{\textsc{msg}}_i \leftarrow \mathcal{S}^{\text{send}}(1^k, \textsc{msg}_i, X_i^{m_i})$, or by the OT scheme honestly, i.e., $\widehat{\textsc{msg}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \textsc{msg}_i)$;
– for all $i \in [j+1, \ell]$, parse c_i into $(\textsc{msg}_i, e_i)$. Then compute $\widehat{\textsc{msg}}_i \leftarrow \mathsf{OT2}(X_i^0, X_i^1, \textsc{msg}_i)$;
– for all $i \in [\ell]$, set $\hat{c}_i := (\widehat{\textsc{msg}}_i, e_i)$;
– Return $(m, sk, pk, C, \hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ to the internally simulated $\mathcal{A}$.

When $i = j$, if $\widehat{\textsc{msg}}_i$ obtained by $\mathcal{B}$ is generated by $\mathcal{S}^{\text{send}}$, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j}$. Otherwise, if $\widehat{\textsc{msg}}_i$ is generated by the OT scheme honestly, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j-1}$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 7 for sender security. This contradicts our assumption that the OT is sender-secure. Therefore, Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable.

Note that $\mathbf{H}_0$ is identical to $\mathbf{H}_{1,0}$. Since Hybrids $\mathbf{H}_{1,j-1}$ and $\mathbf{H}_{1,j}$ are indistinguishable as argued above, we also have that Hybrid $\mathbf{H}_0$ is computationally indistinguishable from Hybrid $\mathbf{H}_{1,\ell}$.

Hybrid $\mathbf{H}_2$: This is the same as Hybrid $\mathbf{H}_{1,\ell}$ except the following: Run the simulator $\mathcal{S}^{\mathcal{G}}$ for the projective garbling scheme to obtain the garbled circuit, input and the decoding function, i.e., $(F, X, \mathsf{d}) \leftarrow \mathcal{S}^{\mathcal{G}}(1^k, C(m_1, \ldots, m_\ell), |C|)$. Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$, and set $\hat{C} = F$. We note that this is exactly the output produced by the simulator $\mathcal{S}$ for circuit privacy.

The hybrids $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable under the assumption that $\mathcal{G}$ is a secure garbling scheme. If there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids with non-negligible probability, then we can construct an adversary $\mathcal{B}$ who breaks the security of the garbling scheme as defined in Definition 6.

Consider an adversary $\mathcal{B}$ who acts as follows:

- run $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$;
- Choose $m = (m_1, \ldots, m_\ell)$. For all $i \in [\ell]$, compute $c_i \leftarrow \mathsf{Enc}_{pk}(m_i)$;
- Choose a circuit C. Make a GARBLE query on (C, m) to obtain the challenge (F, X, d). Set $\hat{C} = F$.
- Parse X as $(\hat{X}_1, \ldots, \hat{X}_\ell)$. For all $i \in [\ell]$, parse c_i into $(\textsc{msg}_i, e_i)$, and compute $\widehat{\textsc{msg}}_i \leftarrow \mathcal{S}^{\mathrm{send}}(1^k, \textsc{msg}_i, \hat{X}_i)$; Set $\hat{c}_i = (\widehat{\textsc{msg}}_i, e_i)$;
- Return $(m, sk, pk, C, \hat{C}, \mathsf{d}, \hat{c}_1, \ldots, \hat{c}_\ell)$ to the internally simulated $\mathcal{A}$.

When $\mathcal{B}$'s challenge (F, X, d) is generated honestly, then the internally simulated $\mathcal{A}$ interacts with $\mathbf{H}_{1,\ell}$. On the other hand, when $\mathcal{B}$'s challenge is generated by $\mathcal{S}^{\mathcal{G}}$, the simulated $\mathcal{A}$ interacts with $\mathbf{H}_2$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can gain a non-negligible advantage as in Definition 6. However, this is a contradiction to our assumption that $\mathcal{G}$ is a secure garbling scheme. Therefore, Hybrid $\mathbf{H}_{1,\ell}$ and Hybrid $\mathbf{H}_2$ are indistinguishable.

Thus, we have that the hybrids $\mathbf{H}_0$ and $\mathbf{H}_2$ are indistinguishable which implies that the scheme satisfies the circuit privacy requirement as defined in Definition 4.

ADAPTIVE SECURITY. Next we give the proof idea for proving the adaptive security as defined in Definition 5. We construct the simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ as follows. The simulator is based on the algorithms $(\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$ of the ℓ-RNCE scheme, and the simulator $(\mathcal{S}_1^{\mathrm{recv}}, \mathcal{S}_2^{\mathrm{recv}})$ of the OT scheme.

- $(pk, c_1, \ldots, c_\ell, s) \leftarrow \mathcal{S}_1(1^k)$:

  Compute $(\mathrm{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, and set $pk := \mathrm{pk}$. Compute $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\mathrm{pk}, z)$.

  For all $i \in [\ell]$, compute $(\textsc{msg}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\mathrm{recv}}(1^k)$, and set $c_i := (\textsc{msg}_i, e_i)$. Store all information into state s.

– $sk \leftarrow \mathcal{S}_2(s, m_1, \ldots, m_\ell)$:
  Upon obtaining $(m_1, \ldots, m_\ell)$, recover $\{\gamma_i\}_{i\in[\ell]}$ from the state s.
  For all $i \in [\ell]$, compute $w_i \leftarrow \mathcal{S}_2^{\mathrm{recv}}(\textsc{msg}_i, \gamma_i, m_i)$.
  Compute $\mathrm{sk} \leftarrow \widetilde{\mathtt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, and set $sk := \mathrm{sk}$.

Next, we develop a sequence of hybrids to show that the real experiment defined in Definition 5 is indistinguishable from the ideal experiment.

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

Compute $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathsf{gen}(1^k)$ and set $pk := \mathrm{pk}$, $sk := \mathrm{sk}$. The adversary $\mathcal{A}$ outputs $(m_1, \ldots, m_\ell)$ after seeing the public key pk. Then for all $i \in [\ell]$, do the following: compute $(\textsc{msg}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$, $e_i \leftarrow \mathtt{enc}_{\mathrm{pk}}(w_i)$. Set $c_i := (\textsc{msg}_i, e_i)$. Return $(pk, c_1, \ldots, c_\ell, sk)$ to the adversary.

Hybrid $\mathbf{H}_{1,j}$: Let $j \in [0, \ldots, \ell]$. The hybrid is the same as the Hybrid $\mathbf{H}_0$ except the following:

For all $i \in [j]$, compute $(\textsc{msg}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\mathrm{recv}}(1^k)$. Then, compute $w_i \leftarrow \mathcal{S}_2^{\mathrm{recv}}(\textsc{msg}_i, \gamma_i, m_i)$.

We argue that for $j \in [1, \ldots, \ell]$, the hybrids $\mathbf{H}_{1,j-1}$ and $\mathbf{H}_{1,j}$ are computationally indistinguishable under the assumption that the OT satisfies adaptive receiver security. Assume there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids. For all $\mathcal{S}^{\mathrm{recv}}$, we next show how to construct $\mathcal{B}$ to distinguish between the Real experiment and the Ideal experiment as in Definition 7.

$\mathcal{B}$ internally simulates $\mathcal{A}$ and receives $(m_1, \ldots, m_\ell)$ from it. Then $\mathcal{B}$ carries out the following:

– run $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathsf{gen}(1^k)$, and set $pk := \mathrm{pk}$, $sk := \mathrm{sk}$;
– for all $i \in [j-1]$, use the OT simulator to obtain $(\textsc{msg}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\mathrm{recv}}(1^k)$ and $w_i \leftarrow \mathcal{S}_2^{\mathrm{recv}}(\textsc{msg}_i, \gamma_i, m_i)$. Compute $e_i \leftarrow \mathtt{enc}_{\mathrm{pk}}(w_i)$;
– for $i = j$, obtain the pair $(\textsc{msg}_i, w_i)$, where the pair is generated by either $\mathcal{S}^{\mathrm{recv}} = (\mathcal{S}_1^{\mathrm{recv}}, \mathcal{S}_2^{\mathrm{recv}})$, i.e., $(\textsc{msg}_i, \gamma_i) \leftarrow \mathcal{S}_1^{\mathrm{recv}}(1^k)$ and $w_i \leftarrow \mathcal{S}_2^{\mathrm{recv}}(\textsc{msg}_i, \gamma_i, m_i)$, or generated by the OT scheme honestly, i.e., $(\textsc{msg}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$;
– for all $i \in [j+1, \ell]$, compute $(\textsc{msg}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$ and $e_i \leftarrow \mathtt{enc}_{\mathrm{pk}}(w_i)$.
– for all $i \in [\ell]$, set $c_i := (\textsc{msg}_i, e_i)$;
– Return $(pk, c_1, \ldots, c_\ell, sk)$ to the internally simulated $\mathcal{A}$.

Note that when $i = j$, if the pair $(\textsc{msg}_i, w_i)$ that $\mathcal{B}$ obtains are generated by $\mathcal{S}^{\mathrm{recv}} = (\mathcal{S}_1^{\mathrm{recv}}, \mathcal{S}_2^{\mathrm{recv}})$, then $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j}$, while if the pair $(\textsc{msg}_i, w_i)$ that $\mathcal{B}$ obtains are generated by the OT scheme honestly, $\mathcal{A}$ interacts with Hybrid $\mathbf{H}_{1,j-1}$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 7 for defining adaptive receiver security. This leads to a contradiction to our assumption that the OT has adaptive receiver security. Therefore, Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable.

Note that $\mathbf{H}_0$ is identical to $\mathbf{H}_{1,0}$. Since Hybrid $\mathbf{H}_{1,j-1}$ and Hybrid $\mathbf{H}_{1,j}$ are indistinguishable, as argued above, we also have that $\mathbf{H}_0$ is computationally indistinguishable from $\mathbf{H}_{1,\ell}$.

Hybrid $\mathbf{H}_2$: The hybrid is the same as Hybrid $\mathbf{H}_{1,\ell}$ except the following:

Compute $(\text{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, and set $pk := \text{pk}$. Compute $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\text{pk}, z)$. Compute $\text{sk} \leftarrow \widetilde{\mathtt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, and set $sk := \text{sk}$. We note that this is exactly the ideal experiment.

We argue that $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are computationally indistinguishable based on the security of the ℓ-RNCE scheme. Assume there is an adversary $\mathcal{A}$ who can distinguish between the two hybrids. We next show how to construct an adversary $\mathcal{B}$ that can break the security of the ℓ-RNCE scheme.

$\mathcal{B}$ receives pk, and internally runs $\mathcal{A}$. Upon receiving $(m_1, \ldots, m_\ell)$ from $\mathcal{A}$, $\mathcal{B}$ computes $(\text{MSG}_i, w_i) \leftarrow \mathsf{OT1}(m_i)$ for all $i \in [\ell]$ and outputs $(w_1, \ldots, w_\ell)$ to its challenger. Upon receiving from its challenger $(e_1, \ldots, e_\ell, \text{sk})$, $\mathcal{B}$ sets $c_i := (\text{MSG}_i, e_i)$ for all $i \in [\ell]$. $\mathcal{B}$ returns $(pk, c_1, \ldots, c_\ell, sk)$ to $\mathcal{A}$ and returns $\mathcal{A}$'s output as its own output.

When $\mathcal{B}$'s received tuple $(\text{pk}, e_1, \ldots, e_\ell, \text{sk})$ is generated by $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ honestly, then the internally simulated $\mathcal{A}$ interacts with $\mathbf{H}_{1,\ell}$. On the other hand, when $\mathcal{B}$'s received tuple is generated by $(\widetilde{\mathtt{gen}}, \widetilde{\mathtt{enc}}, \widetilde{\mathtt{rev}})$, i.e, $(\text{pk}, z) \leftarrow \widetilde{\mathtt{gen}}(1^k)$, $(e_1, \ldots, e_\ell, z') \leftarrow \widetilde{\mathtt{enc}}(\text{pk}, z)$, and $\text{sk} \leftarrow \widetilde{\mathtt{rev}}(z', \{e_i, w_i\}_{i\in[\ell]})$, $\mathcal{A}$ interacts with $\mathbf{H}_2$. Based on the assumption that $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, we can conclude that $\mathcal{B}$ can distinguish Ideal from Real as in Definition 8. This contradicts the security of the ℓ-RNCE scheme. Therefore, Hybrids $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable.

Based on the above argument we have $\mathbf{H}_0$ and $\mathbf{H}_{1,\ell}$ are indistinguishable, and $\mathbf{H}_{1,\ell}$ and $\mathbf{H}_2$ are indistinguishable. Therefore $\mathbf{H}_0$ and $\mathbf{H}_2$ are indistinguishable. Note that Hybrid $\mathbf{H}_2$ is exactly the ideal experiment, and $\mathbf{H}_0$ is the real experiment. We now have that the ideal and the real experiments are indistinguishable. This implies that the scheme satisfies adaptive security.

References

1. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press (2009)
2. Barak, B., Haitner, I., Hofheinz, D., Ishai, Y.: Bounded Key-Dependent Message Security. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 423–444. Springer, Heidelberg (2010)
3. Beaver, D.: Plug and Play Encryption. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 75–89. Springer, Heidelberg (1997)
4. Beaver, D., Haber, S.: Cryptographic Protocols Provably Secure against Dynamic Adversaries. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 307–323. Springer, Heidelberg (1993)
5. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: ACM Conference on Computer and Communications Security, pp. 784–796 (2012)
6. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: 28th Annual ACM Symposium on Theory of Computing (STOC), pp. 639–648. ACM Press (1996)
7. Canetti, R., Halevi, S., Katz, J.: Adaptively-Secure, Non-interactive Public-Key Encryption. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 150–168. Springer, Heidelberg (2005)

8. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th ACM STOC Annual ACM Symposium on Theory of Computing (STOC), pp. 494–503. ACM Press (2002)
9. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved Non-committing Encryption with Applications to Adaptively Secure Protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 287–302. Springer, Heidelberg (2009)
10. Damgård, I., Nielsen, J.B.: Improved Non-committing Encryption Schemes Based on a General Complexity Assumption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 432–450. Springer, Heidelberg (2000)
11. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: 41st Annual ACM Symp. on Theory of Computing (STOC), pp. 169–178. ACM Press (2009)
12. Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), `http://crypto.stanford.edu/craig`
13. Gentry, C., Halevi, S., Vaikuntanathan, V.: i-Hop Homomorphic Encryption and Rerandomizable Yao Circuits. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 155–172. Springer, Heidelberg (2010)
14. Jarecki, S., Lysyanskaya, A.: Adaptively Secure Threshold Cryptography: Introducing Concurrency, Removing Erasures (Extended Abstract). In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 221–242. Springer, Heidelberg (2000)
15. Katz, J., Ostrovsky, R.: Round-Optimal Secure Two-Party Computation. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 335–354. Springer, Heidelberg (2004)
16. Lindell, Y., Pinkas, B.: A proof of security of Yao's protocol for two-party computation. Journal of Cryptology 22(2), 161–188 (2009)
17. Nielsen, J.B.: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002)
18. Rivest, R., Adleman, L., Dertouzos, M.: On data banks and privacy homomorphisms. In: Foundations of Secure Computation, pp. 169–177. Academic Press (1978)
19. Yao, A.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (FOCS), pp. 162–167. IEEE (1986)