Circuit-Private Multi-key FHE

Wutichai Chongchitmate[1(✉)] and Rafail Ostrovsky[1,2]

[1] Department of Computer Science, University of California, Los Angeles, CA, USA
{wutichai,rafail}@cs.ucla.edu
[2] Department of Mathematics, University of California, Los Angeles, CA, USA

Abstract. Multi-key fully homomorphic encryption (MFHE) schemes allow polynomially many users without trusted setup assumptions to send their data (encrypted under different FHE keys chosen by users independently of each other) to an honest-but-curious server that can compute the output of an arbitrary polynomial-time computable function on this joint data and issue it back to all participating users for decryption. One of the main open problems left in MFHE was dealing with malicious users without trusted setup assumptions. We show how this can be done, generalizing previous results of circuit-private FHE. Just like standard circuit-private FHE, our security model shows that even if both ciphertexts and public keys of individual users are not well-formed, no information is revealed regarding the server computation—other than that gained from the output on some well-formed inputs of all users. MFHE schemes have direct applications to server-assisted multiparty computation (MPC), called *on-the-fly* MPC, introduced by López-Alt et al. (STOC '12), where the number of users is not known in advance. In this setting, a poly-time server wants to evaluate a circuit C on data uploaded by multiple clients and encrypted under different keys. Circuit privacy requires that users' work is independent of $|C|$ held by the server, while each client learns nothing about C other than its output. We present a framework for transforming MFHE schemes with no circuit privacy into maliciously circuit-private schemes. We then construct 3-round on-the-fly MPC with circuit privacy against malicious clients in the plain model.

Keywords: Multi-key · Fully homomorphic encryption · Computing on encrypted data · Malicious setting · Server-assisted MPC

R. Ostrovsky—Research supported in part by NSF grant 1619348, US-Israel BSF grant 2012366, by DARPA Safeware program, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. The views expressed are those of the authors and do not reflect position of the Department of Defense or the U.S. Government.

S. Fehr (Ed.): PKC 2017, Part II, LNCS 10175, pp. 241–270, 2017.
DOI: 10.1007/978-3-662-54388-7_9

1 Introduction

The multi-key fully homomorphic encryption scheme (MFHE), introduced by López-Alt et al. [17], allows homomorphic computation on inputs encrypted with different public keys. They construct a MFHE under the ring learning with errors (RLWE) assumption, the decisional small polynomial ratio (DSPR) assumption, and circular security of a multi-key homomorphic encryption scheme $\mathcal{E}_{SH}$ based on a variant of NTRU homomorphic encryption. In this paper we construct a MFHE scheme that satisfies circuit privacy in the malicious setting, where public keys and ciphertexts are not guaranteed to be well-formed. We also present a framework for transforming multi-key homomorphic encryption schemes without circuit privacy or fully homomorphic property into maliciously circuit-private MFHE. We then demonstrate an instantiation of this framework using a modified scheme based on MFHE in [17] without adding further assumptions.

As in [21], we only consider the plain model. In the common reference string (CRS) model, the malicious case can be reduced to the semi-honest case by adding non-interactive zero-knowledge (NIZK) arguments that public key and ciphertext pairs are well-formed. Though, even in this case, difficulties can arise, as the security needs to hold when the pairs are in the support of honestly generated ones, but with different distributions—as discussed in [11].

In [17], the MFHE scheme is used to construct *on-the-fly* multiparty computation (MPC), which can perform arbitrary, dynamically chosen computation on arbitrary sets of users chosen on-the-fly. This construction allows each *client* user to encrypt data without knowing the identity or the number of other clients in the system. The *server* can select any subsets of clients, and perform an arbitrary function on the encrypted data without further input from the selected clients (and without learning clients' inputs). The encrypted result is then broadcast to the clients who cooperate in the retrieval of the output using (short) MPC protocol. Thus, most computation is done by the server while the decryption phase is independent of both the function computed and the total number of parties in the system. The resulting protocol is a five-round on-the-fly MPC secure against *semi-malicious* users [3], which follows the protocol but chooses random coins from an arbitrary distribution. The protocol can be strengthened against malicious adversaries in the CRS model using NIZK arguments without an increase in the number of rounds.

In this paper we construct a three-round on-the-fly MPC with circuit privacy against malicious users in the plain model. Specifically, all players send their inputs to the server, which performs the computation and sends the results back to all users, who then decrypt the result in one round. Since there is no way to enforce which function the server will compute, we assume that the server is honest but curious. As with our MFHE, the circuit privacy is guaranteed against unbounded malicious adversaries corrupting any number of clients. We also note that a variant of circuit privacy can be achieved in [17] construction by allowing the server to participate in the decryption phase MPC described above with its encrypted result as an input. However, our construction allows the server to minimize its interaction with the clients to only two rounds

(i.e., one message from client to server and one broadcast back to client). After the server sends its output back to the clients, the clients communicate with one another in only one additional round in order to decrypt the output. Since we use multi-key homomorphic encryption from [17] as the base of our construction, we also require the number of key pairs or users to be known is advance as in their protocol.

To summarize, our main theorems are as follows:

Theorem 1 *(informal).* *Assuming that there exists a privately expandable multi-key homomorphic encryption scheme, then there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

Theorem 2 *(informal).* *Assuming RLWE and DSPR assumptions, and circular security of $\mathcal{E}_{SH}$, there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

Theorem 3 *(informal).* *Assuming the preconditions of Theorems 1 or 2 hold, there exists a three-round on-the-fly MPC protocol where each client $i \in [U]$ in the system holds x_i, and the server chooses a circuit C with $N < U$ inputs and a subset $V \subseteq [U]$ with $|V| = N$. Only the clients in V learn $C(\{x_i\}_{i \in V})$ (but nothing else, not even $|C|$), and the server learns nothing about $\{x_i\}_{i \in [U]}$.*

1. *The privacy guarantee for clients is indistinguishability-based computational privacy against malicious adversaries corrupting $t < N$ clients and honest-but-curious servers.*
2. *The privacy guarantee for the server is based on unbounded simulation (against possibly unbounded clients).*

We note that condition 2 is incomparable with standard simulation framework as it requires stronger (i.e., information-theoretic) guarantees, but also unbounded simulation. As discussed in [21], this is unavoidable, even for single maliciously circuit-private FHE.

1.1 Previous Work

Multi-key FHE. As stated above, [17] introduces the concept of MFHE and constructs this scheme based on a variant of the NTRU encryption scheme under the RLWE and DSPR assumptions. The work of [7] gives an alternate construction based on [12], the FHE scheme under the LWE assumption. While their construction only relies on standard assumption such as LWE, it requires an additional set up step, equivalent to the CRS model. A recent work of [20] simplifies the construction of [7], and adds a threshold decryption protocol which is used to construct two-round MPC in the CRS model.

Circuit Privacy in FHE. In the semi-honest setting, where public keys and ciphertexts are supported by properly generated pairs, circuit privacy has been considered in [10,25], with the latter using Yao's garbled circuit. The generalization in [11] combines two HE schemes—one compact fully homomorphic and the other semi-honestly circuit-private—into compact semi-honestly circuit-private FHE.

The malicious setting has been addressed in the context of oblivious transfer (OT) [1,13]. The work of [15] constructs maliciously circuit-private HE for a class of depth-bounded branching programs by iteration from leaves of a branching program.

Finally, the work of [21] devises a framework for transforming single-key FHE schemes with no circuit privacy into maliciously circuit-private ones. They use techniques akin to Gentry's bootstrapping [10] and semi-honestly circuit-private HE constructions [1,11] combining FHE schemes with maliciously circuit-private HE schemes.

One-Round OT. Several definitions of OT security have been suggested—such as a general framework for defining two-party computation [5]. The work of [1] proposes a definition for one-round (2 messages) OT using unbounded simulation, which implies information theoretic security for sender, and demonstrates a construction based on the DDH assumption. In [15], Ishai and Paskin construct a one-round OT with perfect sender privacy based on the DJ homomorphic encryption scheme [8] in the semi-honest setting.

On-the-Fly MPC. In standard MPC protocols, the computational and communication complexities of each party depend on the circuit being computed. Thus, it is difficult to construct on-the-fly MPC, where only the server performs most of the computation, while the clients compute very little and do so independent of the circuit. This idea is explored in the work of [14,16]. However, the complexity of clients in the former protocol is still proportional to the size of the circuit, while the latter is only for a small class of functions.

A line of work uses single-key FHE schemes [3,10] by running a short MPC protocol to compute a joint public key and secretly shared corresponding secret key. However, this approach does not capture the dynamic and non-interactive properties of on-the-fly MPC. As mentioned above, López-Alt et al. [17] constructed on-the-fly MPC from multi-key FHE. However, their version is only secure against semi-malicious adversaries unless additional trusted setup assumptions are made.

Circuit Privacy in MPC. Private function evaluation (PFE) is a special case of MPC, where one party holds a function or circuit as an input. PFE follows immediately from MPC by evaluating a universal circuit and taking a circuit one wants to compute as an input. However, the known universal circuits have high complexity, namely, $\mathcal{O}(g^5)$ for arithmetic circuits [23] and $\mathcal{O}(g \log g)$ for Boolean circuits [24] for the class of circuits with at most g gates. This approach also does not hide the size of the circuits evaluated. Previous work [18,19] has constructed more efficient implementation of PFEs, even against an active adversary [19].

Comparison of MPC Protocols from MFHE. The following table illustrates the comparison between our results and other MPC protocols constructed from MFHE. Note that their securities are in different models, and thus are not directly comparable (Table 1).

Table 1. Comparison of MPC protocols from MFHE

Construction	Round	Adversary	Setup	Server-assisted	Circuit privacy
[17]	5	Semi-honest	No	Yes	No
[17]	5	Malicious	Yes	Yes	No
[20]	2	Malicious	Yes	No	No
This work	3	Malicious	No	Yes	Yes

1.2 Our Techniques

We now give an overview of our main construction of circuit-private MFHE in three steps:

Step 1. The first step is to define the main new ingredient of our construction, the *privately expandable* multi-key homomorphic encryption scheme. It is a multi-key HE together with efficient algorithms Expand such that, given a list of public keys and an encryption with respect to one of the keys, the output is a homomorphic encryption that does not depend on which key it was previously encrypted with. We note that in a standard construction of MFHE, a ciphertext may reveal which key is used to encrypt it. This information may persist even after homomorphic evaluation, thus revealing the structure of the evaluating program. Our new property allows the scheme to hide the source of the encryption used at each node of the branching program from an adversary, therefore hiding the branching program itself when combined with the technique in [15].

We show how to construct a privately expandable multi-key HE scheme from the multi-key somewhat homomorphic encryption scheme defined in [17]. The main idea is as follows: first, we re-randomize a given ciphertext to be statistically indistinguishable from a fresh ciphertext using algebraic properties of the scheme. We then show how to add encryptions of zero with respect to each of the other keys, and show how to homomorphically decrypt the result to get a "low-level" ciphertext. In fact, we note that our techniques are applicable to other known multi-key FHE schemes as well, such as in [20] to obtain a privately expandable multi-key FHE.

Step 2. The next step is to construct maliciously circuit-private multi-key HE for a class of depth-bounded branching programs. A (deterministic binary) branching program is represented by a directed acyclic graph whose nonterminal nodes with outdegree 2 are labeled with indices in $[n]$, while terminal nodes with outdegree 0 and edges are labeled with 0 or 1. An input $x \in \{0,1\}^n$ naturally induces a unique path from a distinguished initial node to a terminal node, whose label determines $P(x)$. Any logspace or NC function can be computed by polynomial size branching programs. We inductively compute a ciphertext for each node from terminal nodes upward. Given a ciphertext of each bit of $x \in \{0,1\}^n$, encrypted with different public keys, we expand the ciphertexts to hide public keys it was originally encrypted with. We use private expandability to

homomorphically compute ciphertext at each node with a key-hiding ciphertext indistinguishable from a fresh one. Thus, each ciphertext reveals nothing about the path leading to its corresponding node along the branching program, including which bit each node uses to decide its path. Therefore, the output, which is the ciphertext corresponding to the root, contains no information about the program.

The protocol above is secure against semi-honest adversaries. We then show how to modify the protocol to achieve security against malicious adversaries. We use single-key malicious circuit-private FHE and a modified validation circuit from [21], generalizing their techniques. The server (homomorphically) verifies that public keys and ciphertexts received are well-formed. This guarantees that each corrupted party uses proper public key and ciphertext, independent of other parties. Since we can verify before expanding the ciphertexts, we can use single-key FHE instead of multi-key.

Step 3. In this step we finally combine the protocol from the previous step with compact MFHE with no circuit privacy to get maliciously circuit-private MFHE. We modify the framework in [21] and obtain a framework for multi-key HE. To evaluate a given circuit, we first use MFHE with no circuit privacy to evaluate. Then we homomorphically decrypt the output using maliciously circuit-private HE that can evaluate the decryption function. Then we homomorphically decrypt to the original compact MFHE output, and only return it if public keys and ciphertexts are well-formed. This can be checked homomorphically similarly to the previous step. Using MFHE from [17] for instantiation, we get a maliciously circuit-private MFHE scheme based on RLWE and DSPR assumptions.

Application. Finally, we construct an on-the-fly MPC with circuit privacy from the result of the last step. Unlike [17], we consider the plain model with no setup assumptions and malicious adversaries corrupting an arbitrary number of clients. Along the way, we also construct a one-round 1-out-of-2 OT that is secure against malicious receivers with information theoretic security by augmenting a known construction that is only secure against semi-honest receivers with circuit-private FHE. Finally, by using a garbling scheme and our OT protocol, we can reduce the number of rounds from the construction in [17] to three rounds, which is optimal even against semi-honest adversaries in the plain model. The idea of the third round is as follows: Instead of having the clients run an MPC protocol to decrypt the output, the server constructs a collection of garbled circuits that decrypts the output for each user. The clients create an OT query for each bit of their secret keys and send it to the server along with the ciphertext in the first round. The server then answers those queries with corresponding garbled input for the garbled circuit. Finally, each client decrypts and broadcasts their garbled inputs to all other clients to compute the final output from the garbled circuits by each client.

The security of our protocol is based on unbounded simulation for the server, which is necessary for circuit privacy as discussed in [15,21]. We note that it is

impossible to obtain ideal functionality definition due to the impossibility of any computationally bounded simulators extracting the input in one round (without trusted setup assumptions). Instead, we show the security for honest clients based on indistinguishability of the view of the malicious adversaries corrupting clients and the view of the honest-but-curious server.

2 Background

2.1 Notation

For positive integer $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. For a string $x \in \{0,1\}^*$, let $|x|$ denote its length. Let $\oplus$ denote bitwise XOR operation or bitwise addition modulo 2. For a distribution A, let $x \leftarrow A$ denote x is chosen according to a distribution A. For a finite set S, let $x \leftarrow S$ denote x is chosen uniformly from the set S. Let λ denote the security parameter. A function $f : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every constant $c > 0$, there exists $\lambda_0 \in \mathbb{N}$ such that $f(\lambda) \leq \lambda^{-c}$ for all $\lambda \geq \lambda_0$. Algorithms may be randomized unless stated otherwise. A PPT algorithm runs in probabilistic polynomial-time; otherwise, it is unbounded. For an algorithm A, let $y \leftarrow A(x; r)$ denote running A on input x with random coins r. If r is chosen uniformly at random, we denote $y \leftarrow A(x)$. For two distributions X, Y, $X \simeq^s Y$ means X and Y are statistically closed, i.e. $\Delta(X, Y)$ is negligible. For two distributions X, Y, $X \simeq^c Y$ means X and Y are computationally indistinguishable, i.e. for any PPT algorithm D, $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]|$ is negligible.

Setup vs. Plain Model. We say a protocol is in the *setup model* or the *common reference string (CRS) model* if every party has access to a common random string r that was ideally drawn from some publicly known distribution prior to the beginning of the protocol. Without such setup, we say a protocol is in the *plain model*.

Malicious vs. Honest-but-Curious Party. We say a party participating in a protocol is *honest-but-curious* if it follows the protocol, but may perform additional computation to learn more information than it should. We say a party is *(fully) malicious* if it deviates from the protocol arbitrarily.

Representation Models. In order to use a function or a program as an input of our algorithm, we consider a function represented by a string representation C. The correspondence between a program C and a function f it represents must be universally interpreted by an underlying representation model U. Formally, a *representation model* $U : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is a PPT algorithm that takes a input (C, x) and returns $f(x)$ for a function f represented by C. If (C, x) is syntactically malformed, we let $U(C, x) = 0$ for completeness. We let $|C|$ denote the size of program C as a string representation as opposed to the number of gates as a Boolean circuit.

2.2 Multi-key Homomorphic Encryption

We use the definition similar to the one defined in [17] with some modifications from [20,21]. We fix the order of public keys in Eval and secret keys in Dec, and allow the number of keys to be different from input size of the circuit. This definition better suits our definition of circuit privacy that we will define in the next section.

Definition 1 (Multi-key (Leveled) $(U, \mathcal{C})$-Homomorphic Encryption). *Let $\mathcal{C}$ be a class of circuits. A* multi-key (leveled) $(U, \mathcal{C})$-homomorphic *scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ is described as follows:*

- $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^d)$*: Given a security parameter λ (and the circuit depth d), outputs a public key pk and a secret key sk.*
- $c \leftarrow \mathsf{Enc}(pk, \mu)$*: Given a public key pk and a message μ, outputs a ciphertext c.*
- $\hat{c} \leftarrow \mathsf{Eval}(C, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$*: Given a (description of) a Boolean circuit C (of depth $\leq d$) along with a sequence of N public keys and n couples (I_i, c_i), each comprising of an index $I_i \in [N]$ and a ciphertext c_i, outputs an evaluated ciphertext $\hat{c}$.*
- $b := \mathsf{Dec}(sk_1, \ldots, sk_N, \hat{c})$*: Given a sequence of N secret keys $sk_1, \ldots, sk_N$ and a ciphertext $\hat{c}$, outputs a bit b.*

has the following properties:

- ***Semantic security****:* $(\mathsf{KeyGen}, \mathsf{Enc})$ *satisfies IND-CPA semantic security.*
- ***Correctness****: Let $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^d)$ for $i = 1, \ldots, N$. Let $x = x_1 \ldots x_n \in \{0,1\}^n$ and $C \in \mathcal{C}$ be a Boolean circuit of depth $\leq d$, $C : \{0,1\}^n \to \{0,1\}$. For $i = 1, \ldots, n$, let $c_i \leftarrow \mathsf{Enc}(pk_{I_i}, x_i)$ for some $I_i \in [N]$. Let $\hat{c} \leftarrow \mathsf{Eval}(C, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$. Then*

$$\mathsf{Dec}(sk_1, \ldots, sk_N, \hat{c}) = U(C, (x_1, \ldots, x_n)).$$

$\mathcal{E}$ is compact *if there exists a polynomial p such that $|\hat{c}| \leq p(\lambda, d, N)$ independent of C and n. If a scheme is multi-key $(U, \mathcal{C})$-homomorphic for the class $\mathcal{C}$ of all circuits (of depth $\leq d$), we call it a* multi-key (leveled) fully homomorphic (MFHE)*. A scheme $\mathcal{E}$ is* somewhat homomorphic *if it is leveled $(U, \mathcal{C})$-homomorphic for $d \leq d_{\mathsf{max}}(\lambda, N)$. A scheme $\mathcal{E}$ is* multi-hop *if an output of Eval can be used as an input as long as the sum of the depths of circuits evaluated does not exceed d.*

2.3 López-Alt, Tromer and Vaikuntanathan's Multi-key FHE Scheme

In [17], López-Alt et al. construct a multi-key compact leveled fully homomorphic encryption scheme. They first construct a multi-key leveled somewhat HE scheme $\mathcal{E}_{SH}$, then apply Gentry's bootstrapping [10]. The security of the scheme is based on the ring learning with error (RLWE) assumption, the decisional small polynomial ratio (DSPR) assumption, and the weak circular security of $\mathcal{E}_{SH}$.

Let $q = q(\lambda)$ be an odd prime integer. Let the ring $R = \mathbb{Z}[x]/\langle\phi\rangle$ for polynomial $\phi \in \mathbb{Z}[x]$ of degree $m = m(\lambda)$ and $R_q = R/qR$. Let χ be the B-bounded truncated discrete Gaussian distribution over R for $B = B(\lambda)$.

Definition 2 (Ring Learning With Error (RLWE) Assumption [4]). *The* (decisional) ring learning with error assumption $RLWE_{\phi,q,\chi}$ *states that for any* $l = \mathsf{poly}(\lambda)$,

$$\{(a_i, a_i \cdot s + e_i)\}_{i\in[l]} \simeq^c \{(a_i, u_i)\}_{i\in[l]}$$

where $s, e_i \leftarrow \chi$ *and* a_i, u_i *are sampled uniformly at random over* R_q.

Definition 3 (Decisional Small Polynomial Ratio (DSPR) Assumption [17]). *The* decisional small polynomial ration assumption $DSPR_{\phi,q,\chi}$ *says that it is hard to distinguish the following two distributions:*

- *a polynomial* $h := [2gf^{-1}]_q$, *where* $f', g \leftarrow \chi$ *such that* $f := 2f'+1$ *is invertible over* R_q *and* f^{-1} *is the inverse of* f *in* R_q.
- *a polynomial* u *sampled uniformly at random over* R_q.

We describe the multi-key leveled somewhat HE scheme here as follows.

$\mathsf{KeyGen}_{SH}(1^\lambda, 1^d)$:

1. For $i = 0, 1, \ldots, d$,
   (a) Sample $\tilde{f^i}, g^i \leftarrow \chi$ and compute $f^i := 2\tilde{f^i} + 1$. If f^i is not invertible in R_q, resample $\tilde{f^i}$.
   (b) Let $(f^i)^{-1}$ be the inverse of f^i in R_q.
   (c) Let $h_i := [2g^i(f^i)^{-1}]_{q_i} \in R_{q_i}$.
   (d) For $i \geq 1$, sample $\boldsymbol{s}^i_\gamma, \boldsymbol{e}^i_\gamma, \boldsymbol{s}^i_\zeta, \boldsymbol{e}^i_\zeta \leftarrow \chi^{\lceil \log q_i \rceil}$.
   (e) Let $\gamma^i := \left[h^i \boldsymbol{s}^i_\gamma + 2\boldsymbol{e}^i_\gamma + \mathsf{Pow}(f^{i-1})\right]_{q_i} \in R_{q_i}^{\lceil \log q_i \rceil}$
   and $\zeta^i := \left[h^i \boldsymbol{s}^i_\zeta + 2\boldsymbol{e}^i_\zeta + \mathsf{Pow}\left((f^{i-1})^2\right)\right]_{q_i} \in R_{q_i}^{\lceil \log q_i \rceil}$.
2. Output $pk = (h^0, \gamma^1, \ldots, \gamma^d, \zeta^1, \ldots, \zeta^d)$ and $sk = f^d \in R_{q_d}$.

$\mathsf{Enc}_{SH}(pk, \mu)$:

1. Parse $pk = h$. Sample $s, e \leftarrow \chi$.
2. Output $c = [hs + 2e + \mu]_{q_0} \in R_{q_0}$.

$\mathsf{Eval}_{SH}(C, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$:

1. For $i \in [N]$, parse $pk_i = (h_i, \gamma_i^1, \ldots, \gamma_i^d, \zeta_i^1, \ldots, \zeta_i^d)$
2. Given two ciphertexts $c, c' \in R_{q_i}$ associated with subsets of the public keys K, K', respectively. Let $c_0 = [c + c'] \in R_{q_i}$ and $K \cup K' = \{pk_{i_1}, \ldots, pk_{i_t}\}$. For $j = 1, \ldots, t$, compute

$$c_j = \left[\langle \mathrm{Bit}(c_{j-1}), \gamma^i_{i_j} \rangle\right]_{q_i} \in R_{q_i}$$

   Then let c_{add} be the integral vector closest to $(q_{i+1}/q_i) \cdot c_t$ such that $c_{add} = c_t \pmod 2$. Output $c_{add} \in R_{q_{i+1}}$ and the associated subset $K \cup K'$.

3. Given two ciphertexts $c, c' \in R_{q_i}$ associated with subsets of the public keys K, K', respectively. Let $c_0 = [c \cdot c'] \in R_{q_i}$ and $K \cup K' = \{pk_{i_1}, \ldots, pk_{i_t}\}$. For $j = 1, \ldots, t$,
   (a) If $pk_{i_j} \in K \cap K'$, compute

$$c_j = \left[\langle \mathrm{Bit}(c_{j-1}), \zeta_{i_j}^i \rangle\right]_{q_i} \in R_{q_i}$$

   (b) Otherwise, compute

$$c_j = \left[\langle \mathrm{Bit}(c_{j-1}), \gamma_{i_j}^i \rangle\right]_{q_i} \in R_{q_i}$$

   Then let c_{mult} be the integral vector closest to $(q_{i+1}/q_i) \cdot c_t$ such that $c_{mult} = c_t \pmod 2$. Output $c_{mult} \in R_{q_{i+1}}$ and the associated subset $K \cup K'$.

$\mathsf{Dec}_{SH}(sk_1, \ldots, sk_N, c)$:

1. For $i \in [N]$, parse $sk_i = f_i$.
2. Let $\mu_0 = [f_1 \ldots f_N \cdot c]_{q_d} \in R_{q_d}$.
3. Output $\mu' = \mu_0 \pmod 2$.

Remarks

1. In [17], a different notation for $\mathsf{Eval}_{SH}(C, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$ is used, namely, $\mathsf{Eval}_{SH}(C, (pk_1, c_1), \ldots, (pk_n, c_n))$. These two notations are equivalent when $N = n$ and $I_j = j$ for $j = 1, \ldots, n$. For brevity, we also use this notation under such conditions.
2. We also denote the evaluation on intermediate ciphertexts $\tilde{c}_1, \ldots, \tilde{c}_n$ associated with nonempty subsets of public keys $K_1, \ldots, K_n$, respectively, by $\mathsf{Eval}_{SH}(C, (K_1, \tilde{c}_1), \ldots, (K_n, \tilde{c}_n))$.

Theorem 4 [17]**.** *Assuming the DSPR and RLWE assumptions, and that the scheme* $\mathcal{E}_{SH} = (\mathsf{KeyGen}_{SH}, \mathsf{Enc}_{SH}, \mathsf{Eval}_{SH}, \mathsf{Dec}_{SH})$ *described above is weakly circular secure, then there exists a multi-key compact leveled fully homomorphic encryption scheme for* N *keys for any* $N \in \mathbb{N}$*, obtained by bootstrapping* $\mathcal{E}_{SH}$*.*

2.4 Circuit-Private Homomorphic Scheme

We describe the circuit privacy of single-key homomorphic encryption defined in [15,21]. In the next section we will define our multi-key variant based on this definition.

Definition 4. *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *denote a* $(U, \mathcal{C})$*-homomorphic encryption scheme. We say* $\mathcal{E}$ *is* (maliciously) circuit-private *if there exist unbounded algorithms* $\mathsf{Sim}(1^\lambda, pk^*, c_1^*, \ldots, c_n^*, b)$ *and deterministic* $\mathsf{Ext}(1^\lambda, pk^*, c^*)$ *such that for all* λ*,* pk^**,* $c_1^*, \ldots, c_n^*$*, and all programs* $C : \{0,1\}^n \to \{0,1\} \in (U, \mathcal{C})$*, the following holds:*

– *for* $i = 1, \ldots, n$, $x_i^* := \mathsf{Ext}(1^\lambda, pk^*, c_i^*)$
– $\mathsf{Sim}(1^\lambda, pk^*, c_1^*, \ldots, c_n^*, U(C, x_1^*, \ldots, x_n^*)) \simeq^s \mathsf{Eval}(1^\lambda, C, pk^*, c_1^*, \ldots, c_n^*)$

We say the scheme is semi-honestly circuit-private *if the above holds only for* well-formed $pk^* = pk$, $c_i^* = c_i$, *i.e.* $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $c_i \leftarrow \mathsf{Enc}(pk, x_i)$ *for some* $x_i \in \{0,1\}$, $i = 1, \ldots, n$.

Theorem 5 [21]. *Assume an FHE scheme with decryption circuits in* NC^1 *exists. There exists a maliciously circuit-private single-key fully homomorphic encryption scheme.*

2.5 Branching Program

Definition 5. *A* (binary) branching program P *over* $x = (x_1, \ldots, x_n)$ *is a tuple* $(G = (V, E), v_0, T, \psi_V, \psi_E)$ *such that*

– G *is a connected directed acyclic graph. Let* $\Gamma(v)$ *denote the set of children of* $v \in V$.
– v_0 *is an initial node of indegree 0.*
– $T \subseteq V$ *is a set of terminal nodes of outdegree 0. Any node in* $V \setminus T$ *has outdegree 2.*
– $\psi_V : V \to [n] \cup \{0,1\}$ *is a node labeling function with* $\psi_V(v) \in \{0,1\}$ *for* $v \in T$, *and* $\psi_V(v) \in [n]$ *for* $v \in V \setminus T$.
– $\psi_E : E \to \{0,1\}$ *is an edge labeling function, such that outgoing edges from each vertex is labeled by different values.*

The height *of* $v \in V$, *denoted height*(v), *is the length of the longest path from* v *to a node in* T. *The* length *of* P *is the height of* v_0.

On input x, $P(x)$ *is defined by following the path induced by* x *from* v_0 *to a node* $v_l \in T$, *where an edge* (v, v') *is in the path if* $x_{\psi_V(v)} \in \psi_E(v, v')$. *By the last property, such* v' *is unique. Then* $P(x) = \psi_V(v_l)$. *Similarly, we also define* $P_v(x)$ *by following that path from any node* $v \in V$ *instead of* v_0.

Definition 6. *A* layered *branching program of length* l *is a branching program* $P = (G = (V, E), v_0, T, \psi_V, \psi_E)$ *such that for any* $e = (v, v') \in E$, $height(v) = height(v') + 1$.

Every path from an initial node to a terminal node in a layered branching program has the same length. Every branching program can be efficiently transformed into a layered branching program of the same length [22]. For simplicity, we assume all branching programs are layered.

3 Privately Expandable Multi-key Homomorphic Encryption

In this section we will define the properties of multi-key homomorphic encryption which are required for the construction of multi-key circuit private HE for

branching programs discussed in the next section. Informally, private expandability allows masking of a ciphertext encrypted under a public key using other public keys in order to hide the key it was originally encrypted with. We then show how to modify the multi-key HE from [17] to achieve such property. We note that the multi-key HE from [20] can be modified to have this property in a similar way.[1] However, since it only works in the setup model, we cannot get a meaningful result in circuit privacy.

3.1 Private Expandability

We define an "expanded" ciphertext as one that associates with all public keys to be used in the evaluation algorithm. This notion is also used in [20]. However, expanded ciphertexts in [20] do not hide the original public key it is encrypted with. In both our construction and the one in [20], an expanded ciphertext can be thought of as a single-key homomorphic encryption ciphertext that can be decrypted with some function of all secret keys. In our case, it is the product of all secret keys; in the [20] case, it is the appending of all secret keys.

Definition 7. *A multi-key HE scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *is* privately expandable *if there exist polynomial time algorithms* $\widetilde{\mathsf{Expand}}, \widetilde{\mathsf{Eval}}, \widetilde{\mathsf{Dec}}$ *such that, for* $i = 1, \ldots, N$, $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda)$,

- *Let* $c \leftarrow \mathsf{Enc}(pk_i, \mu)$. *Then for any* $j \in [N]$,

$$\tilde{c} := \widetilde{\mathsf{Expand}}(pk_1, \ldots, pk_N, i, c) \simeq^s \widetilde{\mathsf{Expand}}(pk_1, \ldots, pk_N, j, \mathsf{Enc}(pk_j, \mu))$$

  and $\widetilde{\mathsf{Dec}}(sk_1, \ldots, sk_N, \tilde{c}) = \mu$
- *if for* $i = 1, \ldots, N$, $\widetilde{\mathsf{Dec}}(sk_1, \ldots, sk_N, \tilde{c}_i) = b_i$, *then*

$$\widetilde{\mathsf{Dec}}(sk_1, \ldots, sk_N, \widetilde{\mathsf{Eval}}(P, pk_1, \ldots, pk_N, \tilde{c}_1, \ldots, \tilde{c}_l)) = P(b_1, \ldots, b_l).$$

We sometimes replace Eval and Dec with $\widetilde{\mathsf{Eval}}$ and $\widetilde{\mathsf{Dec}}$, respectively, and denote $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Expand}, \mathsf{Eval}, \mathsf{Dec})$ a privately expandable HE scheme if Expand, Eval and Dec satisfy the above conditions.

3.2 Privately Expandable Multi-key HE Based on LTV Encryption Scheme

In [17], Lopez et al. constructed a multi-key FHE scheme with security based on ring learning with error assumption (RLWE) and decisional small polynomial ration assumption (DSPR) by further assuming circular security. We will show that we can modify the scheme to be privately expandable by constructing $\widetilde{\mathsf{Expand}}, \widetilde{\mathsf{Eval}}, \widetilde{\mathsf{Dec}}$ without additional assumption.

Let $\mathcal{E}_{SH} = (\mathsf{KeyGen}_{SH}, \mathsf{Enc}_{SH}, \mathsf{Eval}_{SH}, \mathsf{Dec}_{SH})$ be the multi-key somewhat homomorphic scheme given in [17] defined in the previous section.

[1] See the full version [6] of this paper for details.

A ciphertext of $\mathcal{E}_{SH}$ is a polynomial in $R_q = \mathbb{Z}_q[x]/(x^n+1)$ which can be represented by a vector in $\mathbb{Z}_q^n$. In this scheme, N must be known in advance. We choose $n = N^{1/\epsilon'}$, $q = 2^{n^\epsilon}$ for some $\epsilon' < \epsilon$. Thus, $q = 2^{N^\delta}$ for $\delta > 1$. We need to use a bootstrappable somewhat homomorphic version instead of a bootstrapped FHE as we need its multi-hop property while we only need to evaluate low depth circuits. Let $t \in \mathbb{N}$ and $\mathcal{U}_t$ be a discrete uniform distribution on $\{0, \ldots, t\}$, which can be sampled in time $O(\log t)$. We define

$\widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, i, c)$:

1. For each $j \in \{1, \ldots, N\}$
   - Parse $pk_j = h_j$.
   - Let $s_j, e_j \leftarrow \mathcal{U}_t^n$.
   - Let $c_j = h_j s_j + 2e_j$
2. Output $\hat{c} = c + \sum_{j=1}^{N} c_j$.

The following lemma is a variant of the smudging lemma in [3]:

Lemma 1. *Let $a_1, a_2 \in \mathbb{Z}^n$ be B-bounded. Then $\Delta(a_1+b, a_2+b) \leq 4nB/t$ where $b \leftarrow \mathcal{U}_t^n$. If t is superpolynomial in λ, then they are statistically indistinguishable.*

Proof. Let $c_1, c_2 \in \mathbb{Z}$ be corresponding entries in a_1 and a_2, respectively. Then $|c_1 - c_2| \leq 2B$. Thus, $\Delta(c_1 + \mathcal{U}_t, c_2 + \mathcal{U}_t) \leq 4B/t$. Therefore, $\Delta(a_1 + b, a_2 + b) \leq 4nB/t$. Since n and B are polynomial in λ, $\Delta(a_1 + b, a_2 + b)$ is negligible for superpolynomial t. □

We apply the above lemma to get the following result.

Lemma 2. *Let $(pk_k, sk_k) \leftarrow \mathsf{KeyGen}_{SH}(1^\lambda, 1^d)$ for $k = 1, \ldots, N$. For $i \in [N]$, let $c \leftarrow \mathsf{Enc}_{SH}(pk_i, \mu)$. Let $t \leq \frac{1}{18}(\frac{q}{N(nB)^N})$. Then*

$$\hat{c} := \widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, i, c) \simeq^s \widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, j, \mathsf{Enc}_{SH}(pk_j, \mu))$$

for any $j \in [N]$, and $\mathsf{Dec}_{SH}(sk_1, \ldots, sk_N, \hat{c}) = \mu$.

Proof. Suppose t is superpolynomial. Then for any $s, e \leftarrow \chi$ and $s_i, e_i \leftarrow \mathcal{U}_t^n$, $[s + s_i] \simeq^s [s_i]$ and $[e + e_i] \simeq^s [e_i]$ by Lemma 1. Thus, for $c = h_i s + 2e + m$, we have $[c + (h_i s_i + 2e_i)] \simeq^s [m + (h_i s_i + e_i)]$. Then

$$\widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, i, c) \simeq^s [m + \sum_{k \in [N]} (h_k s_k + 2e_k)].$$

By the same reason,

$$\widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, j, \mathsf{Enc}_{\mathcal{E}}(pk_j, \mu)) \simeq^s [m + \sum_{k \in [N]} (h_k s_k + 2e_k)].$$

Therefore, they are statistically indistinguishable.

Now let $\hat{c} = m + \sum_{j\in[N]}(h_j s_j + 2e_j)$ where s_j, e_j bounded by t. For each $j \in [N]$, $f_j(h_j s_j + 2e_j) = 2(g_j s_j + f_j e_j)$ is bounded by $E := 2nBt + 2nB(2t+1) = 2nB(3t+1) \leq 8nBt$. Then for $f = f_1 \dots f_N$,

$$f\hat{c} = fm + \sum_{j\in[N]}\Big(\prod_{k\in[N]\setminus\{j\}} f_k\Big) f_j(h_j s_j + 2e_j)$$

is bounded by $(nB)^N + N(nB)^{N-1}E \leq 9N(nB)^N t$, which can be decrypted if it is less than $q/2$. Thus, for $t \leq \frac{1}{18}(\frac{q}{N(nB)^N})$, the correctness follows from that of LTV scheme. Note that as $q = 2^{N^\delta} = (2^{N^{\delta-1}})^N$, t is still superpolynomial in N and thus λ. □

Lemma 3 (implied from [17]). *For any $C > 0$, for sufficiently large $\lambda, N = N(\lambda) \in \mathbb{N}$, there exists a multi-key somewhat homomorphic encryption scheme for N keys and circuits of depth $d \geq Cd_{\mathsf{Dec}}$ where d_{Dec} is the depth of its decryption circuit.*

The depth of circuits that can be evaluated is important here because the construction in the next section will require that the scheme can perform evaluation twice.

Now let t satisfy the above condition. Let $d_0 = d_{\mathsf{Dec}}$ and $d \geq 3d_0 + 2$. We define a scheme $\mathcal{F} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Enc}_{\mathcal{F}}, \mathsf{Expand}_{\mathcal{F}}, \mathsf{Eval}_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$ as follows:

$\mathsf{KeyGen}_{\mathcal{F}}(1^\lambda, 1^d)$:

1. Let $(pk_0, sk_0) \leftarrow \mathsf{KeyGen}_{SH}(1^\lambda, 1^{d_0})$ and $(pk_{\mathcal{E}}, sk_{\mathcal{E}}) \leftarrow \mathsf{KeyGen}_{SH}(1^\lambda, 1^{d+d_0})$
2. Let $f_{sk} = \mathsf{Enc}_{SH}(pk_{\mathcal{E}}, sk_0)$
3. Output $pk = (pk_0, pk_{\mathcal{E}}, f_{sk})$ and $sk = sk_{\mathcal{E}}$.

$\mathsf{Enc}_{\mathcal{F}}(pk, \mu)$:

1. Parse $pk = (pk_0, pk_{\mathcal{E}}, f_{sk})$.
2. Output $\mathsf{Enc}_{SH}(pk_0, \mu)$.

$\mathsf{Expand}_{\mathcal{F}}(pk_1, \dots, pk_N, i, c)$:

1. Parse $pk_j = (pk_{0,j}, pk_{\mathcal{E},j}, f_{sk,j})$.
2. Let $\hat{c} = \widetilde{\mathsf{Expand}}^{t}(pk_{0,1}, \dots, pk_{0,N}, i, c)$
3. Output $\tilde{c} = \mathsf{Eval}_{SH}(\mathsf{Dec}_{SH}(\cdot, \hat{c}), (pk_{\mathcal{E},1}, f_{sk,1}), \dots, (pk_{\mathcal{E},N}, f_{sk,N}))$.

$\mathsf{Eval}_{\mathcal{F}}(P, pk_1, \dots, pk_N, \tilde{c}_1, \dots, \tilde{c}_n)$:

1. Parse $pk_j = (pk_{0,j}, pk_{\mathcal{E},j}, f_{sk,j})$.
2. Let $K = \{pk_1, \dots, pk_N\}$
3. Output $\tilde{c} = \mathsf{Eval}_{SH}(P, (K, \tilde{c}_1), \dots, (K, \tilde{c}_n))$.

$\mathsf{Dec}_{\mathcal{F}}(sk_1, \ldots, sk_N, \tilde{c})$:

1. Parse $sk_j = sk_{\mathcal{E},j}$.
2. Output $\mu' = \mathsf{Dec}_{SH}(sk_{\mathcal{E},1}, \ldots, sk_{\mathcal{E},N}, \tilde{c})$.

Note that $\mathsf{Dec}_{\mathcal{F}}$ has the same size as Dec_{SH}.

Lemma 4. *The scheme* $\mathcal{F} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Enc}_{\mathcal{F}}, \mathsf{Expand}_{\mathcal{F}}, \mathsf{Eval}_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$ *above is a privately expandable multi-key compact somewhat homomorphic scheme that can evaluate circuits up to a depth of* $2d_0 + 2$.

Proof. The security and compactness of $\mathcal{F}$ follows directly from that of $\mathcal{E}$. By Lemma 2, for $c = \mathsf{Enc}_{\mathcal{F}}(pk_i, \mu)$, $\tilde{c} = \mathsf{Expand}_{\mathcal{F}}(pk_1, \ldots, pk_N, i, c)$ is a level-d_0 encryption of μ associated with $K = \{pk_{\mathcal{E},1}, \ldots, pk_{\mathcal{E},N}\}$ under scheme $\mathcal{E}$. Thus, the correctness of evaluation and decryption of $\mathcal{F}$ follows from that of $\mathcal{E}$.

Also, by Lemma 2, $\hat{c} \simeq^s \widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, j, \mathsf{Enc}_{\mathcal{E}}(pk_j, \mu))$. Then the result of homomorphically decrypting both sides gives $\tilde{c} \simeq^s \widetilde{\mathsf{Expand}}_{\mathcal{F}}(pk_1, \ldots, pk_N, j, \mathsf{Enc}(pk_j, \mu))$. Since each $f_{sk,j}$ are level 1 encryption under $\mathcal{E}$, the output of $\widetilde{\mathsf{Expand}}_{\mathcal{F}}$ is of level d_0. Thus, we can further evaluate circuits up to depth $2d_0 + 2$ as required. □

Remarks

1. Recent results of Albrecht et al. [2] give a sub-exponential (in λ) attack on DSPR assumption when q is super-polynomial, which is required in [17].
2. Since our protocol is also based on $\mathcal{E}_{SH}$ with super-polynomial q, security parameter and other variables involved need to be chosen carefully to remain secure under such attack.
3. Another possible solution is to use the recent technique in [9] to construct a privately expandable scheme without adding superpolynomial-size errors to the ciphertexts. However, careful application of this technique is required in order to guarantee that the resulting scheme is both privately expandable and correctly decryptable. We leave this as an open problem.

4 Circuit-Private Multi-key HE for Branching Programs

We first define the multi-key version of circuit privacy given in the previous section.

Definition 8. *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *denote a multi-key* $(U, \mathcal{C})$*-homomorphic encryption scheme. We say* $\mathcal{E}$ *is* (maliciously) circuit-private *if there exist unbounded algorithms* $\mathsf{Sim}(1^\lambda, (pk_1^*, c_1^*), \ldots, (pk_n^*, c_n^*), b)$ *and deterministic* $\mathsf{Ext}(1^\lambda, pk^*, c^*)$ *such that for all* λ, $pk_1^*, \ldots, pk_N^*$, $I_1, \ldots, I_n$, $c_1^*, \ldots, c_n^*$, *and all programs* $C : \{0,1\}^n \to \{0,1\} \in (U, \mathcal{C})$, *the following holds:*

- *for* $i = 1, \ldots, n$, $x_i^* := \mathsf{Ext}(1^\lambda, pk_{I_i}^*, c_i^*)$
- $\mathsf{Sim}(1^\lambda, (pk_1^*, \ldots, pk_N^*), (I_1, c_1^*), \ldots, (I_n, c_n^*), U(C, x_1^*, \ldots, x_n^*))$
  $\simeq^s \mathsf{Eval}(1^\lambda, C, (pk_1^*, \ldots, pk_N^*), (I_1, c_1^*), \ldots, (I_n, c_n^*))$

We say the scheme is semi-honestly circuit-private *if the above holds only for* well-formed $pk_{I_i}^* = pk_{I_i}$, $c_i^* = c_i$ *pairs, i.e.* $(pk_{I_i}, sk_{I_i}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $c_i \leftarrow \mathsf{Enc}(pk_{I_i}, x_i)$.

In this section we construct a circuit-private multi-key HE for a class $\mathcal{C}$ of (depth bound) branching programs. As discussed above, the difficulty in the multi-key setting is that each decision one makes while traversing a branching program is dependent on its corresponding input bit, which in turn is dependent on which public key it is encrypted with. Using such encryption may reveal bit positions of the path it takes to reach a terminal node. Using a privately expandable multi-key HE scheme (previous section) solves this problem. Another implication of private expandability is that we can generate a fresh expanded encryption of bit b that is indistinguishable from an expanded encryption of any given encryption of b. This allows us to construct a simulator for circuit privacy, given an output bit.

We first give a construction that is secure against semi-honest adversaries where each pair of public key and ciphertext is correctly generated. The intuition behind this construction is as follows: given a branching program P, we assign to each node of P a ciphertext that multi-key decrypt to an output computed with that node as a root. Thus, the ciphertext assigned to the actual root will decrypt to the actual output. In order to construct such a ciphertext (called *label* below), we privately expand the input corresponding to a position given by ψ_V of that node in order to hide the position. We then homomorphically construct a ciphertext encrypting each bit of its child that is specified by the encrypted input (without knowing the input bit). Note that this result will be an encryption of an encryption of the output. Finally, we homomorphically decrypt it twice using HE evaluation. We show that, in this case, the output can be simulated knowing the public keys, ciphertext, and the output; it is thus independent of the program being evaluated.

We then show that we can augment this construction to handle malicious public key and ciphertext pairs using a single-key circuit-private FHE since the evaluated output does not depend on the branching program, unlike in the general case.

4.1 Semi-honest Model

Let $\mathcal{F} = (\mathsf{KeyGen}_\mathcal{F}, \mathsf{Enc}_\mathcal{F}, \mathsf{Expand}_\mathcal{F}, \mathsf{Eval}_\mathcal{F}, \mathsf{Dec}_\mathcal{F})$ be a privately expandable multi-hop multi-key compact somewhat homomorphic scheme that can evaluate circuit up to depth $2d_0 + 2$ where d_0 is the depth of $\mathsf{Dec}_\mathcal{F}$. Let l be the length of branching programs, and let $p(\lambda, l)$ be a polynomial to be specified later. Let $\mathsf{Dec}_\mathcal{F}^2(sk_1, \ldots, sk_N, c) = \mathsf{Dec}_\mathcal{F}(sk_1, \ldots, sk_N, \mathsf{Dec}_\mathcal{F}(sk_1, \ldots, sk_N, c))$. We describe $\mathcal{E}_S = (\mathsf{KeyGen}_S, \mathsf{Enc}_S, \mathsf{Eval}_S, \mathsf{Dec}_S)$ together with Expand and $\widetilde{\mathsf{Enc}}$, an expanded

encryption under a random public key. Note that [1] is an encryption of 1 with no randomness.

$\mathsf{KeyGen}_S(1^\lambda, 1^l)$:

1. Let $d = p(\lambda, l)$.
2. Let $(pk_\mathcal{F}, sk_\mathcal{F}) \leftarrow \mathsf{KeyGen}_\mathcal{F}(1^\lambda, 1^d)$.
3. Output $pk = (pk_\mathcal{F}, f_{sk} := \mathsf{Enc}_\mathcal{F}(pk_\mathcal{F}, sk_\mathcal{F}))$ and $sk = sk_\mathcal{F}$.

$\mathsf{Enc}_S(pk, \mu)$:

1. Parse $pk = (pk_\mathcal{F}, f_{sk})$
2. Let $c_\alpha \leftarrow \mathsf{Enc}_\mathcal{F}(pk_\mathcal{F}, \mu)$
3. Output c

$\mathsf{Expand}(pk_1, \ldots, pk_N, i, c)$:

1. For $j = 1, \ldots, N$, parse $pk_j = (pk_{\mathcal{F},j}, f_{sk,j})$.
2. Let $c_\alpha = c$ and $c_\gamma = [1] - c$
3. Compute $\tilde{c}_\alpha = \mathsf{Expand}_\mathcal{F}(pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, i, c_\alpha)$ and $\tilde{c}_\gamma = \mathsf{Expand}_\mathcal{F}(pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, i, c_\gamma)$
4. Output $\tilde{c} = (\tilde{c}_\alpha, \tilde{c}_\gamma)$.

$\widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, \mu)$:

1. Let $i \leftarrow [N]$ and compute $c \leftarrow \mathsf{Enc}(pk_i, \mu)$.
2. Output $\tilde{c} = \mathsf{Expand}(pk_1, \ldots, pk_N, i, c)$.

$\mathsf{Eval}_S(P, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$

1. Let $P = (G = (V, E), v_0, T, \psi_V, \psi_E)$.
2. For $j = 1, \ldots, N$, parse $pk_j = (pk_{\mathcal{F},j}, f_{sk,j})$.
   Let $\tilde{f}_{sk,j} = \mathsf{Expand}_\mathcal{F}(pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, j, f_{sk,j})$
3. For $i = 1, \ldots, n$, Let $(\tilde{c}_{\alpha,i}, \tilde{c}_{\gamma,i}) = \mathsf{Expand}(pk_1, \ldots, pk_N, i, c_i)$.
4. For each $v \in T$, let $label(v) := \psi_V(v)$.
5. For each $v \in V \setminus T$ with both children labeled, let $h := height(v)$, $i := \psi_V(v)$
   (a) For $t = 1, \ldots, s = |label(u_0)|$ where $\Gamma(v) = \{u_0, u_1\}$, $\psi_E(v, u_0) = 0$, $\psi_E(v, u_1) = 1$
      i. Let $r_0 = label(u_0)[t]$ and $r_1 = label(u_1)[t]$.
      ii. Let $z_1, z_2 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, 0)$
      iii. Consider 4 cases:
         A. if $r_0 = r_1 = 0$, $a_t := z_1 + z_2$
         B. if $r_0 = 0; r_1 = 1$, $a_t := \tilde{c}_{\alpha,i} + z_1$
         C. if $r_0 = 1; r_1 = 0$, $a_t := \tilde{c}_{\gamma,i} + z_1$
         D. if $r_0 = r_1 = 1$, $a_t := \tilde{c}_{\alpha,i} + \tilde{c}_{\gamma,i}$
   (b) $a_v = a_1 \ldots a_s$; if $h = 1$, $label(v) \leftarrow a_v$
   (c) otherwise, $label(v) \leftarrow \mathsf{Eval}_\mathcal{F}(\mathsf{Dec}^2_\mathcal{F}, pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, \tilde{f}_{sk,1}, \ldots, \tilde{f}_{sk,N}, a_v)$
6. Output $\tilde{c} = label(root)$

$\mathsf{Dec}_S(sk_1, \ldots, sk_N, \hat{c})$

1. Parse $sk_i = sk_{\mathcal{F},i}$.
2. Output $\mu' := \mathsf{Dec}_\mathcal{F}(sk_{\mathcal{F},1}, \ldots, sk_{\mathcal{F},N}, \hat{c})$

4.2 Correctness and Security Against Semi-honest Adversaries

The correctness is a direct result of the following lemma:

Lemma 5. *Let* $x = x_1 \dots x_n$. *For* $i = 1, \dots, N$, $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^l)$. *For* $i = 1, \dots, n$, $c_i = \mathsf{Enc}(pk_{I_i}, x_i)$ *for some* $I_i \in [N]$. *Then for any branching program* $P = (G = (V, E), v_0, T, \psi_V, \psi_E)$ *and for each* $v \in V \setminus T$ *with* $i = \psi_V(v)$,

1. $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, a_v) = label(u_{x_i})$;
2. $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, label(v)) = P_v(x)$;
3. $\mathsf{Dec}_S(sk_1, \dots, sk_N, \hat{c}) = P(x)$.

Proof. Let $\Gamma(v) = \{u_0, u_1\}$. For each $t \in [s]$, consider the value $\mu = x_i$ that $\tilde{c}_{\alpha,i}$ encrypts. If $\mu = 0$, we get a sum of two encryptions of 0 in the first two cases, and a sum of an encryption of 1 and an encryption of 0 in the last two cases. If $\mu = 1$, we get a sum of two encryptions of 0 in the first case and third case, and a sum of an encryption of 1 and an encryption of 0 in the second case and the last case. All of which are correct with respect to r_0, r_1. Thus, $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, a_v) = label(u_{x_i})$.

For v with $height(v) = 1$, we have $label(v) = a_v$. Thus, $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, label(v)) = label(u_{x_i}) = P_v(x)$ as $u_{x_i} \in T$. Now assume that $height(v) > 1$. Since $label(v) \leftarrow \mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, \tilde{f}_{sk,1}, \dots, \tilde{f}_{sk,N}, a_v)$, inductively, by part 1, we have $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, label(v)) = \mathsf{Dec}^2_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, a_v) = \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, label(u_{x_i})) = P_v(x)$.

Applying part 2 to the case $v = v_0$, we get

$$\mathsf{Dec}_S(sk_1, \dots, sk_N, \hat{c}) = \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \dots, sk_{\mathcal{F},N}, label(v_0)) = P_{v_0}(x) = P(x).$$

□

Now we prove circuit privacy against semi-honest adversaries, i.e., when each public key and ciphertext pair is generated correctly.

Lemma 6. *Assuming* $\mathcal{F}$ *is privately expandable HE scheme with circular security. Then the scheme* $\mathcal{E}_S$ *is a semi-honestly circuit-private HE scheme for branching programs.*

Proof. We construct a simulator Sim_S as follows:

$\mathsf{Sim}_S(1^\lambda, 1^l, (pk_1, \dots, pk_N), (I_1, c_1), \dots, (I_n, c_n), b)$:

1. For $i = 1, \dots, N$, parse $pk_i = (pk_{\mathcal{F},i}, f_{sk,i})$.
2. Let $out_0 = b$.
3. For $h = 1, \dots, l$,
   (a) For $t = 1, \dots, s = |out_{h-1}|$, we construct $out_h[t]$ as follows:
      i. Let $y_0, y_2 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \dots, pk_N, 0)$ and $y_1 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \dots, pk_N, 1)$.
      ii. Consider 2 cases:
         A. If $out_{h-1}[t] = 0$, $out_h[t] := y_0 + y_2$.
         B. If $out_{h-1}[t] = 1$, $out_h[t] := y_1 + y_2$.

(b) If $h \geq 2$, replace out_h with $\mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, \tilde{f}_{sk,1}, \ldots, \tilde{f}_{sk,N}, out_h)$

4. Output $out = out_l$

Let $P = (G = (V, E), v_r, T, \psi_V, \psi_E)$. For $h = 1, \ldots, l$, let $v^h \in V$ be the vertex at height h along the path indicated by x. We have $b = U(P, x_1^*, \ldots, x_n^*) = \psi_V(v^0)$ and $v^l = v_0$. The result follows from the following claim when $h = l$:

Claim. For $h = 0, \ldots, l$, $out_h \simeq^s label(v^h)$.

Proof. Clearly, $out_0 = label(v^0) = U(P, x_1, \ldots, x_n) = b$. Suppose $out_{h-1} = label(v^{h-1})$. Let $i = \psi_V(v^h)$. For each bit $b = out_{h-1}[t]$, if $b = 0$, we have $out_h[t] = y_0 + y_2$ and

$$a_t = \begin{cases} z_1 + z_2 \text{ or } \tilde{c}_{\alpha,i} + z_1 & \text{if } x_i = \psi_E(v^h, v^{h-1}) = 0; \\ \tilde{c}_{\gamma,i} + z_1 & \text{if } x_i = \psi_E(v^h, v^{h-1}) = 1 \end{cases}$$

Clearly, z_1 and y_0 have the same distribution as both are $\widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, 0)$. By private expandability, $\tilde{c}_{\alpha,i}$, $\tilde{c}_{\gamma,i}$ are statistically indistinguishable from y_2 when $x_i = \psi_E(v^h, v^{h-1}) = 0$ and $x_i = \psi_E(v^h, v^{h-1}) = 1$, respectively. We have $a_t \simeq^s out_h[t]$. Similarly, if $b = 1$, we have $out_h[t] = y_1 + y_2$ and

$$a_t = \begin{cases} \tilde{c}_{\gamma,i} + z_1 & \text{if } x_i = \psi_E(v^h, v^{h-1}) = 0; \\ \tilde{c}_{\alpha,i} + z_1 \text{ or } \tilde{c}_{\alpha,i} + \tilde{c}_{\gamma,i} & \text{if } x_i = \psi_E(v^h, v^{h-1}) = 1 \end{cases}$$

By private expandability, $\tilde{c}_{\gamma,i}$, $\tilde{c}_{\alpha,i}$ are statistically indistinguishable from y_1 when $x_i = \psi_E(v^h, v^{h-1}) = 0$ and $x_i = \psi_E(v^h, v^{h-1}) = 1$, respectively, while $\tilde{c}_{\gamma,i}$ is statistically indistinguishable from y_2 and z_1 when $x_i = \psi_E(v^h, v^{h-1}) = 1$. Again, we have $a_t \simeq^s out_h[t]$. Now average over the choice of $out_{h-1} \simeq^s label(v_{h-1})$, we have $a_t \simeq^s out_h$, and the result follows by applying $\mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, \tilde{f}_{sk,1}, \ldots, \tilde{f}_{sk,N}, \cdot)$ to both. □

We have $\mathsf{Sim}_S((pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n), b) \simeq^s \mathsf{Eval}_S(P, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$. □

4.3 Handling Malicious Inputs

Once we have an evaluation algorithm that can hide a branching program when public keys and ciphertexts are well-formed, we then consider the case when they are not properly generated. We use a single-key FHE with circuit privacy in Theorem 5 (such as one constructed in [21]) to homomorphically check the validity of each multi-key public key and ciphertext pair. If the check fails, we "mask" the output using a random string. The simulator can be constructed using the extractor guaranteed by the circuit privacy of single-key FHE to extract random coins and verify directly. If the check fails, it returns a random string with the same distribution as the masked output.

Let $\mathcal{P}$ be a circuit-private single-key FHE. We a define a circuit verifying each public key and corresponding ciphertexts:

$$\mathsf{Validate}_{\lambda,d,n}(pk, sk, r_k, (c_1, r_1), \ldots, (c_n, r_n), out) = \begin{cases} out & \text{if } (pk, sk) \leftarrow \mathsf{KeyGen}_{\mathcal{F}}(r_k) \\ & \text{and for each } i \in [n], \\ & c_i = \mathsf{Enc}_{\mathcal{F}}(pk, \mu_i; r_i) \\ & \text{for some } \mu_i \in \{0,1\}; \\ 0 & \text{otherwise} \end{cases}$$

We add a random string $S \in \{0,1\}^s$, where $s = s(\lambda, d) = |label(root)|$, to the output of Eval and return an encryption of S only if the verification passes. The original output can be computed if S can be recovered; otherwise, it is uniformly distributed. We define

$$v_j = \mathsf{Eval}_{\mathcal{P}}(\mathsf{Validate}(pk_j, \cdot, \cdot, \{(c_i, \cdot)\}_{I_i=j}, S_j), pk_{\mathcal{P},j}, p_{sk,j}, p_{kr,j}, \{p_{re,i}\}_{I_i=j})$$

where $p_{kr,j} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P},j}, r_{k,j})$, $p_{sk,j} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P},j}, sk_j)$ and $p_{re,i} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P},i}, r_{e,i})$, all of which are included in the new public key pk or the new ciphertext c. We also include $sk_{\mathcal{P}}$ in the new secret key sk. Finally, the new Eval returns $(label(root) \oplus (S_1 \oplus \ldots \oplus S_N), v_1, \ldots, v_N)$.

We describe $\mathcal{E}_M = (\mathsf{KeyGen}_M, \mathsf{Enc}_M, \mathsf{Eval}_M, \mathsf{Dec}_M)$ using the above Expand and $\widetilde{\mathsf{Enc}}$.

$\mathsf{KeyGen}_M(1^\lambda, 1^l)$:

1. Let $d = p(\lambda, l)$.
2. Let $(pk_{\mathcal{F}}, sk_{\mathcal{F}}) \leftarrow \mathsf{KeyGen}_{\mathcal{F}}(1^\lambda, 1^d; r_k)$.
3. Let $(pk_{\mathcal{P}}, sk_{\mathcal{P}}) \leftarrow \mathsf{KeyGen}_{\mathcal{P}}(1^\lambda)$.
4. Compute $f_{sk} := \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, sk_{\mathcal{F}}; r_e)$, $p_{kr} := \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, r_k)$ and $p_{sk} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, sk_{\mathcal{F}})$.
5. Output $pk = (pk_{\mathcal{F}}, f_{sk}, p_{kr}, p_{sk})$ and $sk = (sk_{\mathcal{F}}, sk_{\mathcal{P}})$.

$\mathsf{Enc}_M(pk, \mu)$:

1. Parse $pk = (pk_{\mathcal{F}}, f_{sk}, pk_{\mathcal{P}}, p_{kr}, p_{sk})$.
2. Let $c_{\mathcal{F}} \leftarrow \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, \mu; r_e)$
3. Compute $p_{re} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, r_e)$
4. Output $c = (c_{\mathcal{F}}, p_{re})$.

$\mathsf{Eval}_M(P, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$

1. Let $P = (G = (V, E), v_0, T, \psi_V, \psi_E)$.
2. For $j = 1, \ldots, N$,
   (a) Parse $pk_j = (pk_{\mathcal{F},j}, f_{sk,j}, pk_{\mathcal{P},j}, p_{kr,j}, p_{sk,j})$.
   (b) Let $S_j \leftarrow \{0,1\}^s$ and $v_j = \mathsf{Eval}_{\mathcal{P}}(\mathsf{Validate}(pk_j, \cdot, \cdot, \{(c_i, \cdot)\}_{I_i=j}, S_j), pk_{\mathcal{P},j}, p_{sk,j}, p_{kr,j}, \{p_{re,i}\}_{I_i=j})$.
   (c) Let $\tilde{f}_{sk,j} = \mathsf{Expand}_{\mathcal{F}}(pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, j, f_{sk,j})$
3. For $i = 1, \ldots, n$,

(a) Parse $c_i = (c_{\mathcal{F},i}, p_{re,i})$.
(b) Let $(\tilde{c}_{\alpha,i}, \tilde{c}_{\gamma,i}) = \mathsf{Expand}(pk_1, \ldots, pk_N, i, c_{\mathcal{F},i})$.
4. For each $v \in T$, let $label(v) := \psi_V(v)$.
5. For each $v \in V \setminus T$ with both children labeled, let $h := height(v)$, $i := \psi_V(v)$
(a) For $t = 1, \ldots, s = |label(u_0)|$ where $\Gamma(v) = \{u_0, u_1\}$, $\psi_E(v, u_0) = 0$, $\psi_E(v, u_1) = 1$
i. Let $r_0 = \widetilde{label(u_0)}[t]$ and $r_1 = label(u_1)[t]$.
ii. Let $z_1, z_2 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, 0)$
iii. Consider 4 cases:
A. if $r_0 = r_1 = 0$, $a_t := z_1 + z_2$
B. if $r_0 = 0; r_1 = 1$, $a_t := \tilde{c}_{\alpha,i} + z_1$
C. if $r_0 = 1; r_1 = 0$, $a_t := \tilde{c}_{\gamma,i} + z_1$
D. if $r_0 = r_1 = 1$, $a_t := \tilde{c}_{\alpha,i} + \tilde{c}_{\gamma,i}$
(b) $a_v = a_1 \ldots a_s$; if $h = 1$, $label(v) \leftarrow a_v$
(c) otherwise, $label(v) \leftarrow \widetilde{\mathsf{Eval}}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, \tilde{f}_{sk,1}, \ldots, \tilde{f}_{sk,N}, a_v)$
6. Output $\hat{c} = (label(root) \oplus (S_1 \oplus \ldots \oplus S_N), v_1, \ldots, v_N)$

$\mathsf{Dec}_M(sk_1, \ldots, sk_N, \hat{c})$

1. Parse $\hat{c} = (\tilde{c}, v_{k,1}, \ldots, v_{k,N})$.
2. For $j = 1, \ldots, N$,
(a) Parse $sk_j = (sk_{\mathcal{F},j}, sk_{\mathcal{P},j})$.
(b) Let $S_j = \mathsf{Dec}_{\mathcal{P}}(sk_{\mathcal{P},j}, v_{k,j})$.
3. Let $\tilde{c}' = \tilde{c} \oplus (S_1 \oplus \ldots \oplus S_N)$
4. Output $\mu' := \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \ldots, sk_{\mathcal{F},N}, \tilde{c}')$

4.4 Security Against Malicious Adversaries

We now prove that the above construction is secure against malicious adversaries as defined in Definition 8 by constructing a pair of algorithms Ext_M and Sim_M.

Theorem 6. *Assume $\mathcal{F}$ is a privately expandable multi-key HE scheme with circular security and $\mathcal{P}$ is maliciously circuit-private FHE. Then the above construction is a maliciously circuit-private HE scheme for the branching program.*

Proof. Let $\mathsf{Ext}_{\mathcal{P}}$ and $\mathsf{Sim}_{\mathcal{P}}$ be as defined in Definition 4. We construct Ext_M and Sim_M as follows:

$\mathsf{Ext}_M(1^\lambda, 1^l, pk^*, c^*)$:

1. Parse $pk^* = (pk^*_{\mathcal{F}}, f^*_{sk}, pk^*_{\mathcal{P}}, p^*_{kr}, p^*_{sk})$. If it is malformed, output 0.
2. Let $r^*_e = \mathsf{Ext}_{\mathcal{P}}(pk^*_{\mathcal{P}}, p^*_{re})$ and $sk^*_{\mathcal{F}} = \mathsf{Ext}_{\mathcal{P}}(pk^*_{\mathcal{P}}, p^*_{sk})$.
3. If $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) \neq \mathsf{KeyGen}_{\mathcal{F}}(1^\lambda, 1^d; r^*_e)$, return 0.
4. If $c^* = \mathsf{Enc}_{\mathcal{F}}(pk^*_{\mathcal{F}}, \mu; r^*_e)$ for some $\mu \in \{0, 1\}$, output μ.
5. Otherwise, output 0.

$\mathsf{Sim}_M(1^\lambda, 1^l, (pk_1^*, \ldots, pk_N^*), (I_1, c_1^*), \ldots, (I_n, c_n^*), b)$:

1. For $i = 1, \ldots, n$,
   (a) Parse $c_i^* = (c_{\mathcal{F},i}^*, p_{re,i}^*)$.
   (b) Let $\tilde{c}_i^* = \mathsf{Expand}(pk_1^*, \ldots, pk_N^*, i, c_i^*)$.
2. For $j = 1, \ldots, N$,
   (a) Parse $pk_j^* = (pk_{\mathcal{F},j}^*, f_{sk,j}^*, pk_{\mathcal{P},j}^*, p_{kr,j}^*, p_{sk,j}^*)$.
   (b) Do the same test as in Ext for pk_j^* and $\{c_i^*\}_{I_i=j}$. If any of the test fails, let $v_{k,j} = \mathsf{Sim}_{\mathcal{P}}(pk_{\mathcal{P},j}^*, p_{sk,j}^*, p_{kr,j}^*, \{p_{re,i}^*\}_{I_i=j}, 0)$.
   (c) Otherwise, let $S_j \leftarrow \{0,1\}^s$ and $v_j = \mathsf{Sim}_{\mathcal{P}}(pk_{\mathcal{P},j}^*, p_{sk,j}^*, p_{kr,j}^*, \{p_{re,i}^*\}_{I_i=j}, S_j)$.
   (d) Let $\tilde{f}_{sk,j}^* = \mathsf{Expand}_{\mathcal{F}}(pk_{\mathcal{F},1}^*, \ldots, pk_{\mathcal{F},N}^*, j, f_{sk,j}^*)$
3. If any of the tests above fail, let *out* be a random string of length s and skip to the last step.
4. Otherwise, let $out_0 = b$.
5. For $h = 1, \ldots, l$,
   (a) For $t = 1, \ldots, s = |out_{h-1}|$, we construct $out_h[t]$ as follows:
      i. Let $y_0, y_2 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, 0)$ and $y_1 \leftarrow \widetilde{\mathsf{Enc}}(pk_1, \ldots, pk_N, 1)$.
      ii. Consider 2 cases:
         A. If $out_{h-1}[t] = 0$, $out_h[t] := y_0 + y_2$.
         B. If $out_{h-1}[t] = 1$, $out_h[t] := y_1 + y_2$.
   (b) If $h \geq 2$, replace out_h with $\mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}_{\mathcal{F}}^2, pk_{\mathcal{F},1}^*, \ldots, pk_{\mathcal{F},N}^*, \tilde{f}_{sk,1}^*, \ldots, \tilde{f}_{sk,N}^*, out_h)$
6. Output $out = (out_l \oplus (S_1 \oplus \ldots \oplus S_N), v_{k,1}, \ldots, v_{k,N})$

We show that they satisfy the Definition 8.

Assume there exists $j \in [N]$ such that $\mathsf{Validate}(pk_{\mathcal{F},j}^*, sk_{\mathcal{F},j}^*, r_{k,j}^*, \{(c_i^*, r_{e,i}^*)\}_{I_i=j}, S_j) = 0$ for $sk_{\mathcal{F},j}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},j}^*, p_{sk,j}^*)$, $r_{k,j}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},j}^*, p_{kr,j}^*)$ and $r_{e,i}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},j}^*, p_{re,i}^*)$ for $I_i = j$. Then by circuit privacy of $\mathcal{P}$, v_i is statistically indistinguishable from $\mathsf{Sim}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},j}^*, p_{sk,j}^*, p_{kr,j}^*, \{p_{re,i}^*\}_{I_i=j}, 0)$ independent from S_j. Thus, *out* has the same distribution as a random string of length s in both Eval and Sim_M.

Now suppose that all $\mathsf{Validate}$'s are not zero, then $pk_{\mathcal{F},i}^*$ and $c_{\mathcal{F},i}^*$ are generated correctly. Since out_l is computed the same way as in Sim_M, the result follows from Lemma 6. □

Combining the above result with Lemma 4 results in the following theorem:

Theorem 7. *Let $\mathcal{F}$ be a privately expandable multi-hop multi-key compact somewhat homomorphic encryption scheme that can evaluate a circuit up to depth $2d + 2$ where d is the depth of $\mathsf{Dec}_{\mathcal{F}}$. Then the scheme described above is a maliciously circuit-private multi-key HE scheme for branching programs.*

Corollary 1. *Assuming RLWE and DSPR assumptions, and circular security for $\mathcal{E}_{SH}$, there exists a maliciously circuit-private multi-key HE scheme for branching programs.*

5 Circuit-Private Multi-key FHE

In this section we devise a framework turning a compact MFHE scheme and a circuit-private multi-key HE scheme into a circuit-private MFHE. This is a multi-key variant of the framework in [21]. As we discussed earlier, it is difficult to turn a single-key circuit-private HE scheme and a MFHE scheme into a circuit-private MFHE in the plain model. When both homomorphic encryption schemes are multi-key, each pair of public key and secret key can be generated together, thus allowing homomorphic decryption between two schemes. We use MFHE evaluation to evaluate a given circuit. We then switch to the circuit-private scheme to verify the input. Finally, we switch it back to the original scheme for compactness. Unlike the single-key case, we cannot verify all public keys and ciphertexts at once as it would lead to a larger verification circuit. We rely on the fully homomorphic property of the former to combine the result.

Let $\mathcal{F} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Enc}_{\mathcal{F}}, \mathsf{Eval}_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$ be a leveled compact multi-key FHE scheme and $\mathcal{P} = (\mathsf{KeyGen}_{\mathcal{P}}, \mathsf{Enc}_{\mathcal{P}}, \mathsf{Eval}_{\mathcal{P}}, \mathsf{Dec}_{\mathcal{P}})$ be a leveled multi-key circuit-private homomorphic scheme. Define the following programs:

$$\mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F}},out}(sk_{\mathcal{F}}, r_{\mathcal{F}K}) = \begin{cases} out \text{ if } (pk_{\mathcal{F}}, sk_{\mathcal{F}}) = \mathsf{KeyGen}_{\mathcal{F}}(1^{\lambda}, 1^{d}; r_{\mathcal{F}K}) \\ 0 \quad \text{otherwise.} \end{cases}$$

$$\mathsf{CValidate}^{\lambda,d}_{pk_{\mathcal{F}},c_{\mathcal{F}},out}(r_{\mathcal{F}E}) = \begin{cases} out \text{ if } c_{\mathcal{F}} = \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, b_i; r_{\mathcal{F}E}) \text{ for some } b_i \in \{0,1\} \\ 0 \quad \text{otherwise.} \end{cases}$$

$$\mathsf{CombineDec}(sk_{\mathcal{P},1}, \ldots, sk_{\mathcal{P},N}, c_1, \ldots, c_{N+n}) = \begin{cases} m \text{ if } \mathsf{Dec}_{\mathcal{P}}(sk_{\mathcal{P},1}, \ldots, sk_{\mathcal{P},N}, c_i) = m \\ \quad \text{for } \forall i = 1, \ldots, N+n \\ 0 \quad \text{otherwise.} \end{cases}$$

5.1 Construction

$\mathsf{KeyGen}(1^{\lambda}, 1^{d})$:

1. Let $(pk_{\mathcal{F}}, sk_{\mathcal{F}}) = \mathsf{KeyGen}_{\mathcal{F}}(1^{\lambda}, 1^{d}; r_{\mathcal{F}K})$ and $(pk_{\mathcal{P}}, sk_{\mathcal{P}}) \leftarrow \mathsf{KeyGen}_{\mathcal{P}}(1^{\lambda}, 1^{d_0})$ where d_0 is the maximum between the depth of $\mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F}},out}$, $\mathsf{CValidate}^{\lambda,d}_{pk_{\mathcal{F}},c_{\mathcal{F}},out}$ and $\mathsf{Dec}_{\mathcal{F}}$.
2. Let $p_{sk_{\mathcal{F}}} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, sk_{\mathcal{F}})$, $p_{r_{\mathcal{F}K}} = \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, r_{\mathcal{F}K})$ and $f_{sk_{\mathcal{P}}} = \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, sk_{\mathcal{P}})$.
3. Output $pk = (pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk_{\mathcal{F}}}, p_{r_{\mathcal{F}K}}, f_{sk_{\mathcal{P}}})$, $sk = sk_{\mathcal{F}}$.

$\mathsf{Enc}(pk, \mu)$:

1. Parse $pk = (pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk_{\mathcal{F}}}, p_{r_{\mathcal{F}K}}, f_{sk_{\mathcal{P}}})$.
2. Let $c_{\mathcal{F}} = \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, \mu; r_{\mathcal{F}E})$ and $p_{r_{\mathcal{F}E}} \leftarrow \mathsf{Enc}_{\mathcal{P}}(pk_{\mathcal{P}}, r_{\mathcal{F}E})$.
3. Output $c = (c_{\mathcal{F}}, p_{r_{\mathcal{F}E}})$.

$\mathsf{Eval}(C, (pk_1, \ldots, pk_N), (I_1, c_1), \ldots, (I_n, c_n))$

1. For $i = 1, \ldots, N$, parse $pk_i = (pk_{\mathcal{P},i}, pk_{\mathcal{F},i}, p_{sk_{\mathcal{F}},i}, p_{r_{\mathcal{F}K},i}, f_{sk_{\mathcal{P}},i})$.
2. For $i = 1, \ldots, n$, parse $c_i = (c_{\mathcal{F},i}, p_{r_{\mathcal{F}E},i})$.

3. If C is syntactically malformed, does not match n, or pk_i or c_i has incorrect size, replace C with a program returning 0.
4. Let $out_{\mathcal{F}} = \mathsf{Eval}_{\mathcal{F}}(C, (pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}), (I_1, c_{\mathcal{F},1}), \ldots, (I_n, c_{\mathcal{F},n}))$.
5. Let $out_{\mathcal{P}} = \mathsf{Eval}_{\mathcal{P}}(\mathsf{Dec}_{\mathcal{F}}(\cdot, out_{\mathcal{F}}), (pk_{\mathcal{P},1}, \ldots, pk_{\mathcal{P},N}), (1, p_{sk_{\mathcal{F}},1}), \ldots, (N, p_{sk_{\mathcal{F}},N}))$.
6. For $i = 1, \ldots, N$, let $out_{K,i} = \mathsf{Eval}_{\mathcal{P}}(\mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F},i}, out_{\mathcal{P}}}, (pk_{\mathcal{P},1}, \ldots, pk_{\mathcal{P},N}), (i, p_{sk_{\mathcal{F}},i}), (i, p_{r_{\mathcal{F}K},i}))$.
7. For $i = 1, \ldots, n$, let $out_{C,i} = \mathsf{Eval}_{\mathcal{P}}(\mathsf{CValidate}^{\lambda,d}_{pk_{\mathcal{F},i}, c_{\mathcal{F},i}, out_{\mathcal{P}}}, (pk_{\mathcal{P},1}, \ldots, pk_{\mathcal{P},N}), (i, p_{r_{\mathcal{F}E},i}))$.
8. Output $\hat{c} = \mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}_{\mathcal{P}}(\cdot, \mathsf{CombineDec}(\cdot, out_{K,1}, \ldots, out_{K,N}, out_{C,1}, \ldots, out_{C,n})), (pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}), (1, f_{sk_{\mathcal{P}},1}), \ldots, (N, f_{sk_{\mathcal{P}},N}))$.

$\mathsf{Dec}(sk_1, \ldots, sk_N, \hat{c})$

1. For $i = 1, \ldots, N$, parse $sk_i = sk_{\mathcal{F},i}$.
2. Output $y = \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \ldots, sk_{\mathcal{F},N}, \hat{c})$.

We now prove that this construction gives a leveled compact circuit-private MFHE.

Theorem 8. *Assume a compact leveled MFHE scheme $\mathcal{F}$ and a leveled $(U, \mathcal{C}_{\mathcal{F}})$-homomorphic circuit-private multi-key HE scheme $\mathcal{P}$ exist., where $\mathcal{C}_{\mathcal{F}}$ includes $\mathsf{Dec}_{\mathcal{F}}(\cdot, out_{\mathcal{F}})$, $\mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F}}, out_{\mathcal{P}}}$ and $\mathsf{CValidate}^{\lambda,d}_{pk_{\mathcal{F}}, c_{\mathcal{F}}, out_{\mathcal{P}}}$ for all $\lambda, d, pk_{\mathcal{F}}, c_{\mathcal{F}}$, $out_{\mathcal{P}}, out_{\mathcal{F}}$. The resulting scheme in the above construction is a leveled compact circuit-private MFHE.*

We refer to the full version of this paper for the proof.

5.2 Instantiation

Finally, if we instantiate the result of Theorem 8 by our construction in Theorem 7, we get the following results:

Corollary 2. *Assume there exists a privately expandable multi-hop multi-key compact somewhat homomorphic encryption scheme that can evaluate circuits up to depth $2d+2$ where d is the depth of its decryption circuit. Then there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

Corollary 3. *Assuming RLWE and DSPR assumptions, and circular security for $\mathcal{E}_{SH}$, there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

6 Three-Round On-the-Fly MPC with Circuit Privacy

In this section we consider one application of the circuit-private MFHE scheme—on-the-fly MPC protocol. In this setting, a large number of clients P_i uploaded

their encrypted inputs to a server or a cloud, denoted by S. The server selects an N-input function F on a subset of clients' input, and performs the computation without further information. Afterward, the server and the clients whose inputs are chosen run the rest of the protocol. At the end of an on-the-fly MPC protocol, only those clients learn the output while the server and other parties learn nothing. Furthermore, the communication complexity and the running time of clients should be independent of the function F. As in standard MPC, the input of each client should not be revealed to any other parties, including the server. In addition, we require circuit privacy for the server. Clients should not learn anything about the function other than its output. We give the formal definition of on-the-fly MPC protocol from [20] as follows:

Definition 9. *Let $\mathcal{C}$ be a class of functions with at most U inputs. An* on-the-fly multi-party computation protocol *Π for $\mathcal{C}$ is a protocol between $P_1, \ldots, P_U, S$ where P_i is given x_i as input, for $i \in [U]$, and S is given an ordered subset $V \subseteq [U]$ of size N and a function F on N inputs. At the end of the protocol, each party P_i for $i \in V$ outputs $F(\{x_i\}_{i \in V})$ while P_i for $i \notin V$ and S output $\perp$. The protocol consists of two phases:*

- Offline phase *is performed before F, V is chosen. All parties participate in this phase.*
- Online phase *starts after F, V is chosen. Only S and P_i for $i \in V$ participate in this phase, and ignore all messages from P_i, $i \notin V$.*

We require that the communication complexity of the protocol and the computation time of $P_1, \ldots, P_U$ be independent of (the complexity of) the function F. Furthermore, the computation time of P_i for $i \notin V$ is independent of the output size of F.

We then define the security and circuit privacy of on-the-fly MPC protocol in the plain model against malicious adversaries.

Definition 10. *An adversary $\mathcal{A}$ corrupting a party receives all messages directed to the corrupted party and controls the messages that it sends. Since the server ignores messages from parties outside V, we assume w.l.o.g. that an adversary only corrupts computing parties, i.e., parties in V.*

Let $\mathsf{View}_{\Pi,S}(F, V, \boldsymbol{x})$ denote the collection of messages the server S receives in an execution of protocol Π on a subset $V \subseteq [U]$ with $|V| = N$, an N-input function $F \in \mathcal{C}$ and input vector $\boldsymbol{x}$. Let $\mathsf{View}_{\Pi,\mathcal{A}}(F, V, \boldsymbol{x})$ denote the joint collection of messages $\mathcal{A}$ receives through corrupted parties in an execution of protocol Π on V, F and $\boldsymbol{x}$.

An on-the-fly multi-party computation protocol Π for $\mathcal{C}$ is secure if

- *for every adversary $\mathcal{A}$ corrupting parties $\{P_i\}_{i \in T}$ with $|T| = t < N$, for all $V \subseteq [U]$ with $|V| = N$, for all N-input function $F \in \mathcal{C}$ and for all input vectors $\boldsymbol{x}, \boldsymbol{x}'$ such that $x_i = x'_i$ for any $i \in T$,*

$$[\mathsf{View}_{\Pi,\mathcal{A}}(F, V, \boldsymbol{x}) | y = F(\{x_i\}_{i \in V})] \simeq^c [\mathsf{View}_{\Pi,\mathcal{A}}(F, V, \boldsymbol{x}') | y = F(\{x'_i\}_{i \in V})] .$$

– *for every server* S*, for all* $V \subseteq [U]$ *with* $|V| = N$*, for all* N*-input function* $F \in \mathcal{C}$ *and for all input vectors* $\boldsymbol{x}, \boldsymbol{x'}$*,*

$$[\mathsf{View}_{\Pi,S}(F, V, \boldsymbol{x})|y = F(\{x_i\}_{i\in V})] \simeq^c [\mathsf{View}_{\Pi,S}(F, V, \boldsymbol{x'})|y = F(\{x'_i\}_{i\in V})]\,.$$

Let the ideal world protocol be where the computation of F *is performed through a trusted functionality* $\mathcal{F}$*. Each party* P_i *sends their input* x_i *to* $\mathcal{F}$*, the server sends* F *and* V *to* $\mathcal{F}$*, which performs the computation and sends the output* $F(\{x_i\}_{i\in V})$ *to each* P_i*,* $i \in V$*. Let* $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}}(F, V, x)$ *denote the joint output of the ideal-world adversary* $\mathcal{S}$*, parties* $P_1, \ldots, P_U$ *and the server* S*. Let* $\mathsf{REAL}_{\Pi,\mathcal{A}}(F, V, x)$ *denote the joint output of the real-world adversary* $\mathcal{S}$*, parties* $P_1, \ldots, P_U$ *and the server* S*.*

The protocol Π *has* (malicious) circuit privacy *if for every malicious (and possibly unbounded) adversary* $\mathcal{A}$ *corrupting any number of clients, there exists an unbounded simulator* $\mathcal{S}$ *with black-box access to* $\mathcal{A}$ *such that for all* $V \subseteq [U]$ *with* $|V| = N$*, for all* N*-input function* $F \in \mathcal{C}$ *and for all input vectors* $\boldsymbol{x}$*,* $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}}(F, V, x) \simeq^s \mathsf{REAL}_{\Pi,\mathcal{A}}(F, V, x)$*.*

Adding circuit privacy to an on-the-fly MPC protocol via circuit-private MFHE scheme has two implications beyond the definition state above. First, it automatically strengthen the protocol against malicious adversaries without using setup. This is because the evaluated output only depends on the output and encrypted input even against malformed public keys and ciphertexts. On the other hand, it implies that the clients do not know the function being evaluated, which in turn makes it difficult, if even possible, to verify against a malicious server. Therefore, we assume that the server is only honest-but-curious, that it follows the protocol, but may try to learn clients' input data.

Naturally, the MFHE scheme leads to server-assisted MPC by having each client generate keys, and encrypt its inputs and uploads to the server. The server then runs an evaluation algorithm on the encrypted inputs. However, in order to decrypt the evaluated output, one needs to have all secret keys. One solution, as in [17], is to run another MPC protocol with each client's secret key as input to decrypt. However, this results in multiple rounds in the plain model. In order to solve this problem, we use a projective garbling scheme.

After the server runs the evaluation algorithm, it creates a garbled circuit of MFHE decryption with secret keys as input. In order to create a garbled input, the server cannot give e to the clients as it will allow the clients to generate multiple garbled inputs, thus rendering the security meaningless. We solve this problem by using a 1-out-of-2 oblivious transfer (OT). In order to minimize the round complexity of our MPC protocol, we consider an OT protocol that runs in one round. However, the standard one-round 1-out-of-2 OT protocols known are only secure against semi-honest receivers.

We refer to the full version of this paper for the formal definitions of the garbling scheme and OT, and the construction of a one-round 1-out-of-2 OT protocol that is secure against malicious receivers from maliciously circuit-private single-key FHE.

Theorem 9. *Assuming a circuit-private single-key FHE, there exists a one-round 1-out-of-2 oblivious transfer protocol that is secure against malicious receivers.*

6.1 Construction

Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ be a (leveled) compact maliciously circuit-private MFHE scheme with secret key length $s = s(\lambda)$ and using $r = r(\lambda)$ random bits for key generation. For simplicity, we assume that each client's input is 1 bit. The protocol can be easily generalized to the case where each client holds many bits of input. Compactness of the MFHE implies that the evaluated output do not depend on the size of the input. Thus, the rest of our protocol stays the same. Let $(G_{\mathsf{OT}}, Q_{\mathsf{OT}}, A_{\mathsf{OT}}, D_{\mathsf{OT}})$ be a one-round 1-out-of-2 OT protocol. Let $(\mathsf{GarbCircuit}, \mathsf{GarbEval})$ be a projective gabling scheme. Let U be the set of indices of all clients in the system. We describe an on-the-fly MPC protocol $\Pi_N(V, F, x)$ as follows:

On-the-Fly MPC Protocol

Step 1: For $i \in [U]$, client P_i generates a key pair $(pk_i, sk_i) = \mathsf{KeyGen}(1^\lambda; r_i)$ and encrypts his input $c_i \leftarrow \mathsf{Enc}(pk_i, x_i)$. For each $j = 0, \ldots, s + r - 1$, P_i also generates $(pk^j_{OT,i}, sk^j_{\mathsf{OT},i}) \leftarrow G_{\mathsf{OT}}(1^\lambda)$. It computes bitwise $q^j_i = Q_{\mathsf{OT}}(pk^j_{\mathsf{OT},i}, sk_i[j])$ for $j = 0, \ldots, s-1$, and $q^{s+j}_i = Q_{\mathsf{OT}}(pk^j_{\mathsf{OT},i}, r_i[j])$ for $j = 0, \ldots, r-1$. It then sends $(pk_i, c_i, pk_{\mathsf{OT},i}, \overrightarrow{q}_i)$ to the server S.

The server S then selects a circuit C representing the function F on inputs $\{x_i\}_{i \in V}$ for a subset $V \subseteq U$ such that $|V| = N$. We may assume w.l.o.g. that $V = [N]$.

Step 2: The server S computes $c = \mathsf{Eval}(C, pk_1, \ldots, pk_N, c_1, \ldots, c_N)$. S computes a garbled circuit $(G, e) = \mathsf{GarbCircuit}(1^\lambda, g_{c,pk_1,\ldots,pk_N})$ where

$$g_{c,pk_1,\ldots,pk_N}((sk_1, r_1), \ldots, (sk_N, r_N)) = \begin{cases} \mathsf{Dec}(sk_1, \ldots, sk_N, c) & \text{if } (pk_i, sk_i) = \\ & \mathsf{KeyGen}(1^\lambda; r_i) \\ & \text{for all } i \in [N]; \\ \perp & \text{otherwise} \end{cases}$$

and $e = (X^0_0, X^1_0, \ldots, X^0_{N(r+s)-1}, X^1_{N(r+s)-1})$. For each $i \in [N]$ and $j = 0, \ldots, r+s-1$, it computes $a^j_i = A_{\mathsf{OT}}(pk_{\mathsf{OT},i}, q^j_i, X^0_{i(r+s)+j}, X^1_{i(r+s)+j})$. It sends $(G, a^0_i, \ldots, a^{r+s-1}_i)$ (and V) to P_i for each $i \in V$.

Step 3: For $i \in V$, client P_i computes its garbled input $X_{i(r+s)+j} = D_{\mathsf{OT}}(sk_{OT,i}, a^j_i)$ for $j = 0, \ldots, r+s-1$ and broadcasts to other $P_{i'} \in V$. Each client computes $y = \mathsf{GarbEval}(G, X_0, \ldots, X_{N(r+s)-1})$.

Remarks

1. The upper bound on the number of clients whose inputs are used in a computation must be known in advance. This requirement is inherited from the multi-key homomorphic encryption scheme in [17] that we use to construct our MFHE. It is also the case for the on-the-fly MPC construction in [17].
2. Private channel (from the server) between clients is required to prevent the server learning clients' secret keys. This requirement can be done by the honest-but-curious server passing public keys of all parties in V along with its messages in step 2. The public key of P_i can be used to encrypt a garbled input from P_j to P_i.
3. We require circular security between MFHE and OT schemes. This can be done without additional assumptions by using OT constructed from the same circuit-private homomorphic scheme in Sect. 4.

Theorem 10. *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *be a leveled compact MFHE scheme. Let* $\mathsf{OT} = (G_{\mathsf{OT}}, Q_{\mathsf{OT}}, A_{\mathsf{OT}}, D_{\mathsf{OT}})$ *be an OT protocol. Let* $Gb = (GarbCir\text{-}\ cuit, GarbEval)$ *be a projective garbling scheme. If* $\mathcal{E}$ *is maliciously circuit-private,* OT *is secure against malicious receivers, and* Gb *is a secure garbling scheme, then the protocol* Π_N *is a 3-round secure on-the-fly MPC protocol with circuit privacy.*

We refer to the full version of this paper for the proof.

7 Conclusion and Open Questions

We have shown that we can construct circuit-private MFHE from the existing multi-key HE and single-key circuit-private FHE. We also use it to construct an on-the-fly MPC with circuit privacy against malicious clients in the plain model. However, our construction inherits the same assumption as the construction of MFHE of López-Alt et al., including DSPR and RLWE. So, the main open question is:

Is it possible to construct a multi-key homomorphic encryption (with circuit privacy) under standard assumptions such as LWE in the plain model?

Since our technique only relies on properties that exist in many single-key constructions, we expect that we can apply it to other multi-key HE as well. Moreover, circuit privacy for on-the-fly MPC requires some degree of trust toward a server party. Our construction assumes the server to be honest-but-curious. We would like to capture a wider range of unintended behavior of the server while still achieving circuit privacy. So, another open question is:

Is there a better model for on-the-fly MPC with circuit privacy?

References

1. Aiello, B., Ishai, Y., Reingold, O.: Priced oblivious transfer: how to sell digital goods. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 119–135. Springer, Heidelberg (2001). doi:10.1007/3-540-44987-6_8
2. Albrecht, M., Bai, S., Ducas, L.: A subfield lattice attack on overstretched NTRU assumptions: cryptanalysis of some FHE and graded encoding schemes. Technical report, Cryptology ePrint Archive, Report 2016/127 (2016)
3. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29011-4_29
4. 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). doi:10.1007/978-3-642-22792-9_29
5. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1), 143–202 (2000)
6. Chongchitmate, W., Ostrovsky, R.: Circuit-private multi-key FHE. Cryptology ePrint Archive, Report 2017/010 (2017). http://eprint.iacr.org/
7. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48000-7_31
8. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Kim, K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, London (2001). doi:10.1007/3-540-44586-2_9. http://dl.acm.org/citation.cfm?id=648118.746742
9. Ducas, L., Stehlé, D.: Sanitization of FHE ciphertexts. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 294–310. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49890-3_12
10. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009)
11. 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). doi:10.1007/978-3-642-14623-7_9
12. 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). doi:10.1007/978-3-642-40041-4_5
13. Halevi, S., Kalai, Y.T.: Smooth projective hashing and two-message oblivious transfer. J. Cryptol. **25**(1), 158–193 (2012)
14. Halevi, S., Lindell, Y., Pinkas, B.: Secure computation on the web: computing without simultaneous interaction. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 132–150. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22792-9_8. http://dl.acm.org/citation.cfm?id=2033036.2033047
15. Ishai, Y., Paskin, A.: Evaluating branching programs on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 575–594. Springer, Heidelberg (2007). doi:10.1007/978-3-540-70936-7_31. http://dl.acm.org/citation.cfm?id=1760749.1760790

16. Kamara, S., Mohassel, P., Raykova, M.: Outsourcing multi-party computation. IACR Cryptology ePrint Archive 2011, 272 (2011)
17. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of 44th Annual ACM Symposium on Theory of Computing, STOC 2012, NY, USA, pp. 1219–1234 (2012). http://doi.acm.org/10.1145/2213977.2214086
18. Mohassel, P., Sadeghian, S.: How to hide circuits in MPC an efficient framework for private function evaluation. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 557–574. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38348-9_33
19. Mohassel, P., Sadeghian, S., Smart, N.P.: Actively secure private function evaluation. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 486–505. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45608-8_26
20. Mukherjee, P., Wichs, D.: Two round mutliparty computation via multi-key FHE. Cryptology ePrint Archive, Report 2015/345 (2015). http://eprint.iacr.org/
21. Ostrovsky, R., Paskin-Cherniavsky, A., Paskin-Cherniavsky, B.: Maliciously circuit-private FHE. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 536–553. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44371-2_30
22. Pippenger, N.: On simultaneous resource bounds. In: Proceedings of 20th Annual Symposium on Foundations of Computer Science, SFCS 1979, pp. 307–311 (1979). http://dx.doi.org/10.1109/SFCS.1979.29
23. Raz, R.: Elusive functions and lower bounds for arithmetic circuits. In: Proceedings of 40th Annual ACM Symposium on Theory of Computing, STOC 2008, NY, USA, pp. 711–720 (2008). http://doi.acm.org/10.1145/1374376.1374479
24. Valiant, L.G.: Universal circuits (preliminary report). In: Proceedings of 8th Annual ACM Symposium on Theory of Computing, STOC 1976, NY, USA, pp. 196–203 (1976). http://doi.acm.org/10.1145/800113.803649
25. Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010). doi:10.1007/978-3-642-13190-5_2