Title
New Models for Multi-party Computation

Author
Chongchitmate, Wutichai

Publication Date
2016

Peer reviewed|Thesis/dissertation

University of California
Los Angeles

New Models for Multi-party Computation

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Wutichai Chongchitmate

2016

Abstract of the Dissertation

New Models for Multi-party Computation

by

Wutichai Chongchitmate

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2016

Professor Rafail Ostrovsky, Chair

Secure multi-party computation (MPC) is one of the most important primitives in cryptography. Several directions to construct and improve MPC protocols have been studied over the past decades. The majority of existing MPC protocols rely on assumptions such as every party communicates with every other party or every party needs to know every other party and the computation algorithm well in advance. Such assumptions can be difficult to achieve in modern large scale networks. This thesis explores two distinct lines of work leading to new practical models in response to these scenarios.

First, we examine *communication locality*, the total number of other parties that each party communicates with in the protocol, as a new measure for an MPC protocol. Although progress has been made, the question of achieving low communication locality and round complexity at the same time for adaptive adversaries corrupting $t < n/2$ parties remains open. We show that by assuming a public-key infrastructure (PKI) and a symmetric-key infrastructure (SKI) the above question can be answered affirmatively, under standard assumptions. In particular, we describe a protocol with polylogarithmic communication locality and round complexity which tolerates up to $t < n/2$ adaptive corruptions. Our results are based on a new model of hidden random graphs obtained from the SKI, and using them to emulate a complete network in polylogarithmic rounds and polylogarithmic locality.

We then examine multi-key fully homomorphic encryption (MFHE) schemes, which allow computation on data encrypted under different public keys chosen independently of each other. One of the main problems left in MFHE has been on how to deal with malicious users without trusted setup assumptions. We show how this can be done through *circuit privacy*, which guarantees that even if both ciphertexts and public keys are not well-formed, no information is revealed regarding the computation, other than what follows from the output on some well-formed inputs. Generalizing the result for circuit-private FHE, we provide a framework for adding circuit privacy to existing MFHE schemes.

Finally, we incorporate the circuit privacy in a variant of server-assisted MPC, called *on-the-fly MPC*, where a server or a "cloud" performs an arbitrary dynamically chosen computation on a subset of data uploaded by multiple clients and encrypted under different keys *on-the-fly*. Circuit privacy allows the server to keep the algorithm used in the computation private even from a malicious adversary corrupting any number of clients. In particular, we construct a three-round on-the-fly MPC protocol with circuit privacy against malicious clients without the trusted setup.

The dissertation of Wutichai Chongchitmate is approved.

Eliezer M. Gafni

Igor Pak

Rafail Ostrovsky, Committee Chair

University of California, Los Angeles

2016

Table of Contents

List of Figures

Acknowledgments

First of all, I would like to thank my advisor Rafail Ostrovsky for teaching me various topics both in his cryptography class and throughout our weekly meetings. Most importantly, I would like to thank him for continuing patience and encouragement. He spent a lot of time with me on several projects, even though some of them do not give interesting result.

Additionally, I would like to thank my coauthors, Nishanth Chandran, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky, and Vassilis Zikas, for intellectual discussions. Special thanks goes to Nishanth Chandran for additional discussions through Skype calls from India. In particular, chapter three is a joint work with Nishanth Chandran, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky and Vassilis Zikas, entitled "The Hidden Graph Model: Communication Locality and Optimal Resiliency with Adaptive Faults," published in Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS '15). Chapter four is a joint work with Rafail Ostrovsky, entitled "Circuit-Private Multi-key FHE."

I would like to thank UCLA and Department of Mathematics, including Maggie Albert, Martha Contreras and Maida Bassili. I would like to thank Mark Green for introducing me to modern cryptography in his algebra class, and Benny Sudakov for teaching probabilistic combinatorics classes which provide important tools for the work in chapter three. I would like to thank Igor Pak, Richard Elman and Eliezer Gafni for being on my thesis committee.

Finally, I would like to thank my family in Thailand for emotional support and encouragement.

Vita

2007–2010	Grader, Mathematics Department, Duke University.
2008–2009	Undergraduate Teaching Assistant, Computer Science Department, Duke University.
2010	B.S. (Mathematics) and A.B. (Computer Science), Duke University.
2012	M.A. (Mathematics), UCLA, Los Angeles, California.
2010–present	Teaching Assistant, Mathematics Department, UCLA.

Publications

The Hidden Graph Model: Communication Locality and Optimal Resiliency with Adaptive Faults. Nishanth Chandran, *Wutichai Chongchitmate*, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky, and Vassilis Zikas. 2015. In Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS '15), ACM, pp. 153-162.

An Atlas of Legendrian Knots. Wutichai Chongchitmate, Lenhard Ng. 2013. Experimental Mathematics, 22(1), pp. 26-37.

Architectural Implications on Nanoscale Integrated Sensing and Computing. Constantin Pistol, *Wutichai Chongchitmate*, Christopher Dwyer and Alvin R. Lebeck. 2009. Architectural Support for Programming Languages and Operating Systems: Proceeding of the

14th International Conference on Architectural Support for Programming Languages and Operating Systems, ACM, pp. 13-24.

CHAPTER 1

Introduction

An improvement in online technology such as social media and web applications has escalated in the last decade. Now, our personal data lives online in many forms. Security and privacy have become more relevant to ordinary people than they were a few years ago. Many applications rely on external servers or "clouds" to store users' information and perform intensive computation such as analyzing or data mining. Users, or even the application providers, have little control over these external servers. At the same time, the technology industry has also made an improvement in scaling their applications to an exponentially larger scale. In order to maintain personal privacy over such an insecure network, we rely on cryptography.

Cryptography is the study of secure and reliable communication in the presence of adversaries. One of the most fundamental primitives in secure computing is secure multi-party computation (MPC). An MPC protocol allows multiple distrusted parties to evaluate a chosen function over their inputs without revealing each party's input to other parties. In order to compute a function, the parties will engage in a multiple-round protocol, where they communicate with other parties. At the end of the protocol, designated parties will obtain an output of the function evaluated with inputs from the parties involved. Each party should learn nothing about the other parties' inputs beyond what they can deduce from the output they receive. Such a security guarantee needs to hold even in the existence of "corrupted" parties, possibly colluding with each other, or controlled by a single adversary.

Many directions of MPC have been studied intensively since its introduction by Yao in 1982 including the two-parties case, honest-but-curious parties, static or adaptive adversaries

who corrupt multiple parties, and different classes of functions being evaluated. Several efficiency metrics have been considered and improved such as round complexity, the number of rounds the protocol needs to run before the goal is achieved, communication complexity, the number and length of messages sent by each party, and computation complexity, the amount of computation each party need to perform.

Several models of MPC have also been considered such as the setup model where all parties share common reference string (CRS), public-key infrastructure (PKI) or symmetric-key infrastructure (SKI). Another direction is server-assisted MPC, where most computation will be performed asymmetrically by a computationally powerful party, called "server" or "cloud." This allows computation of more complicated algorithms on the users' data without an increase in the work done by each user. It also coincides with the growing trend of commercial cloud services for online applications.

One of many directions to construct an MPC protocol is via another cryptographic primitive, called homomorphic encryption (HE). Homomorphic encryption schemes allow computation to be carried out on encrypted data. The result of such computation is still encrypted. Upon decryption, the output is as if one performs the same computation of plaintexts. Fully homomorphic encryption (FHE) allows any functions to be evaluated in this manner.

This thesis contains two distinct lines of work; one based on expansion properties of undirected random graphs, while another is based on homomorphic encryption. While each study starts off from a different approach, both result in new practical models for secure multi-party computation in a large network setting, such as the internet, each with additional properties that are relevant in real world secure computing.

In chapter three, we study expansion properties of Erdős-Rényi random graphs, in which each pair of vertices are connected independently with low probability. We then construct a communication model based on "hidden" random graphs, where any user can securely send an authenticated message to any other users under the presence of malicious adversaries who adaptively corrupt a constant fraction of users. In this model, each user only needs to communicate with a small number of other users over a small number of rounds.

This results in an improvement over a new efficiency metric for MPC protocols, *communication locality*, the number of other parties each party needs to communicate with. While most MPC protocols assume a complete secure network, where every pair of users can securely communicate with each other, such an assumption may not hold in a large network, where the number of users can be much larger than the complexity of function evaluated. We construct an MPC protocol where each user only communicates with polylogarithmic number of other users in each round over polylogarithmic number of rounds. This protocol is secure against an adaptive malicious adversary who adaptively chooses to corrupt less than half of the users.

In chapter four, we first consider two novel properties of homomorphic encryption: *multi-key*, where ciphertexts encrypted under different public keys can be securely evaluated, and *circuit privacy*, where the encrypted output reveals nothing about the computation even against unbound malicious adversaries who may provide malformed public keys and ciphertexts. Homomorphic encryption schemes with one of these properties have been recently constructed. So, the natural question to ask is "is it possible to have both properties in the same scheme?" We answer this question affirmatively by constructing the first homomorphic encryption scheme with both properties. We also give a framework adding *circuit privacy* to the existing multi-key FHE (MFHE) schemes without adding further assumption.

We then turn to an application of circuit-private MFHE schemes in the *on-the-fly* MPC setting, a variant of server-assisted MPC where the server can compute an arbitrary function on a chosen subset of users "on-the-fly." In this setting, a large number of users upload their encrypted data to the server without any prior knowledge of functions being computed nor existence of other users. Upon users or application providers' request, the server performs a computation on encrypted data while the users can be "offline". Only the users whose input are used in the computation need to engage in "online" decryption in order to obtain the output. We construct an on-the-fly MPC protocol with circuit privacy, where even unbounded malicious adversaries learn nothing about algorithms used to compute on encrypted data.

These new models for MPC protocols can be used in practical settings. The MPC with

low communication locality allows secure computation over a large scale network, where the total number of users is the main concern, which is the case for many global scale web application. The on-the-fly MPC not only allows a large number of users in the system (and computation complexity of each user is independent of such number), it also lets the users to stay offline while the server performs intensive computation. Circuit privacy allows service or application providers to keep their algorithms secret as intellectual properties or for security reasons.

CHAPTER 2

Preliminaries

In this chapter, we give basic notations and define basic cryptographic terminologies we will use throughout the rest of this thesis.

2.1 Basic Notations

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 $x \leftarrow A$ and $A \rightarrow x$ denote x is chosen according to a distribution A. For a finite set S, let $x \leftarrow S$ and $S \rightarrow x$ denote x is chosen uniformly from the set S. Let λ denote a security parameter. A function $f : \mathbb{N} \rightarrow \mathbb{R}^+$ is *negligible* if for every constant $c > 0$, there exists $\lambda_0 \in \mathbb{N}$ such that $f(\lambda) \leq \lambda^{-c}$ for all $\lambda \geq \lambda_0$. Let $\mathrm{negl}(\lambda)$ denote an unspecified negligible function. We say an event occurs *with overwhelming probability* if it occurs except with negligible probability. Algorithms may be randomized unless stated otherwise. A PPT algorithm runs in probabilistic polynomial-time; otherwise, it is unbounded. For an algorithm A, let $y \leftarrow A(x; r)$ and $A(x; r) \rightarrow y$ denote running A on input x with random coins r. If r is chosen uniformly at random, we denote $y \leftarrow A(x)$. Let $\log n$ denote logarithmic base 2. let $\mathrm{polylog}(n)$ denote polylogarithmic function in n, i.e. $p(\log n)$ for some polynomial p.

2.2 Definitions

Let X, Y be random variables over a discrete space S.

Definition 2.2.1. *The* statistical distance *between X and Y, denoted $\Delta(X, Y)$, is defined as*

$$\frac{1}{2}\sum_{s\in S} |\Pr[X = s] - \Pr[Y = s]|.$$

We say X and Y are statistically closed *or* statistically indistinguishable*, denoted $X \simeq^s Y$ if $\Delta(X, Y)$ is negligible.*

Definition 2.2.2. *We say X and Y are* computationally indistinguishable*, denoted $X \simeq^c Y$ if for any PPT algorithm D, $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]|$ is negligible.*

Definition 2.2.3. *A cryptosystem $(\mathsf{KeyGen}, \mathsf{Enc})$ over a plaintext space $\mathcal{M}$ is* semantically secure *if for any $m, m' \in \mathcal{M}$, for $pk \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\mathsf{Enc}(pk, m)$ and $\mathsf{Enc}(pk, m')$ are computationally indistinguishable.*

2.2.1 Cryptographic Primitives

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.

Homomorphic Encryption A homomorphic encryption (HE) scheme is a public-key cryptosystem that allows computation on encrypted data. We give the formal description as follows:

Definition 2.2.4. *Let $\mathcal{C}$ be a class of circuits. A* (leveled) *$(U, \mathcal{C})$*-homomorphic encryption *scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ 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*.*
- $\hat{c} \leftarrow \mathsf{Eval}(C, pk, c_1, \ldots, c_n)$*: Given a (description of) a boolean circuit* $C \in \mathcal{C}$ *(of depth* $\leq d$*), a public key* pk *and* n *ciphertexts* $c_1, \ldots, c_n$*, outputs an evaluated ciphertext* $\hat{c}$*.*
- $b := \mathsf{Dec}(sk, \hat{c})$*: Given a secret key* sk *and a ciphertext* $\hat{c}$*, outputs a bit* b*.*

has the following properties:

- ***Semantic security****:* $(\mathsf{KeyGen}, \mathsf{Enc})$ *is semantically secure.*
- ***Correctness****: Let* $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda, 1^d)$*. Let* $x = x_1 \ldots x_n \in \{0,1\}^n$ *and* $C \in \mathcal{C}$ *be a boolean circuit (of depth* $\leq d$*) representing a function* $f : \{0,1\}^n \to \{0,1\}$*. For* $i = 1, \ldots, n$*, let* $c_i \leftarrow \mathsf{Enc}(pk, x_i)$*. Let* $\hat{c} \leftarrow \mathsf{Eval}(C, pk, c_1, \ldots, c_n)$*. Then* $\mathsf{Dec}(sk, \hat{c}) = U(C, (x_1, \ldots, x_n))$*.*

$\mathcal{E}$ *is* compact *if there exists a polynomial* p *such that* $|\hat{c}| \leq p(\lambda, d)$ *independent of* $|C|$ *and* n*. If a scheme is* $(U, \mathcal{C})$*-homomorphic for the class* $\mathcal{C}$ *of all circuits (of depth* $\leq d$*), we call it a* (leveled) fully homomorphic (FHE)*.*

Signature Scheme A secure (digital) signature scheme allows a party to sign any given message with his signing key such that an adversary without signing key cannot forge the signature with non-negligible probability.

Definition 2.2.5. *A* digital signature *scheme* $\Sigma = (\mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver})$ *described as follows:*

- $(sk, vk) \leftarrow \mathsf{KeyGen}(1^\lambda)$*: Given a security parameter* λ*, outputs a signing key* sk *and a verification key* vk*.*
- $\sigma \leftarrow \mathsf{Sig}(sk, m)$*: Given a signing key* sk *and a message* m*, outputs a signature* σ*.*

- $\mathsf{Ver}(vk, m, \sigma) \in \{0,1\}$*: Given a verification key* vk*, a message* m *and a signature* σ*, outputs a bit.*

has the following properties:

- ***Correctness****:* $\Pr[(sk, vk) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{Ver}(vk, m, \mathsf{Sig}(sk, m)) = 1] = 1$ *for every message* m*.*
- ***Security****: For any PPT adversary* $\mathcal{A}$*, let* Q *be the set of messages* $\mathcal{A}$ *queries, then* $\Pr[(sk, vk) \leftarrow \mathsf{KeyGen}(1^\lambda), (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sig}(sk,\cdot)}(1^\lambda, vk), m \notin Q, \mathsf{Ver}(vk, m, \sigma) = 1] < \mathrm{negl}(\lambda)$.

Pseudorandom functions A pseudorandom function family is a collection of efficiently-computable functions such that a function drawn uniformly at random is computationally indistinguishable from a random oracle. Formally, for a polynomial p, a collection of polynomial-time function $\mathcal{F} = \{f_i : \{0,1\}^{p(\lambda)} \to \{0,1\}^{p(\lambda)} | i \in I\}$ is a *pseudorandom function family* if $f \leftarrow \mathcal{F}$ is computationally indistinguishable from a function drawn from a collection of all functions $\{f : \{0,1\}^{p(\lambda)} \to \{0,1\}^{p(\lambda)}\}$.

Trapdoor permutation A trapdoor permutation is a easy-to-compute bijective function that is hard to invert unless with a special information, called "trapdoor."

Definition 2.2.6. *A collection of* trapdoor permutations *is a collection of efficiently-computable bijective functions indexed by* I *such that each* $f_i : D_i \to D_i$ *can be efficiently sampled by* $\mathcal{G}$ *together with its* trapdoor t_i *with the following properties:*

- ***Easy to sample domain****: there exists PPT algorithm* χ *such that* $\chi(i)$ *uniformly sampling from* D_i
- ***Hard to invert****: for* $(i, t_i) \leftarrow \mathcal{G}(1^\lambda)$*,* $x \leftarrow \chi(i)$*, for any PPT adversary* $\mathcal{A}$*,* $\Pr[\mathcal{A}(i, f_i(x)) = x] < \mathrm{negl}(\lambda)$.

- ***Easy to invert with trapdoor****: there exists an polynomial-time algorithm T such that for $(i, t_i) \leftarrow \mathcal{G}(1^\lambda)$, $T(i, t_i, f_i(x)) = x$ for all $x \in D_i$.*

f has reverse domain sampler *if there exists PPT algorithm χ^{-1} such that for $x \leftarrow \chi(i; r)$ and $r' \leftarrow \chi^{-1}(i, x)$, (i, x, r) and (i, x, r') are computationally indistinguishable.*

Most known constructions of trapdoor permutations have reverse domain sampler [DN00].

2.2.2 Multi-party Computation Protocol

A standard secure multi-party computation (MPC) protocol allows n parties, $P_1, \ldots, P_n$, each holding a private input x_i, to securely compute a function f on their private data, $f(x_1, \ldots, x_n)$, without revealing x_i to other parties.

Behavior of parties in MPC We say a party in an MPC protocol is *honest* if it strictly follows the protocol. We say a party is *semi-honest or honest-but-curious* if it follows the protocol, but may perform additional computation to learn more information from the protocol. We say a party is *(fully) malicious* if it may deviate from the protocol arbitrarily.

Adversaries An adversary $\mathcal{A}$ corrupting a subset of parties in the protocol receives all messages directed to the corrupted parties and controls all messages that they send. The corrupted parties share all information they have including logs of activities before being corrupted. We say an adversary is *semi-honest* (resp. *malicious*) if the parties it corrupts behave semi-honestly (resp. maliciously).

A *static* adversary only learns such information after he chooses all corrupted parties. On the other hand, an *adaptive* adversary will learn the information at the time of choosing, and may use such information to adaptively choose additional parties to corrupt. Moreover, the *adaptive* adversary may wait for some period of time before corrupting additional parties.

Setup vs. Plain Model We say a protocol is in *setup model* or *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 *plain model*.

2.2.3 Security Models

Here we define some security models for multi-party computation protocols.

Real world/Ideal world paradigm Consider an ideal world where there exists an incorruptible trusted party, called *functionality* $\mathcal{F}$. If such party exists, each party P_i can just sends its input x_i to $\mathcal{F}$. Then $\mathcal{F}$ performs the computation, say of a function f, then returns the output $f(x_1, \ldots, x_n)$ to the designated parties. Clearly, each party knows nothing other than what it can deduce from the output as long as $\mathcal{F}$ keeps everything secret. However, in the real world, such trusted party does not exist. Instead, the parties need to participate in a protocol Π to compute $f(x_1, \ldots, x_n)$. Informally, the protocol Π is secure if an adversary learns no more information from Π in the real world than he would in the ideal world.

Formally, for every (PPT or unbounded) (static or adaptive) adversary $\mathcal{A}$ in the real world, there exists a corresponding simulated adversary $\mathcal{S}$ in the ideal world with black-box access to $\mathcal{A}$ satisfying the following: let $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}}(f, \vec{x})$ denote the joint output of the ideal-world adversary $\mathcal{S}$ and parties $P_1, \ldots, P_n$; let $\mathsf{REAL}_{\Pi,\mathcal{A}}(f, \vec{x})$ denote the joint output of the real-world adversary $\mathcal{S}$ and parties $P_1, \ldots, P_n$; then for all functions f and for all input vectors $\vec{x}$, $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}}(f, \vec{x})$ and $\mathsf{REAL}_{\Pi,\mathcal{A}}(f, \vec{x})$ are (computationally or statistically) indistinguishable.

Indistinguishability At times, we need to consider a different notion of security for MPC protocols with small number of rounds. This can be unavoidable as a simulator that can retrieve an input from adversaries in one round need to "break" the security of a cryptosystem in order to do so. Such situation is discussed in more details for the case of oblivious transfer (OT) in [HK12, AIR01]. In this model, a protocol is secure if an adversary cannot distinguish

the protocol with different inputs as long as the outputs are the same, using *only his view* as opposed to every party.

Formally, let $\mathsf{View}_{\Pi,\mathcal{A}}(f, \vec{x})$ denote the collection of messages an adversary $\mathcal{A}$ receives in an execution of protocol Π on a function f and an input vector $\vec{x}$. We say the protocol Π has *indistinguishability-based computational privacy against* $\mathcal{A}$ if for any two input vectors $\vec{x}, \vec{x}'$ such that $x_i = x_i'$ for each corrupted party P_i, $[\mathsf{View}_{\Pi,\mathcal{A}}(f, \vec{x}) | f(\vec{x}) = y]$ and $[\mathsf{View}_{\Pi,\mathcal{A}}(f, \vec{x}' | f(\vec{x}') = y]$ are computationally indistinguishable for all y.

CHAPTER 3

Hidden Graph Model: MPC with Low Communication Locality

3.1 Introduction

Secure multi-party computation (MPC) [Yao86, GMW87, BGW88, CCD88] is one of the most fundamental primitives in cryptography. A lot of progress has been made in several efficiency measures such as communication complexity, round complexity, and computation complexity. However, most MPC protocols require all parties to communicate directly with each other over a complete network of point-to-point channels and/or have access to a broadcast channel. While this requirement is achievable when the number of participant is small, in large networks, such as the internet, this assumption can be infeasible.

Boyle, Goldwasser and Tessaro [BGT13] recognize this problem and introduce a new metric called *communication locality*. The communication locality of a protocol is the total number of different point-to-point channels that each party uses in the protocol. In other words, it is the number of other parties that each party communicate with during the protocol. The protocols in [BGT13] for computing any polynomial-time function f achieve a communication locality of polylog(n), assuming a public-key infrastructure (PKI), a common reference string (CRS), and the existence of a semantically secure public-key encryption and existentially unforgeable signatures. An example of a scenario where the complexity of the function may be much smaller than the number of parties, is when securely computing the output of a sublinear algorithm, which takes inputs from a small subset of $q = o(n)$ of parties. Sublinear algorithms are particularly useful for computing statistics on large popu-

lations. By assuming, in addition to the PKI and semantically secure public-key encryption, the existence of a multi-signature scheme [MOR01, LOS06], a certifiable fully homomorphic encryption (FHE) scheme [BGV12, BV14], and simulation-sound adaptive non-interactive zero-knowledge (NIZK) [BFM88, FLS99], [BGT13] also obtains a protocol for computing sublinear functions, which communicates $\mathcal{O}((\lambda + n)\text{polylog}(n))$-bit messages.

However, their protocols are only secure against static adversaries corrupting up to $t < \frac{n}{3}$ parties. Thus, the solution of [BGT13] leaves two open questions:

- Can we achieve low communication locality and round complexity while tolerating adaptive adversaries?
- Can we achieve low communication locality while tolerating up to $t < \frac{n}{2}$ corrupted parties?

In this work, we answer both questions affirmatively. We construct an MPC protocol with low communication locality and round complexity tolerating adaptive adversaries corrupting up to $t < \frac{n}{2}$ parties assuming a PKI, a symmetric-key infrastructure (SKI) and trapdoor permutations with reverse domain sampler.

3.1.1 Previous Work

Byzantine agreement The problem of Byzantine Agreement (BA) [PSL80, LSP82] is for n parties, each with initial input v_i, to agree on an output v. If, for all honest parties, $v_i = v^*$, then $v = v^*$. Such guarantee must hold in the presence of an adversary corrupting t parties. A BA protocol can be viewed as an MPC protocol for a specific class of functions. The known optimal bounds of t for BA are $t < \frac{n}{3}$ for the plain model [PSL80, LSP82], and $t < n$ for the model where parties share a PKI [DS83].

BA/MPC on incomplete networks Another line of work on BA [DPP88, Upf94, CGO10, CGO12] and MPC [GO08] protocols has been done over a sparse, public network. In the

case of BA, Dwork *et al.* [DPP88] construct various graphs of specific degrees on which one could run BA protocols. For example, they construct a graph G of degree $d = O(n^\epsilon)$, for any constant $0 < \epsilon < 1$, along with a BA protocol in which every party in the protocol communicates only with its neighbors in G. Such a protocol could tolerate $t = \alpha n$ corrupt parties for some constant $\alpha < \frac{1}{3}$. As another example, they also construct a graph of constant degree, along with a BA protocol, that could tolerate $t = \mathcal{O}(\frac{n}{\log n})$ corrupted parties.

Since in these models, the communication graph is public and chosen prior to the adversary corrupting parties, one cannot hope to achieve BA among all honest parties as an adversary could always corrupt just the neighbors of an honest party, thereby isolating it. Hence, Dwork *et al.* [DPP88] introduce and achieve a notion of *almost everywhere (a.e.)* BA protocol which "gives up" a small number of honest parties, i.e., provides no guarantees for them. For the case of MPC, Garay and Ostrovsky [GO08], introduce the notion of almost-everywhere MPC (similar to a.e. BA) and show how to take any a.e. BA protocol and convert it into an a.e. MPC with the same (asymptotic) parameters.

We remark that all the above protocols provide information-theoretic security against an adaptive, computationally-unbounded adversary.

BA/MPC on complete networks with low communication locality There are several models where parties are connected by a complete network, but only communicate with small number of other parties during the protocol. The works of King, Saia, Sanwalani, and Vee [KSS06a, KSS06b] consider these models and construct protocols for the task of leader election as well as a.e. BA with communication locality of $\mathcal{O}(\log^c n)$ for some constant $c > 1$. In fact, King *et al.* show a stronger result and construct protocols in which every party only sends $\mathcal{O}(\log^c n)$ bits in the entire protocol. However, unlike the works on incomplete networks, the works of King *et al.* only consider the case of *static* adversaries. These works also provide information-theoretic security.

To overcome the drawback of having to give up some honest parties, Boyle, Goldwasser and Tessaro *et al.* [BGT13] consider a computationally bounded adversary, and using cryp-

tographic assumptions and a PKI, they construct MPC protocols with a communication locality of $\mathcal{O}(log^c n)$ for some constant $c > 1$. They tolerate αn corrupted parties for any constant $\alpha < \frac{1}{3}$. Similar to King *et al.*'s, the protocol of Boyle *et al.* is only secure against static adversaries.

3.1.2 Our Result

In this paper, we construct a secure multi-party computation protocol with polylogarithmic communication locality for both static and adaptive adversaries corrupting any $t < n/2$ parties which is optimal [GMW87, Cle86]. Our constructions assume a PKI and an SKI. Furthermore, our protocols have polylog(n) round complexity.

In more detail, we show the following:

Theorem 3.1.1 (Informal)**.** *Assuming a PKI, an SKI and trapdoor permutations with reverse domain sampler, there exists an MPC protocol secure against an adaptive adversary corrupting up to $t < n/2$ parties, and satisfying the following conditions with overwhelming probability:*

- *(Polylogarithmic communication locality) Every party communicates with at most $\mathcal{O}(\log^{1+\epsilon} n)$ other parties, for some constant $\epsilon > 0$.*
- *(Polylogarithmic round complexity) The protocol terminates after $\mathcal{O}(\log^{\epsilon'} n)$ rounds, for some constant $\epsilon' > 0$.*

We note that our protocols do not "give up" any honest party. In this case, the best communication locality that one can achieve is $\omega(\log n)$. Otherwise, if a party only communicates with $\mathcal{O}(\log n)$ other parties, an adversary can simply guess with non-negligible probability, and corrupt them, thereby isolating this honest party. Hence, our protocols are near optimal in terms of communication locality as well.

3.1.3 Our Techniques

We now summarize our main techniques and provide a high-level overview of our MPC construction. Before we do that, let us first describe our model in a bit more detail. All parties are connected via a complete network of point-to-point channels. For simplicity, we assume that the channels are secure; however, as we assume a public-key infrastructure (PKI), these channels can be implemented by encryption and authentication [GMW87]. In addition, our construction assumes a symmetric-key infrastructure (SKI), where every pair of parties P_i, P_j shares a uniformly random key $sk_{i,j} \in \{0,1\}^\lambda$ for some security parameter λ.

SKI as a private graph setup Central to our results is a novel way of interpreting a symmetric key-infrastructure into a special type of setup, which we refer to as *hidden-graph setup*.

Let $G = (V, E)$ be an undirected graph, where $V = [n]$ is the vertex set and E is the set of edges in G. We let $G(n, p)$ denote the Erdős-Rényi random graph on n vertices where for every $i, j \in V$, $\Pr[(i, j) \in E] = p$. We refer to such a graph as a *p-random graph*.

We say that the parties P_i, $i \in [n]$, hold a *hidden p-random graph setup* if, after sampling $G = G(n, p)$, every party $i \in [n]$ is given his corresponding neighbor set $\Gamma_G(i) \subseteq V$ and no other information on E. Thus, if P_i communicates with q other parties, his setup will be of size $q \log(n)$. Throughout this paper we only consider $p = \frac{\log^{1+\epsilon}(n)}{n}$ for some $\epsilon > 0$. Thus, in our setting, $q = \text{polylog}(n)$ with overwhelming probability. Hence, the private graph setup is also of size $\text{polylog}(n)$ for each party. Whenever p is clear from the context we might omit p and just refer to the setup as a *(hidden) random graph setup*.

We now show how such a setup can be efficiently (and locally) computed from a SKI. Recall that in a SKI every pair of parties P_i and P_j is given a uniformly random key $sk_{i,j}$. We use this key as a seed to a pseudo-random function (PRF). Parties P_i and P_j will use the PRF (keyed with $sk_{i,j}$) to (locally) compute the random coins needed to sample (i, j) for

the graph G; i.e., P_i and P_j will use the output of the PRF as coins in a sampling algorithm which picks a bit b to be 1 with probability p. If $b = 1$, then P_i and P_j will communicate with each other directly in the protocol and (i, j) will be an edge in the communication graph G. The security of the PRF ensures that the bit b computed as above is distributed indistinguishably from the output of the sampling algorithm on uniformly random coins. Without loss of generality, we will henceforth assume that the PRF keys that parties share can be used to sample as many independent random graphs as needed.

Our adaptively secure construction will make use of several (polylogarithmic) independent hidden random graphs. A sequence of D-many hidden random graphs that is indistinguishable from a sequence of D independent p-random graphs can be generated as above, by querying the PRF on distinct (fixed) inputs.

Overview of our construction The main part of our construction is a protocol for reliable message transmission (RMT) in the communication locality setting. Such a protocol allows a sender P_i to reliably send a message to a receiver P_j. Note that as we assume a completely connected network, a trivial way of implementing RMT would be for party i to use the point-to-point channel he shares with each P_j, $j \in [n]$. However, our goal is to achieve RMT where each party utilizes only a polylogarithmic number of its direct point-to-point channels. Clearly, in such a setting we cannot allow an adversary to know which parties an honest party P_i communicates with, as this would enable the adversary to "cut-off" party P_i from the rest of the parties by corrupting all of its neighbors.

This is where we utilize the hidden-graph setup: every party will only exchange messages with its neighbors in each hidden graph and ignore all other parties. We note that the adversary might try to send messages to honest parties using all the corrupted parties. However, the honest parties will ignore messages from all parties that are not their neighbors in their hidden graphs. We show that the adversary who corrupts up to any constant fraction $q < 1$ of parties cannot make the length of the shortest honest path between any two honest parties to be greater than $\log^{\epsilon'}(n)$, for some $\epsilon' > 0$, except with negligible probability. In

particular, if G' denotes the graph that is obtained by deleting from G all parties/nodes that such an adversary corrupts, then with overwhelming probability, every two vertices in G' (i.e., every two honest parties) are connected in G' by a path of length at most $\log^{\epsilon'} n$.

Thus, parties can achieve RMT by simply "flooding" the network: Party P_i sends message m, signed under its signing key, to all its neighbors; then, for $\log^{\epsilon'}(n)$ rounds, all parties in every round, forward (the first validly signed) message that they receive to all its neighbors. Since i and j are connected by a path of length $D = \log^{\epsilon'} n$ in G', then after D rounds, P_j will receive at least one copy of m that is signed under P_i's signing key and hence will reliably receive the message m. Observe that the above RMT protocol tolerates any constant fraction $q < 1$ of corruptions (i.e., up to $t \leq qn$ corrupted parties) and requires a standard PKI for digital signatures. We assume standard digital signatures secure against chosen-plaintext attacks. Furthermore, since the message is guaranteed to reach all honest parties within D rounds, the above RMT protocol can be used to have a message sent to all honest parties.We note, however, that if the sender is corrupted, there is no guarantee that the message is sent consistently.

Unfortunately, the above approach only works for a static adversary. The reason is that, while corrupting parties (even adaptively) and learning their setup, does not reveal anything about the hidden graph other than the neighbors of corrupted parties themselves, the protocol itself might reveal whether or not $(i, j) \in E$ for honest parties P_i, P_j. For example, if an adversarial party P_i sends a message to another adversarial party P_j, and P_j receives this message in 2 rounds, then it must be the case that there exists a path of length 2 between i and j. If the adversary sends different messages to each neighbor of P_i, he will know which neighbor of P_i directly connect to a neighbor of P_j. One idea might be to have P_i randomly delay sending its message; however, the adversary can still learn some information though less precise. Note that we want to use RMT for every pair of parties; thus, the adversary might use information on the hidden graph learned in an execution of RMT with a corrupted sender and/or receiver to attack another RMT with honest sender and receiver. As a result, constructing an RMT protocol for the adaptive-corruption case

ends up being much more challenging than in the static case.

The high-level idea behind the protocol in the adaptive case is to sample a new Erdős-Rényi random graph $G = G(n, p)$, with $p = \frac{\log^{\epsilon} n}{n}$, at *every round* of the protocol. As long as the total number of rounds of the protocol is polylogarithmic, so will be the total number of point-to-point channels that an honest party uses (since in each round, every honest party might speak to at most polylog(n), potentially new, neighbors). The intuition for choosing a different hidden random graph for each round is that any corruptions made by the adversary before round i is independent of the graph selected in round i and hence this would be equivalent to the static adversary case. However, now proving that honest parties can communicate reliably (and that there exists a path of polylogarithmic length between any two honest parties) is more complicated.

Having an RMT, the next step is to design the MPC protocol. Our goal is a protocol with full security, tolerating optimal $t < \frac{n}{2}$ corruptions [Cle86, GMW87]. One idea to achieve this, is as follows: Since we have already established RMT between any two honest parties, we can invoke any known MPC protocol Π secure for $t < \frac{n}{2}$ assuming authenticated channels, over the virtual network induced by RMT. Whenever party P_i is instructed in Π to send a message m to party P_j, we invoke RMT for this purpose. This approach would give an MPC protocol tolerating up to $t < \frac{n}{2}$ corruptions.

Observe that in our adaptively secure protocol, increase of the round complexity implies the same (asymptotic) increase of the honest parties' communication locality. Indeed, since using our RMT, every party communicates with $\mathcal{O}(\log^{c} n)$ (potentially new) parties in every round $1 \leq \rho \leq D$, we can only afford to run a protocol that runs in $\log^{c'} n$ number of rounds for some $c' > 0$.

Thus, in order for the above idea to work we need an adaptive MPC protocol over point-to-point authenticated channels which terminates in polylog(n) rounds. Such a protocol can be obtained by taking any constant-round MPC protocol that utilizes a point-to-point network of secure channels and a broadcast channel such as the protocol in [BMR90], and modifying it as follows: transmission over the point-to-point secure channels are emulated

by calls to our RMT protocol where the message is encrypted using the receiver's public key; and transmission over the broadcast channel are emulated by a (randomized, authenticated) broadcast protocol which terminates in polylog(n) rounds such as the protocol in [KK06]. Our MPC protocol results from emulating these protocols using our RMT.

3.2 Definitions and Background

Public-key and symmetric-key infrastructure A public-key infrastructure (PKI) is a network with a third-party trusted authority, who can generate and distribute public and secret keys for encryption, decryption, authentication and verification according to some public-key cryptosystem. As already mentioned earlier, we assume a PKI setting, where all parties share a PKI for digital signature scheme. In other words, before the beginning of the protocol, every party has a private signing key sk_i and public verification key vk_i pair for a secure signature scheme authentically generated and distributed.

Similarly, a symmetric-key infrastructure (SKI) is a network with a third-party trusted authority, who can generate and distribute secret keys according to some symmetric-key cryptosystem. Our construction assume an SKI for hidden random graphs generation. In other words, in addition to signature key pair, every pair of party P_i, P_j share a secret key $sk_{i,j}$. Additionally, parties are connected by a fully connected *synchronous* network; however, in our constructions, every party will only communicate with polylog(n) other parties.

Adversaries and security models In our main result of this chapter, we will consider adaptive corruption up to $t < \frac{n}{2}$ parties by a *rushing* adversary. Such an adversary is allowed to corrupt parties dynamically during the protocol, and depending on his view, the adversary may postpone the sending of any given round's messages until after he receives the messages from the honest parties. We consider the standard simulation-based notion of security for multi-party protocols via the real/ideal world paradigm as defined in the previous chapter. We assume that the number of parties $n > \lambda$; thus, our parameter will be based on n.

We refer to the setting with low communication locality i.e. each party communicates with polylogarithmic number of other parties as *locality model*.

Reliable message transmission A reliable message transmission (RMT) protocol allows any party P_i to reliably send a message to a receiver P_j. For a completely connected network, this protocol is trivial. However, our protocol only allow each party to communicate with at most polylog(n) other parties. An RMT protocol can be achieved by flooding the network with a signed message as discussed earlier. A secure message transmission (SMT) protocol is an RMT protocol that guarantees that the receiver receives the message authentically and privately. By establishing an SMT for every pair of parties, we can simulate a complete network while maintaining polylogarithmic locality.

3.2.1 Cryptographic Building Blocks

Authenticated broadcast We review the definition of authenticated broadcast protocol [PSL80, LSP82] below.

Definition 3.2.1. *A protocol for parties $\mathcal{P} = \{P_1, \cdots, P_n\}$, where a distinguished player $P^* \in \mathcal{P}$ holds an initial input m, is a* broadcast protocol tolerating t malicious parties *if the following conditions hold for any adversary corrupting at most t parties:*

- *Agreement: All honest parties output the same value v.*
- *Validity: If P^* is honest, then $v = m$.*

Theorem 3.2.2 ([KK06])**.** *Assuming a PKI, there exists a protocol Π_{BC} which achieves broadcast with overwhelming probability against $t < n/2$ adaptive corruptions, running for $\log^c(n)$ rounds on a complete network, for some constant $c > 0$.*

Non-committing encryption In order to obtain adaptive security for our SMT protocols, we require non-committing encryption [CFG96]. We recall the definition of non-committing (bit) encryption.

Definition 3.2.3. *A* non-committing bit encryption scheme *is of a tuple of PPT algorithms* $(\mathsf{NCGen}, \mathsf{NCEnc}, \mathsf{NCDec}, \mathsf{NCSim})$ *where* $(\mathsf{NCGen}, \mathsf{NCEnc}, \mathsf{NCDec})$ *is a secure encryption scheme and* NCSim *is a simulation algorithm such that on input* 1^λ*, it outputs* $(e, c, \sigma_G^0, \sigma_E^0, \sigma_G^1, \sigma_E^1)$ *with the following property: for* $b \in \{0,1\}$*, the following distributions are computationally indistinguishable:*

- *The joint view of an honest sender and an honest receiver in a normal encryption of* b*:*

$$\{(e, c, \sigma_G, \sigma_E) | (e, d) = \mathsf{NCGen}(1^\lambda; \sigma_G), c = \mathsf{NCEnc}(e, b; \sigma_E)\}$$

- *The simulated view of an encryption of* b*:*

$$\{(e, c, \sigma_G^b, \sigma_E^b) | \mathsf{NCSim}(1^\lambda) \to (e, c, \sigma_G^0, \sigma_E^0, \sigma_G^1, \sigma_E^1)\}$$

We note that in order to encrypt a message of length l, l independent public keys are required, one for each bit of the message.

Theorem 3.2.4 ([DN00])**.** *Assuming trapdoor permutations with reverse domain sampler, there exists a non-committing bit encryption scheme.*

3.2.2 Graphs

As we mention earlier, a Erdős-Rényi random graph or p-random graph on n vertices, denoted $G(n,p)$, is a graph where each pair of vertices is connected with probability p independently of other pairs. Unless stated otherwise, we assume that $p = \frac{d}{n}$ where $d = \log^{1+\epsilon} n$ for some constant $\epsilon > 0$. It is easy to show using Chernoff bound that, except with negligible probability, every vertex of $G(n,p)$ will have degree $\mathcal{O}(d) = \mathcal{O}(\log^{1+\epsilon} n)$.

Let $G = (V, E)$ be an undirected graph. Let $\Gamma_G(v)$ denote a set of all neighbors of v in G. Let $S \subseteq V$ be a subset of vertices with $|S| \leq |V|/2$. Denote $\overline{S} = V \setminus S$. Let $\Gamma_G(S) = \{v \in V | \text{there exists } v' \in V \setminus S \text{ such that } (v, v') \in E\}$ be the set of all neighbors of S that are not in S. Let $e_G(S, S') = |\{(v, v') \in E | v \in S, v' \in S'\}|$.

Definition 3.2.5. *The* edge expansion *of a graph G on $V = [n]$ is*

$$h(G) = \min_{S \subseteq V, 0 < |S| \leq \frac{n}{2}} \frac{e_G(S, \overline{S})}{|S|}.$$

Definition 3.2.6. *The* vertex expansion *of a graph G on $V = [n]$ is*

$$h_{out}(G) = \min_{S \subseteq V, 0 < |S| \leq \frac{n}{2}} \frac{|\Gamma_G(S)|}{|S|}.$$

A graph where every vertex has degree $\mathcal{O}(d)$ with $\mathcal{O}(d)$ edge or vertex expansion is an *expander graph*. For any pair of vertices on such an expander graph, a random walk from one vertex reaches another within $\mathcal{O}(\log n)$ steps. Thus, an expander graph has logarithmic diameter.

3.3 Statically Secure RMT with Low Communication Locality

In this section and the following section, we construct a reliable message transmission protocol between every pair of honest parties in the locality model, assuming a standard PKI (for digital signatures) as well as an SKI, against static adversary and adaptive adversary, respectively. The constructions in both sections tolerate any constant fraction of corrupted parties. In other words, we allow the number of corrupted parties to be $t \leq qn$, for any constant $q < 1$. We note that this requirement is weaker than that of fully secure MPC, which requires majority of parties to be honest.

We first show an RMT protocol that is secure against static corruptions. This will illustrate some of the ideas that are needed for our adaptively secure construction.

Setup phase. Recall that we work in a model in which parties share a public-key as well as a symmetric-key infrastructure. That is, in the setup phase, party P_i receives a private key sk_i for a signature scheme, and every party P_j receives the public key vk_i corresponding to sk_i, for all $i \in [n]$. The SKI allows for a hidden p-random graph setup, with $p = \frac{\log^{1+\epsilon} n}{n}$ (for appropriately chosen $\epsilon > 0$), as explained above.

Construction idea. The hidden graph setup ensures that the adversary does not get to know whether party P_i communicates with party P_j, unless he corrupts one of them. We first show that given such a hidden p-random graph, an adversary who (non-adaptively) corrupts any constant fraction $q < 1$ of the parties cannot isolate any of the honest parties. In fact, we show a much stronger property for the graph G' formed by removing (in the hidden graph) $t \leq qn$ corrupted vertices; namely, that with overwhelming probability (in n), every pair P_i, P_j of honest parties is connected by a path of length at most $N = \log^{\epsilon'}(n)$, for some $\epsilon' > 0$ which depends only on ϵ.

Note that since parties start with a PKI, we only require that honest parties P_i, P_j are connected by a path of length $N = \log^{\epsilon'}(n)$, for some $\epsilon' > 0$ in graph G'. Parties can then achieve RMT by simply "flooding" the network; i.e., party P_i will simply send message m, signed under its signing key, to all its neighbors. Next, all parties in every round simply forward (the first validly signed) message that they receive to all of its neighbors. If P_i and P_j are connected by a path of length N in G', then after N rounds j will receive at least one copy of m that is signed under i's signing key, and hence will reliably receive the message m.

3.3.1 Connectivity against Static Adversary

Let $G = (V, E)$ be a hidden p-random graph on n vertices representing a network. Let $\mathcal{A}$ be an adversary who non-adaptively chooses a set of parties to corrupt and by doing so learns all their neighbors in G. Let $U \subseteq V$ be the set of vertices representing corrupted parties. Then U is independent of E.

Lemma 3.3.1. *Let n be a positive integer, $d = \log^{1+\epsilon} n$ for some $\epsilon > 0$ and $p = \frac{d}{n} = \frac{\log^{1+\epsilon} n}{n}$. Let $G = (V, E)$ be a p-random graph on n vertices. Let $U \subseteq V$ chosen independent of E, with $|U| \leq qn$ for some constant $q < 1$. Let G' be the induced subgraph on $V' = V \setminus U$. Then the diameter of G' is at most $\mathcal{O}(\log n)$ with overwhelming probability.*

Proof. Since each pair of vertices in G' is connected with probability p independently of U, G' is a p-random graph on $n' = (1 - q)n$ vertices. Let $0 < k < \frac{1-q}{2}$. Then, for each $S \subseteq V'$

with $|S| = r \leq \frac{n'}{2}$, we have

$$e_{G'}(S, \overline{S}) = \sum_{v \in S, v' \in \overline{S}} X_{v,v'},$$

where $\overline{S} = V' \setminus S$ and $X_{v,v'}$ is the indicator whether there exists an edge between v and v'. Then

$$\mathbb{E}[e_{G'}(S, \overline{S})] = \sum_{v \in S, v' \in \overline{S}} \mathbb{E}[X_{v,v'}] = |S||\overline{S}|p = r(n' - r)p.$$

By the Chernoff bound,

$$\Pr[e_{G'}(S, \overline{S}) < kd|S|] \leq e^{-\left(1 - \frac{kn}{n'-r}\right)^2 r(n'-r)p} = \left(e^{-\frac{\left(1 - \frac{kn}{n'-r}\right)^2 (n'-r)}{2n}}\right)^{rd} = \left(e^{-\frac{\left(\frac{n'-r}{n} - k\right)^2}{2 \cdot \frac{n'-r}{n}}}\right)^{rd}.$$

Since $0 < r < \frac{n'}{2}$, we have

$$\frac{1-q}{2} = \frac{n'}{2n} \leq \frac{n'-r}{n} \leq \frac{n'}{n} = 1 - q < 1.$$

Thus,

$$\frac{\left(\frac{n'-r}{n} - k\right)^2}{2 \cdot \frac{n'-r}{n}} \geq \frac{1}{2} \cdot \left(\frac{1-q}{2} - k\right)^2 = c > 0.$$

For $d = \log^{1+\epsilon} n$, we have

$$\Pr[e_{G'}(S, \overline{S}) < kd|S|] \leq \left(e^{-c}\right)^{rd} = \left(\frac{1}{n^{c' \log^{\epsilon} n}}\right)^r,$$

and by the union bound, the probability that $e_{G'}(S, \overline{S}) < kd|S|$ for some subset S, $|S| \leq |V'|/2$ is bounded by

$$\begin{aligned}
\sum_{r=1}^{\frac{n'}{2}} \sum_{S, |S|=r} \Pr[e_{G'}(S, \overline{S}) < kd|S|] &\leq \sum_{r=1}^{\frac{n'}{2}} \binom{n'}{r} \left(\frac{1}{n^{c' \log^{\epsilon} n}}\right)^r \\
&\leq \sum_{r=1}^{\frac{n'}{2}} n^r \left(\frac{1}{n^{c' \log^{\epsilon} n}}\right)^r \\
&= \sum_{r=1}^{\frac{n'}{2}} \left(\frac{1}{n^{c' \log^{\epsilon} n - 1}}\right)^r \\
&< \frac{\frac{1}{n^{c' \log^{\epsilon} n - 1}}}{1 - \frac{1}{n^{c' \log^{\epsilon} n - 1}}} = \mathrm{negl}(n),
\end{aligned}$$

Hence, G' is an expander with edge expansion kd with overwhelming probability. Thus, G' has at most $\mathcal{O}(\log n)$ diameter with overwhelming probability. □

For a given graph $G = ([n], E)$, we say that two parties P_i and P_j, $i, j \in [n]$ are *G-connected by an honest path of length l* if there exists a sequence of connected nodes $\texttt{PATH}(i, j)$ from i to j in G such that for every node $k \in \texttt{PATH}(i, j)$, node k is honest, and $|\texttt{PATH}(i, j)| = l$.

Corollary 3.3.2. *Let $\epsilon > 0$, $p = \frac{\log^{1+\epsilon} n}{n}$, and G be a hidden p-random graph on $[n]$. For any adversary who (non-adaptively) corrupts at most $t = qn$ parties, then except with negligible (in n) probability, there exists some $\epsilon' > 0$ which depends only on ϵ such that any two honest parties are G-connected by an honest path of length at most $\log^{\epsilon'}(n)$.*

3.3.2 RMT Construction

Let G be a hidden p-random graph constructed from an SKI. Here we describe a reliable message transmission protocol based on G, denoted by $\mathsf{RMT}_{i,j}(m)$, in the locality model for a sender P_i to send a message m to a receiver P_j. Let $\Gamma(i) = \Gamma_G(i)$ denote the set of P_i's neighbors in G. We describe the protocol in Figure 3.1.

The security of protocol $\mathsf{RMT}_{i,j}(m)$ follows from the above corollary, as no matter how the (static) adversary chooses the corrupted parties he cannot increase the diameter of the graph defined by the honest parties and the hidden graph setup to more than polylog(n).

Theorem 3.3.3. *Let $T \subseteq [n]$ be the set of (non-adaptively) corrupted parties with $|T| = t \leq qn$ for any constant $0 < q < 1$. Assuming a PKI and an SKI, $\mathsf{RMT}_{i,j}$ is a secure RMT protocol between any two honest parties P_i, P_j, $i, j \in [n] \setminus T$ satisfying the following two conditions with overwhelming probability:*

- *every party communicates with at most $\mathcal{O}(\log^{1+\epsilon} n)$ other parties;*
- *the protocol terminates after $\mathcal{O}(\log^{\epsilon'} n)$ rounds, for some $\epsilon' > 0$.*

Proof. Since Corollary 3.3.2 shows that any message sent by an honest party P_i will reach any honest P_j within $\mathcal{O}(\log^{\epsilon'}(n))$ rounds, it follows from the unforgeability property of the

Protocol $\mathsf{RMT}_{i,j}(m)$

Round 1: P_i sends $(m, \mathtt{sig}_{sk_i}(m))$ to all parties in $\Gamma(i)$.

For each round $\rho = 2, \ldots, \log^{\epsilon'}(n)$**:**

- For every P_k, $k \in [n] \setminus \{i, j\}$: If a message (m, σ), where σ is P_i's valid signature on m, was received *for the first time* from some of its neighbors (in $\Gamma(k)$) in the previous round, then P_k sends (m, σ) to all its neighbors and halts. (If multiple validly signed pairs were received in that round for the first time, then take the first one in a lexicographic order.)

- For the receiver P_j: If a message (m, σ), where σ is party i's valid signature on m, is received *for the first time* from some parties in $\Gamma(j)$ then output m and halt. (If multiple validly signed pairs are received in that round for the first time, then take the first one in a lexicographic order.)

Figure 3.1: Reliable message transmission protocol (static case)

signature scheme that P_j will always accept the message sent by honest P_i. Hence, the above protocol is a secure RMT protocol. The communication locality of the protocol follows from the degree of p-random graph G which is $\mathcal{O}(pn) = \mathcal{O}(d) = \mathcal{O}(\log^{1+\epsilon} n)$, except with negligible probability. □

3.3.3 Parallel Composition of RMT

In our MPC construction, we will require all parties to execute their respective RMT protocols in parallel simultaneously. That is, let $m_{i,j}$ be the message that P_i wants to send to P_j via the RMT protocol, denoted $\mathsf{RMT}_{i,j}(m_{i,j})$ as above. Now, let $\mathsf{RMT}_{\mathtt{all}}(\vec{m})$ denote the protocol executed by all parties when $\mathsf{RMT}_{i,j}(m_{i,j})$ for all $i, j \in [n]$ are executed in parallel in the same network graph G. That is, in round k of $\mathsf{RMT}_{\mathtt{all}}(\vec{m})$, all parties execute the kth round of protocol $\mathsf{RMT}_{i,j}(m_{i,j})$, for all $i, j \in [n]$. $\mathsf{RMT}_{\mathtt{all}}$ is composed of n^2 individual RMT protocols. We have the following corollary:

Corollary 3.3.4. *Under the same assumption as Theorem 3.3.3,* $\mathsf{RMT}_{\mathtt{all}}$ *is a secure RMT protocol for sending* $m_{i,j}$ *from* P_i *to* P_j *for every pair of honest parties* P_i, P_j, $i, j \in [n]$, *satisfying the following properties with overwhelming probability:*

1. *every party communicates with at most* $\mathcal{O}(\log^{1+\epsilon} n)$ *other parties;*
2. *the protocol terminates after* $\mathcal{O}(\log^{\epsilon'} n)$ *rounds for some* $\epsilon' > 0$.

Proof. The probability that each $\mathsf{RMT}_{i,j}(m_{i,j})$ fails is negligible for honest P_i, P_j, $i, j \in [n]$. Thus, by the union bound, with overwhelming probability, for every pair of honest P_i, P_j, $i, j \in [n]$, $\mathsf{RMT}_{i,j}(m_{i,j})$ satisfies Theorem 3.3.3. The protocol's round complexity is equal to the maximum round complexity of its components, which is $\mathcal{O}(\log^{\epsilon'} n)$. Since we use the same network graph G for all n^2 individual RMT protocol, the communication locality of every party in $\mathsf{RMT}_{\mathtt{all}}(\vec{m})$ is equal to the communication locality of the party in $\mathsf{RMT}_{i,j}(m_{i,j})$, for any $i, j \in [n]$. Hence, the corollary follows. □

3.4 Adaptively Secure RMT with Low Communication Locality

As discussed in Section 3.1, the above proof technique fails against adaptive adversaries. Informally, the issue is that an adversary can use the rounds in which a corrupted party/relayer receives a message to deduce information on the communication graph. In this section, we describe an RMT protocol that is secure against such an adaptive adversary. The idea is have the parties use a different, independent communication graph for each round in the transmission scheme. As long as the transmission scheme runs in polylog(n) rounds, and in each round, every party communicates with at most polylog(n) (potentially different) parties, the overall locality with be polylog(n).

The main challenge in the above idea is to prove that in this dynamically updated communication graph, the message will reach each recipient through an honest path in at most polylog(n) rounds. Proving this constitute the main technical contribution of our work. The adaptively secure RMT protocol AdRMT is similar to the protocol in the static case, except

that in round ρ, parties forward messages received in the previous round to their neighbors in the communication graph G_ρ. We first describe the corresponding setup that it requires.

Setup phase As in the static case, the parties share both a PKI and an SKI. The SKI will be used here in similar way, except that instead of generating one Erdős-Rényi graph, $G = G(n,p)$ with $p = \frac{\log^\epsilon n}{n}$, it will be used to generate D such graphs, denoted $\mathcal{G} = (G_1, \ldots, G_D)$. These graphs can be sampled using the same PRF key $sk_{i,j}$ that parties P_i and P_j share. As before, every party only knows its own neighbors, and when the adversary corrupts a party P_j, he only learns P_j's neighbors in $G_1, \ldots, G_D$. We shall show that there exists a path of length at most $\mathcal{O}(\log^{\epsilon'}(n))$ between any two honest parties P_i, P_j when we consider the collection of communication graphs $\mathcal{G}$ that selects graph G_ρ as the communication graph in round ρ.

3.4.1 Reachability against Adaptive Adversary

We formally define when a message sent by an honest party P_i will reach another honest party P_j in our hidden graphs model against adaptive adversaries. We define the notion of *reachability* as follows.

Definition 3.4.1. *For a positive integer D and a set V, let $\mathcal{G} = (G_1, \ldots, G_D)$ be an ordered collection of graphs on subsets $(V_1, \ldots, V_D)$ of V. For $1 \leq l \leq D$ and $v \in V_1$, we say $v' \in V_l$ is* reachable from v with respect to $\mathcal{G}$ by a path of length l *if there exist $v_1, \ldots, v_{l-1} \in V$, such that $(v_{i-1}, v_i) \in E(G_i)$, for $i = 1, \ldots, l$, where $v_0 = v$ and $v_l = v'$. We denote $N_l(v) = N_l^{\mathcal{G}}(v) \subseteq V_l$ the subset of all vertices that are reachable from v with respect to $\mathcal{G}$ with a path of length l.*

In order to prove the reachability, we start by showing that an adaptive adversary cannot reduce the size of honest neighbors in the new graph by more than half via the following the Hoeffding's Inequality.

Lemma 3.4.2. (Hoeffding's Inequality [Hoe63]) *Let $S = \{x_1, \ldots, x_N\}$ be a finite set of*

real numbers with $a = \min_i x_i$ *and* $b = \max_i x_i$. *Let* $X_1, \ldots, X_n$ *be a random sample drawn from* S *without replacement. Let* $\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$ *and* $\mu = \frac{\sum_{i=1}^{N} x_i}{N} = \mathbb{E}[X_j]$. *Then for all* $\delta > 0$, $\Pr[\overline{X} - \mu \geq \delta] \leq e^{-\frac{2n\delta^2}{(b-a)^2}}$.

We show the following lemma.

Lemma 3.4.3. *Let* $V = [n]$, *and* $C \subseteq V$, $|C| = m$, *be a subset chosen uniformly at random. Let* $0 < q < 1$ *be a constant and* $U \subseteq V$, $|U| = qn$, *be a subset chosen independently of* C. *Then, for all* $0 < \delta < 1-q$, $|C \backslash U| > (1-q-\delta)m$ *except with probability* $e^{-2m\delta^2}$. *In particular, for* $m = \log^{1+\epsilon'} n$, $|C \setminus U| > \left(\frac{1-q}{2}\right) m$ *except with negligible probability. Furthermore, for* $q = \frac{1}{2} - \epsilon$, $|C \setminus U| > \frac{1}{2}m$ *except with negligible probability.*

Proof. Let $S = \{x_1, \ldots, x_n\}$ where $x_i = 1$ if $i \in U$, 0 otherwise. Then $a = \min_i x_i = 0$, $b = \max_i x_i = 1$ and $\mu = \frac{\sum_{i=1}^{n} x_i}{n} = q$. For each $i = 1, \ldots, m$, let X_i be the indicator of whether each element of C is in U. Then X_i is a random sample drawn from S without replacement, and $|C \cap U| = \sum_{i=1}^{m} X_i = m\overline{X}$. By Hoeffding's Inequality,

$$\Pr[|C \cap U| \geq (q+\epsilon)m] = \Pr[\overline{X} - \mu \geq \delta] \leq e^{-2m\delta^2}.$$

Therefore, except with probability $e^{-2m\delta^2}$, $|C \setminus U| = m - |C \cap U| > (1 - q - \delta)m$.

Now let $m = \log^{1+\epsilon'} n$ and $\delta = \frac{1-q}{2}$. We have that $|C \setminus U| > \left(\frac{1-q}{2}\right) m$ except with probability

$$e^{-2\left(\frac{1-q}{2}\right)^2 \log^{1+\epsilon'} n} = \frac{1}{n^{c \log^{\epsilon'} n}} = \mathrm{negl}(n),$$

where $c = \frac{1}{2}(1-q)^2 \log e$.

Finally, let $q = \frac{1}{2} - \epsilon$ and $\delta = \epsilon$. We have that $|C \setminus U| > \left(1 - \left(\frac{1}{2} - \epsilon\right) - \epsilon\right) m = \frac{1}{2}m$ except with probability $\frac{1}{n^{c' \log^{\epsilon'} n}} = \mathrm{negl}(n)$, where $c' = 2\epsilon^2 \log e$. □

Next, we show prove that at every step of the protocol, even if an adaptive adversary corrupts a constant fraction of the nodes in the random graph, the honest neighbors of any set S of size at most $\frac{n}{d}$ that are not in S will be at least of size $\mathcal{O}(d|S|)$.

Lemma 3.4.4. *Let n be a positive integer, $d = \log^{1+\epsilon} n$ for some $\epsilon > 0$, $p = \frac{d}{n} = \frac{\log^{1+\epsilon} n}{n}$. Let $G = (V, E)$ be a p-random graph on n vertices. Let $U \subseteq V$ such that $|U| \leq qn$ for some constant q, chosen adaptively while learning edges connecting to U. Let G' be the induced subgraph on $V' = V \setminus U$. Then, for any constant $0 < k < \frac{1-q}{2}$, there exists a constant $c > 0$ such that, for sufficiently large n and for any $S \subseteq V'$ with $|S| = r \leq \frac{n}{d} = \frac{1}{p}$, the set of all neighbors of S that are not in S, $\Gamma(S)$, has size at least $kd|S|$ except with negligible probability.*

Proof. Let $0 < k < \frac{1-q}{2}$ and $S \subseteq V'$ with $|S| = r \leq \frac{n}{d} = \frac{1}{p}$. Denote $n' = |V'| \geq (1-q)n$. Since each pair of vertices in G' is connected with probability p independently of other edges, an adaptive adversary learns nothing about G' from edges of U. Thus, G' is a p-random graph on n' vertices. The result follows from vertex expansion of Erdős-Rényi graphs. We include the proof here for completion.

For each $v \in V' \setminus S$, let X_v be the indicator of whether $v \in \Gamma(S) = \Gamma_{G'}(S)$. Then

$$\Pr[X_v = 0] = \Pr[\text{no edge between } v \text{ and any vertex in } S] = (1-p)^r.$$

Since $rp < 1$,

$$\mathbb{E}[X_v] = \Pr[X_v = 1] = 1 - (1-p)^r > \frac{rp}{2}.$$

Then

$$\mathbb{E}[|\Gamma(S)|] = \mathbb{E}[\sum_{v \notin S} X_v] > \frac{(n'-r)rp}{2}.$$

Since the X_v's are independent, by the Chernoff Bound,

$$\Pr[|\Gamma(S)| \leq (1-\delta)\frac{(n'-r)rp}{2}] \leq \Pr[|\Gamma(S)| \leq (1-\delta)\mathbb{E}[|\Gamma(S)|]] \leq e^{-\frac{\delta^2 \mathbb{E}[|\Gamma(S)|]}{2}} \leq e^{-\frac{\delta^2 (n'-r)rp}{4}}.$$

Now let $\delta = 1 - \frac{2kn}{n'-r}$. Since $r \leq \frac{n}{d}$, we have

$$(1-q) - \frac{1}{d} \leq \frac{n'-r}{n} \leq \frac{n'}{n} < 1.$$

Let n be large enough such that $d = \log^{1+\epsilon} n > \frac{2}{1-q-2k}$. Then

$$c_0 = \frac{1}{16} \cdot ((1-q) - 2k)^2 \leq \frac{1}{4} \cdot \left((1-q) - \frac{1}{d} - 2k\right)^2 \leq \frac{\left(\frac{n'-r}{n} - 2k\right)^2}{4 \cdot \frac{n'-r}{n}}.$$

Thus,

$$\Pr[|\Gamma(S)| \le kdr] \le e^{-\frac{\left(1-\frac{2kn}{n'-r}\right)^2 (n'-r)rp}{4}} = \left(e^{-\frac{\left(\frac{n'-r}{n}-2k\right)^2}{4\cdot\frac{n'-r}{n}}}\right)^{dr} \le \left(\frac{1}{e^{c_0}}\right)^{dr} = \left(\frac{1}{n^{c\log^{\epsilon} n}}\right)^{r} = \mathrm{negl}(n).$$

where $c = c_0 \log e$. □

Finally, using the above lemmas, we prove our main lemma.

Lemma 3.4.5. *Let n be a positive integer, $d = \log^{1+\epsilon} n$ for some $\epsilon > 0$, $p = \frac{d}{n} = \frac{\log^{1+\epsilon} n}{n}$ and $D = \mathcal{O}(\log n)$. Let $G_1, \ldots, G_D$ be independent p-random graphs on $V = [n]$. Let $U_1, U_2, \ldots, U_D \subseteq V$ be disjoint subsets of V with $U = \cup_{j=1}^{D} U_j$ such that $|U| = qn$ for some constant $q < 1$, where U_j is chosen independently from $G_{j+1}, \ldots, G_D$, but adaptively, learning the neighbors of U_i in G_i for $i \le j$. Let G'_i be the induced subgraph on $V_i = V \setminus (\cup_{j=1}^{i-1} U_j)$ for $i = 1, \ldots, D$. Then, except with negligible probability, any pair of vertices $v, v' \in V' = V \setminus U$ are reachable with respect to $\mathcal{G}' = (G'_1, \ldots, G'_D)$ by a path of length at most D.*

Proof. For each $v \in V'$, we will show that, except with negligible probability, there exists $l = l(v) < D$ such that $V' \subseteq N_l(v) \cup N_{l+1}(v)$. Hence, by the union bound over $|V'| = (1-q)n$ vertices, the proposition holds except with negligible probability.

Let $v \in V'$ and $0 < k < \frac{1-q}{2}$. For each $i < D$, denote $r_i = |N_i(v) \setminus U_i|$ and observe that $\Gamma_{G'_{i+1}}(N_i(v) \setminus U_i) \subseteq N_{i+1}(v)$. For i such that $r_i \le \frac{n}{d}$, we have

$$|N_{i+1}(v)| \ge |\Gamma_{G'_{i+1}}(N_i(v) \setminus U_i)| > kd|N_i(v) \setminus U_i|$$

except with negligible probability by Lemma 3.4.4.

Since each pair of vertices is connected independently in G'_i, $N_i(v)$ is uniform on V_i. Since U_i is chosen from V_i independently of $N_i(v)$, by Lemma 3.4.3, except with negligible probability,

$$|N_i(v) \setminus U_i| > \left(\frac{1-q}{2}\right)|N_i(v)|.$$

Inductively, $r_i = |N_i(v) \setminus U_i| > \left(\left(\frac{1-q}{2}\right) kd\right)^i$ for all i such that $r_{i-1} \leq \frac{n}{d}$, except with negligible probability. Let l_0 is the largest integer such that $r_{l_0} \leq \frac{n}{d}$. Since for $D = \mathcal{O}(\log n)$, $d^D \gg n$, $l_0 \ll D$.

Let $n' = |V'| = (1-q)n$. There are two possibilities for $r_{l_0+1} = |N_{l_0+1}(v) \setminus U_{l_0+1}|$: either 1) $\frac{n}{d} < r_{l_0+1} \leq \frac{n'}{2}$ or 2) $r_{l_0+1} > \frac{n'}{2}$.

Case 1: Assume that $\frac{n}{d} < r_{l_0+1} \leq \frac{n'}{2}$. Denote $r = r_{l_0+1}$ and $n_0 = |V_{l_0+1}| \geq n'$. Then $\frac{n}{d} = \frac{1}{p} < r \leq \frac{n'}{2} \leq \frac{n_0}{2}$. As in the proof of Lemma 3.4.4, we apply the Chernoff bound to

$$\mathbb{E}[|\Gamma(N_{l_0+1}(v) \setminus U_{l_0+1})|] = (n_0 - r)(1 - (1-p)^r) \geq (n_0 - r)(1 - e^{-rp}),$$

we have

$$\begin{aligned}
\Pr[|\Gamma(N_{l_0+1}(v) \setminus U_{l_0+1})| \leq \frac{n_0}{4}] &\leq e^{-\frac{\left(1-\frac{n_0}{4(n_0-r)(1-e^{-rp})}\right)^2 (n_0-r)(1-e^{-rp})}{2}} \\
&\leq \left(e^{-\frac{\left(\frac{(n_0-r)(1-e^{-rp})}{n_0} - \frac{1}{4}\right)^2}{2 \cdot \frac{(n_0-r)(1-e^{-rp})}{n_0}}} \right)^{n_0} \\
&\leq \frac{1}{c_1^{n_0}} \leq \frac{1}{c_1^{n'}},
\end{aligned}$$

where $c_1 = e^{\frac{\frac{1}{2}(1-e^{-1})-\frac{1}{4}}{2}} > 1$ as $1 - e^{-1} < 1 - e^{-rp} < 1$ and $\frac{1}{2} \leq \frac{n_0 - r}{n_0} < 1$. Thus, except with negligible probability,

$$r_{l_0+2} = |N_{l_0+2}(v) \setminus U_{l_0+2}| \geq \left(\frac{1-q}{2}\right) |\Gamma(N_{l_0+1}(v) \setminus U_{l_0+1})| > \left(\frac{1-q}{8}\right) n_0 \geq \left(\frac{1-q}{8}\right) n'$$

by Lemma 3.4.3. In this case, let $l = l_0 + 2$.

Case 2: $r_{l_0+1} > \frac{n'}{2}$. In this case, let $l = l_0 + 1$.

In both cases, we have $|N_l(v) \setminus U_l| = r_l > c_2 n'$ for some constant $0 < c_2 < 1$ except with negligible probability. Then, for each $v \in V' \setminus N_l(v)$, the probability that v does not connect to any vertex in $N_l(v) \setminus U_l$ is $(1-p)^{r_l} \leq e^{-r_l p} \leq \frac{1}{n^{c_3 \log^\epsilon n}}$, where $c_3 = c_2(1-q)\log e$. By the union bound, the probability that any node in $V' \setminus N_l(v)$ is not in $\Gamma(N_l(v) \setminus U_l) \subseteq N_{l+1}(v)$ is at most $\frac{1}{n^{c_3 \log^\epsilon n - 1}}$, which is negligible. Hence, except with negligible probability, $V' = N_l(v) \cup \Gamma(N_l(v)) \subseteq N_l(v) \cup N_{l+1}(v)$. Therefore, any $v' \in V'$ is reachable from v by a path of length at most D. □

Protocol $\mathsf{AdRMT}_{i,j}(m)$

Round 1: P_i sends $(m, \mathtt{sig}_{sk_i}(m))$ to all parties in $\Gamma_1(i)$.

For each round $\rho = 2, \ldots, \log^{\epsilon'}(n)$**:**

- For every P_k, $k \in [n] \setminus \{i, j\}$: If a message (m, σ), where σ is P_i's valid signature on m, was received *for the first time* from some of its neighbors in $\Gamma_{\rho-1}(k)$ in the previous round, then P_k sends (m, σ) to all its neighbors in $\Gamma_\rho(k)$ and halts. (If multiple validly signed pairs were received in that round for the first time, then take the first one in a lexicographic order.)

- For the receiver P_j: If a message (m, σ), where σ is party i's valid signature on m, is received *for the first time* from some parties in $\Gamma_\rho(j)$ then output m and halt. (If multiple validly signed pairs are received in that round for the first time, then take the first one in a lexicographic order.)

Figure 3.2: Reliable message transmission protocol (adaptive case)

3.4.2 RMT Construction

Here we describe our RMT protocol secure against adaptive adversaries, denoted by $\mathsf{AdRMT}_{i,j}(m)$, in the locality model. The protocol is similar to the protocol in the static case, except that in round ρ parties use the communication graph G_ρ. Let $\Gamma_\rho(i) = \Gamma_{G_\rho}(i)$ denote the set of P_i's neighbors in G_ρ. We describe the protocol in Figure 3.2.

Similar to the static case, the security of protocol $\mathsf{AdRMT}_{i,j}(m)$ follows from the above lemma.

Theorem 3.4.6. *Let $T \subseteq [n]$ be the set of adaptively corrupted parties with $|T| = t \leq qn$ for any constant $0 < q < 1$. Assuming a PKI and an SKI, protocol $\mathsf{RMT}_{i,j}$ is a adaptively secure RMT protocol between any two honest parties P_i, P_j, $i, j \in [n] \setminus T$ satisfying the following two conditions with overwhelming probability:*

- *every party communicates with at most* $\mathcal{O}(\log^{1+\epsilon} n)$ *other parties, for some* $\epsilon > 0$*;*
- *the protocol terminates after* $\mathcal{O}(\log^{\epsilon'} n)$ *rounds, for some* $\epsilon' > 0$.

Proof. First, we may assume that the adaptive adversary corrupts parties at the end of each round. Let U_j be the set of parties corrupted at the end of round j. Then Lemma 3.4.5 shows that any message sent by an honest party P_i will reach any honest P_j within $\mathcal{O}(\log^{\epsilon'}(n))$ rounds, it follows from the unforgeability property of the signature scheme that an honest P_j will always accept the message sent by an honest P_i. Hence, the above protocol is a secure RMT protocol. The communication locality of the protocol follows from the degree of p-random graph G_ρ which is $\mathcal{O}(pn) = \mathcal{O}(d) = \mathcal{O}(\log^{1+\epsilon} n)$, except with negligible probability. Thus, over $\mathcal{O}(\log^{\epsilon'}(n))$ rounds, each party communicates with $\mathcal{O}(\log^{1+\epsilon+\epsilon'}(n))$ other parties. □

3.4.3 Parallel Composition of RMT

Once again, we will require all parties to execute their respective RMT protocols in parallel simultaneously. Let $\mathsf{AdRMT}_{\mathtt{all}}(\vec{m})$ denote the protocol executed by all parties when $\mathsf{AdRMT}_{i,j}(m_{i,j})$ for all $i, j \in [n]$ are executed in parallel. That is, in round ρ of $\mathsf{AdRMT}_{\mathtt{all}}(\vec{m})$, all parties execute round ρ of protocol $\mathsf{AdRMT}_{i,j}(m_{i,j})$ for all $i, j \in [n]$. Note that the graph G_ρ used in round ρ of the protocol depends only on ρ and not on i and j; i.e., we use the same graph G_k to send all messages in round ρ of protocol $\mathsf{AdRMT}_{\mathtt{all}}(\vec{m})$. We have the following corollary:

Corollary 3.4.7. *Under the same assumption as Theorem 3.4.6,* $\mathsf{AdRMT}_{\mathtt{all}}$ *is a adaptively secure RMT protocol for sending* $m_{i,j}$ *from* P_i *to* P_j *for every pair of honest parties* P_i, P_j*,* $i, j \in [n]$*, satisfying the following properties with overwhelming probability:*

1. *every party communicates with at most* $\mathcal{O}(\log^{1+\epsilon} n)$ *other parties, for some* $\epsilon > 0$*;*
2. *the protocol terminates after* $\mathcal{O}(\log^{\epsilon'} n)$ *rounds, for some* $\epsilon' > 0$.

The proof of this corollary is similar to the proof of Corollary 3.3.4 using Theorem 3.4.6 instead of Theorem 3.3.3.

3.5 Secure MPC with Low Communication Locality

In this section, we describe an MPC protocol for securely evaluating any n-party function in the communication locality model. We use a constant-round adaptively secure MPC protocol that secure against $t < \frac{n}{2}$ adaptive corruptions over secure point-to-point channels and broadcast protocol such as in [BMR90]. We show that our secure RMT protocol can be used to construct point-to-point channels and broadcast protocol in the communication locality model.

We let Π_{BC} denote the authenticated broadcast protocol guaranteed by Theorem 3.2.2. The protocol achieves broadcast with overwhelming probability against $t < \frac{n}{2}$ adaptive corruptions, running for $\log^c n$ rounds on a complete network, for some constant $c > 0$. As pointed out in [KK06], assuming unique process and message ID's as in [LLR06], Π_{BC} remains secure under parallel composition.

Let Π^*_{BC} denote the protocol which results by having the parties execute Π_{BC} where in each round instead of using the point-to-point channels for exchanging their messages, the parties invoke $\mathsf{AdRMT}_{\mathtt{all}}$ from Section 3.4.3. Then it follows immediately from the security of $\mathsf{AdRMT}_{\mathtt{all}}$ (Corollary 3.4.7) and the fact that each message transmission requires polylog(n) rounds that protocol Π^*_{BC} is also a secure broadcast protocol with polylogarithmic round complexity and communication locality.

Lemma 3.5.1. *The protocol Π^*_{BC} described above is a broadcast protocol secure against an adaptive adversary corrupting $t < n/2$ parties, and satisfying the following conditions with overwhelming probability:*

1. *each party communicates with at most $\mathcal{O}(\log^{1+\epsilon} n)$ other parties, for some $\epsilon > 0$;*
2. *the protocol terminates after $\mathcal{O}(\log^{\epsilon'} n)$ rounds, for some $\epsilon' > 0$.*

Proof. The security of Π^*_{BC} follows directly from the security of protocols Π_{BC} and $\mathsf{AdRMT}_{\mathtt{all}}$. First, we compute the asymptotic round complexity. For each round ρ of Π_{BC}, the protocol Π^*_{BC} executes $\mathsf{AdRMT}_{\mathtt{all}}$ to have the parties exchange their round ρ messages. Thus, for each round in Π_{BC} we need $\mathcal{O}(\log^{\epsilon'} n)$ rounds in Π^*_{BC}. Because Π_{BC} runs in $\mathcal{O}(\log^{c} n)$ rounds, the total round complexity of Π^*_{BC} is $\mathcal{O}(\log^{c+\epsilon'} n)$ rounds. We compute the communication locality. With overwhelming probability, in each round of Π^*_{BC}, every party might communicate with at most to $\mathcal{O}(\log^{1+\epsilon} n)$ (potentially different) parties for executing $\mathsf{AdRMT}_{\mathtt{all}}$. Thus, since the total number of rounds is $\mathcal{O}(\log^{c+\epsilon'} n)$, then with overwhelming probability (by the union bound) the total number of parties that each $i \in [n]$ exchanges messages with using the point-to-point channels is $\mathcal{O}(\log^{1+c+\epsilon+\epsilon'} n)$. □

The next step is to construct a *secure* message transmission protocol (SMT) which will allow a sender P_i to securely (i.e., authentically and privately) send a message $m_{i,j}$ to a receiver P_j. Since we have a PKI and an adaptively secure broadcast protocol Π^*_{BC}, we can use the standard reduction of secure channels to broadcast: the sender P_i encrypts $m_{i,j}$ under the receiver's public key and broadcasts the corresponding ciphertext $c_{i,j}$. Upon receiving $c_{i,j}$, party P_j decrypts it using its secret key and recovers $m_{i,j}$. However, in order for the above reduction to be secure in a simulation-based manner against an adaptive adversary, we need to ensure that a simulator can "open" a ciphertext to any message of its choice. This can be achieved by the use of a non-committing encryption scheme for computing the ciphertext $c_{i,j}$ [CFG96]. As proved in [DN00], a non-committing encryption scheme can be constructed assuming the existence of families of trapdoor permutations with reversed domain sampler. We denote an adaptively secure SMT above by $\mathsf{AdSMT}_{i,j}$, and the protocol composed of n^2 parallel SMT protocol for all $i, j \in [n]$ by $\mathsf{AdSMT}_{\mathtt{all}}$.

Using the adaptively secure broadcast protocol Π^*_{BC}, the adaptively secure SMT $\mathsf{AdSMT}_{\mathtt{all}}$, and a constant-round MPC protocol over a complete network of point-to-point channels and broadcast which is secure against an adaptive adversary corrupting corrupting $t < \frac{n}{2}$ parties, denoted Π_{MPC}, we now describe an adaptively secure MPC protocol.

Theorem 3.5.2. *Assuming PKI, SKI and trapdoor permutations with reverse domain sampler, there exists an MPC protocol securely computing any given n-party function against an adaptive adversary corrupting $t < n/2$ parties, and satisfying the following conditions with overwhelming probability:*

- *each party communicates with at most $\mathcal{O}(\log^{1+\epsilon} n)$ other parties, for some $\epsilon > 0$;*
- *the protocol terminates after $\mathcal{O}(\log^{\epsilon'} n)$ rounds, for some $\epsilon' > 0$.*

Proof. Let Π_{MPC} denote a constant-round MPC protocol which is secure against adaptive corruptions of up to $t < \frac{n}{2}$ parties, where parties communicate over a complete network of point-to-point channels and broadcast [BMR90]. Furthermore, let Π^*_{MPC} denote the protocol that results by instantiating in Π_{MPC} the calls to the secure channels and broadcast by invocations of protocols AdSMT and Π^*_{BC}, respectively. We argue that Π^*_{MPC} satisfies all the properties claimed in the theorem.

Let $\mathcal{S}$ be the simulator for Π_{MPC}. We construct a simulator $\mathcal{S}^*$ for Π^*_{MPC} based on $\mathcal{S}$ as follows: every time $\mathcal{S}$ sends a message via point-to-point channels or broadcast protocol, $\mathcal{S}$ sends the message via AdSMT or Π^*_{BC}, respectively. This would also require $\mathcal{S}^*$ to simulate the honest parties by forwarding messages as require in $\mathsf{AdRMT}_{\mathtt{all}}$ which is a subprotocol of AdSMT and Π^*_{BC}. Every time that $\mathcal{S}^*$ generates an encryption of message $m_{i,j}$ on behalf of honest P_i to be sent to honest P_j in $\mathsf{AdRMT}_{\mathtt{all}}$ as describe above, $\mathcal{S}^*$ uses non-committing encryption NCSim to instead generate a simulated encryption of $m_{i,j}$ (as it does not know the content). When the adversary $\mathcal{A}$ chooses to corrupt P_i or P_j, $\mathcal{S}$ does so and learns all of its sent and received messages. $\mathcal{S}^*$ then sends σ^b_G, σ^b_E to $\mathcal{A}$ explaining the now opened messages. Thus, the security of Π^*_{MPC} follows from the security of the underlying protocol Π_{MPC} and the security of protocols Π^*_{BC} and $\mathsf{AdSMT}_{\mathtt{all}}$.

We compute the round complexity as follows. For each round in Π_{MPC}, all message exchanges (i.e., point-to-point transmissions or broadcast calls) are exchanged in Π^*_{MPC} by appropriate (parallel) executions of protocols Π^*_{BC} and $\mathsf{AdSMT}_{\mathtt{all}}$, where the executions have unique round, protocol, and message IDs. Thus, for every round in Π_{MPC} we need

$\mathcal{O}(\log^{\epsilon'} n)$ rounds in Π^*_{MPC}, for some given constant $\epsilon' > 0$. Because Π_{MPC} terminates in a constant number of rounds, the round complexity of Π^*_{MPC} is also $\mathcal{O}(\log^{\epsilon'} n)$. In each of these rounds, every party might communicate with at most $\mathcal{O}(\log^{1+\epsilon} n)$ (potentially different) parties, Thus, the total number of parties that each P_i, $i \in [n]$, communicates with is $\mathcal{O}(\log^{1+\epsilon+\epsilon'} n)$. □

3.6 Getting Rid of the SKI

In this section, we show how to get rid of the symmetric-key setup assumption, at the cost of increasing the communication locality (but not the round complexity) by a factor of $\sqrt{n}$.

The idea for getting rid of the SKI is to have the parties compute some kind of an alternative hidden graph setup. This can be done as follows: each party P_i locally decides which of his n point-to-point channels it will use; a channel between two (honest) parties P_i, P_j, $i, j \in [n]$, is then used only if both parties choose it. By having each party decide to use each of its channels with probability $p = \frac{\log^{\delta} n}{\sqrt{n}}$ for some given constant $\delta > \frac{1}{2}$ (and ignore all other channels) we ensure that, with overwhelming probability, each honest party uses at most $\mathcal{O}(\sqrt{n}\log^{\delta} n)$ of its point-to-point channels. Furthermore, each edge between two honest parties P_i and P_j is chosen with probability $p' = p^2 = \frac{\log^{2\delta} n}{n}$, thus the resulting communication graph will include Erdős-Rényi graph $G(n, p')$ which will allow us to use our constructions from the previous sections. Note however, that as the corrupted parties might choose to speak to all their neighbors, the communication locality is no longer guaranteed to be $\mathcal{O}(\log^{2\delta} n)$. Instead, it is guaranteed to be $\mathcal{O}(\sqrt{n}\log^{\delta} n)$ with overwhelming probability.

3.7 Conclusion and Open Questions

We have shown that we can construct an adaptively secure MPC protocol with polylogarithmic communication locality and round complexity tolerating up to $t < \frac{n}{2}$ adaptive corruptions, assuming PKI, SKI and trapdoor permutations with reversed domain sampler,

answering opened questions in [BGT13]. We have also shown that we can get rid of the SKI assumption at the cost of communication locality. So, the natural open question is:

Can we achieve the same communication locality and round complexity under fewer assumptions?

Our result achieves the communication locality of $\mathcal{O}(\log^c n)$ for some constant $c > 1$, which is near optimal. Another open question is:

Can we achieve the optimal round complexity of $\mathcal{O}(\log^{1+\epsilon} n)$ *for* any $\epsilon > 0$*?*

CHAPTER 4

On-the-fly MPC with Circuit Privacy via Circuit-Private MFHE

4.1 Introduction

Multi-key fully homomorphic encryption scheme (MFHE), introduced by López-Alt, Tromer and Vaikuntanathan [LTV12], allows homomorphic computation on inputs encrypted with different public keys. In this chapter, 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 [LTV12] without adding further assumptions.

As in [OPP14], 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, difficulty can arise, as the security needs to hold when the pairs are in the support of honestly generated ones, but with different distribution as discussed in [GHV10].

In [LTV12], 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

subset 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 broadcasted to the clients to cooperate to retrieve the output using (short) MPC protocol. Thus, most computation is done by server while the decryption phase is independent of both the function computed and the total number of parties in the system. Their resulting protocol is a 5-round on-the-fly MPC secure against *semi-malicious* users [AJL12], which follow the protocol but choose 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 chapter, we construct a 3-round on-the-fly MPC with circuit privacy against malicious clients in the plain model. Specifically, all parties send their inputs to the server, who performs the computation and sends his result back to all clients, who then decrypt the result in one round. Since there is no way to enforce which function the server will compute in the plain model, we assume that the server is honest-but-curious. As with our MFHE, the circuit privacy is guaranteed against unbounded malicious adversary corrupting any number of clients. We also note that a variant of circuit privacy can be achieved in [LTV12] 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 clients to server and one broadcast back to clients). After the server sends its output back to the clients, the clients communicate with one another in only 1 additional round in order to decrypt the output.

To summarize, our main theorems are as follows:

Theorem 4.1.1 (informal)**.** *Assuming that there exists a privately expandable multi-hop multi-key compact somewhat homomorphic encryption scheme, then there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

Theorem 4.1.2 (informal)**.** *Assuming RLWE and DSPR assumptions, and circular security, there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

Theorem 4.1.3 (informal)**.** *Assuming the preconditions of Theorem 4.1.1 or Theorem 4.1.2 hold, there exists a 3-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 server.*
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 [OPP14] this is unavoidable, even for maliciously circuit-private single-key FHE.

4.1.1 Previous Work

Multi-key FHE As stated above, [LTV12] 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 [CM15] gives an alternate construction based on [GSW13] 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 [MW15] simplifies the construction of [CM15], and adds threshold decryption protocol, which is used to construct 2-round MPC in the CRS model.

Circuit privacy in FHE In the semi-honest setting, where public keys and ciphertexts are in the support of properly generated pairs, circuit privacy has been considered in [Gen09, VGH10] with the latter using Yao's garbled circuit. The generalization in [GHV10] 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) [AIR01, HK12]. The work of [IP07] 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 [OPP14] 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 [Gen09] and semi-honestly circuit-private HE constructions [AIR01, GHV10] combining FHE schemes with maliciously circuit-private HE schemes.

One-Round OT Several definitions of security of OT have been suggested such as a general framework for defining two-party computation [Can00]. The work of [AIR01] proposes a definition for 1-round (2 messages) OT using unbounded simulation, which implies information theoretic security for sender, and demonstrates a construction based on the DDH assumption. In [IP07], Ishai and Paskin construct a 1-round OT with perfect sender privacy based on the DJ homomorphic encryption scheme [DJ01] 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 computation while the clients compute very little and independently from the circuit. This idea is explored in the work of [KMR11, HLP11]. 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 [AJL12, Gen09] 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 no-the-fly MPC. As mentioned above, López-Alt *et al* [LTV12] constructed on-the-fly MPC from multi-key FHE. However, their version is only secure against semi-malicious adversary 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 universal circuits known have high complexity, namely, $\mathcal{O}(g^5)$ for arithmetic circuits [Raz08] and $\mathcal{O}(g \log g)$ for boolean circuits [Val76] for the class of circuits with at most g gates. This approach also does not hide the size of the circuits evaluated. Previous work [MS13, MSS14] has constructed more efficient implementation of PFEs even against an active adversary [MSS14].

Circuit privacy is useful when learning about the function compromises privacy, reveals security vulnerabilities or intellectual property. Examples of this are in software diagnostic [BPS07], medical applications [BFK09], and intrusion detection systems [NSM13].

4.1.2 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, *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 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 branching program from an adversary, therefore hiding the branching program itself when combining with the technique in [IP07].

We show how to construct a privately expandable multi-key HE scheme from the multi-key somewhat homomorphic encryption scheme defined in [LTV12]. 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 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 scheme as well, such as in [MW15] 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. A input $x \in \{0,1\}^n$ naturally induces a unique path from a distinguished initial nodes to a terminal node, whose label determines $P(x)$. Any logspace or NC function can be computed by a polynomial size branching programs. We inductively compute a ciphertext for each node from terminal nodes upward. Given ciphertext of each bit of $x \in \{0,1\}^n$, encrypted with different public keys, we expand the ciphertexts to hide public keys it originally encrypted with. We use private expandability to homomorphically compute ciphertext at each node with a key-hiding ciphertext indistinguishable from fresh one. Thus, each ciphertext reveals nothing about the path led 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 [OPP14], 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, independently 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 [OPP14] 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 [LTV12] 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 in [LTV12], we consider the plain model with no setup assumptions and malicious adversaries corrupting arbitrary number of clients. Along the way, we also construct a 1-round 1-out-of-2 OT that secure against malicious receiver with information theoretic security by augmenting a known construction that only secure against semi-honest receiver with circuit-private FHE. Finally, by using garbled circuit and our OT protocol, we can reduce the number of rounds from the construction in [LTV12] 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 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 in [IP07, OPP14]. We note that it is impossible to obtain ideal functionality definition due to the impossibility for any computationally bounded simulators

to extract the input in one round (without trusted setup). Instead, we show the security for honest clients based on indistinguishability of the view of the malicious adversaries corrupting clients and the view of honest-but-curious server.

4.2 Background

4.2.1 Multi-key Homomorphic Encryption

We use the definition similar to the one defined in [LTV12] with some modifications from [MW15] and [OPP14]. 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 4.2.1 (Multi-key (Leveled) $(U, \mathcal{C})$-Homomorphic Encryption)**.** *Let $\mathcal{C}$ be a class of circuits. A* multi-key (leveled) $(U, \mathcal{C})$-homomorphic encryption *scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ 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**:* $(\mathsf{KeyGen}, \mathsf{Enc})$ *is semantically secure.*

- ***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.*

4.2.1.1 López-Alt, Tromer and Vaikuntanathan's Multi-key FHE scheme

Our construction modifies the multi-key leveled somewhat HE scheme constructed in [LTV12] by López-Alt, Tromer and Vaikuntanathan. Multi-key leveled fully homomorphic scheme can be obtained by bootstrapping.

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 4.2.2 (Ring Learning With Error (RLWE) Assumption [BV11])**.** *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 4.2.3 (Decisional Small Polynomial Ratio (DSPR) Assumption [LTV12])**.** *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 $\vec{s^i_\gamma}, \vec{e^i_\gamma}, \vec{s^i_\zeta}, \vec{e^i_\zeta} \leftarrow \chi^{\lceil \log q_i \rceil}$.
   (e) Let $\gamma^i := \left[h^i \vec{s^i_\gamma} + 2\vec{e^i_\gamma} + \mathsf{Pow}(f^{i-1})\right]_{q_i} \in R_{q_i}^{\lceil \log q_i \rceil}$
   and $\zeta^i := \left[h^i \vec{s^i_\zeta} + 2\vec{e^i_\zeta} + \mathsf{Pow}\left((f^{i-1})^2\right)\right]_{q_i} \in R_{q_i}^{\lceil \log q_i \rceil}$.
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

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

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

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

   (b) Otherwise, compute

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

   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 $\mu' = \mu_0 \pmod 2$.

Theorem 4.2.4 ([LTV12])**.** *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* $\mathcal{E}_{SH}$*.*

4.2.2 Circuit-Private Homomorphic Scheme

We define multi-key variant of circuit privacy defined in [IP07, OPP14].

Definition 4.2.5. *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* $c_i \leftarrow \mathsf{Enc}(pk_{I_i}, x_i)$*.*

4.2.3 Branching Program

Definition 4.2.6. *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 4.2.7. *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 [Pip79]. For simplicity, we assume all branching programs are layered.

4.2.4 Oblivious Transfer

We use statistical indistinguishability definition for OT instead of real/ideal world definition as simulator for ideal world can break the receiver security for 1-round protocol.

Definition 4.2.8. *A* 1-round 1-out-of-2 OT protocol *is a tuple of PPT algorithms $(G_{\mathsf{OT}}, Q_{\mathsf{OT}}, A_{\mathsf{OT}}, D_{\mathsf{OT}})$ involving two parties, a sender and a receiver. The sender's input is a pair (s_0, s_1) such that $|s_0| = |s_1| = \tau$. The receiver's input is a bit $b \in \{0, 1\}$. The protocol proceeds as follows:*

- *The receiver generates $(pk, sk) \leftarrow G_{\mathsf{OT}}(1^\lambda)$, computes $q \leftarrow Q_{\mathsf{OT}}(1^\lambda, 1^\tau, pk, b)$, and sends (pk, q) to the sender.*
- *The sender computes $a \leftarrow A_{\mathsf{OT}}(s_0, s_1, pk, q)$ and sends a to the receiver.*
- *The receiver compute $D_{\mathsf{OT}}(sk, a)$.*

The protocol is correct if $D_{\mathsf{OT}}(sk, a) = s_b$.

Receiver privacy: $(G_{\mathsf{OT}}, Q_{\mathsf{OT}})$ *is semantically secure.*

Sender privacy (semi-honest case): *There exists an expected polynomial time simulator* Sim_{SH} *such that for any* $b \in \{0,1\}$, $(pk, sk) \leftarrow G_{\mathsf{OT}}(1^\lambda)$, $q \leftarrow Q_{\mathsf{OT}}(1^\lambda, 1^\tau, pk, b)$ *and* $s_0, s_1 \in \{0,1\}^\tau$,

$$A_{\mathsf{OT}}(s_0, s_1, pk, q) \simeq^s \mathsf{Sim}_{SH}(s_b, pk, q).$$

Sender privacy (malicious case): *There exists an unbounded simulator* Sim_M *such that for any* pk^*, q^* *of appropriate length, and* $s_0, s_1 \in \{0,1\}^\tau$, *there exists* $b^* = b(pk^*, q^*)$ *such that*

$$A_{\mathsf{OT}}(s_0, s_1, pk^*, q^*) \simeq^s \mathsf{Sim}_M(s_{b^*}, pk^*, q^*).$$

The protocol has perfect *sender privacy if the distributions are the same.*

We note that the malicious case implies the semi-honest case as we can construct PPT Sim_{SH} from $A_{\mathsf{OT}}(s_b, s_b, pk, q)$ and the indistinguishability follows from the malicious case.

4.2.5 Garbling Scheme

Definition 4.2.9. *A* garbling scheme *is a tuple of PPT algorithms* $(GarbCircuit, GarbEval)$ *such that for any circuit* $C : \{0,1\}^n \to \{0,1\}$ *and* $x \in \{0,1\}^n$, $(G, e) \leftarrow GarbCircuit(1^\lambda, C)$ *and* $X = e(x)$, *we have* $GarbEval(G, X) = C(x)$.

Security: *For any circuits* $C_0, C_1 : \{0,1\}^n \to \{0,1\}$ *and* $x \in \{0,1\}^n$ *such that* $C_0(x) = C_1(x)$, *if, for* $i = 0, 1$, $(G_i, e_i) = GarbCircuit(1^\lambda, C_i)$ *and* $X_i = e_i(x)$, *then* (G_0, X_0) *is computationally indistinguishable from* (G_1, X_1), *i.e. for any PPT adversary* $\mathcal{A}$,

$$|\Pr[\mathcal{A}(G_0, X_0) = 1] - \Pr[\mathcal{A}(G_1, X_1) = 1]| < \mathrm{negl}(\lambda).$$

A garbling scheme is projective *if each bit of the garbled input* $X = e(x)$ *only depends on one bit of* x. *In this case, we may assume that* e *can be represented by* $(X_1^0, X_1^1, \ldots, X_n^0, X_n^1)$ *where* $e(x_1 \ldots x_n) = X_1^{x_1} \ldots X_n^{x_n}$.

4.3 Privately Expandable Multi-key Homomorphic Encryption

In this section, we will define the property of multi-key homomorphic encryption which are required for the construction of multi-key circuit private HE for branching programs in the next section. We also show how to modify the multi-key HE from [LTV12] to achieve such property. We note that the multi-key HE from [MW15] can be modified to have this property in a similar way. However, since it only works in setup model, we do not include it here.

4.3.1 Private Expandability

We define an "expanded" ciphertext as a ciphertext that associates with all public keys to be used in evaluation algorithm. This notion is also used in [MW15]. However, expanded ciphertext in [MW15] do not hide the original public key is encrypted with. In both our construction and the one in [MW15], 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 while in [MW15] case, it is the appending of all secret keys.

Definition 4.3.1. *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 (KeyGen,

$\mathsf{Enc}, \mathsf{Expand}, \mathsf{Eval}, \mathsf{Dec}$) a privately expandable HE scheme if Expand, Eval and Dec satisfy the above conditions.

4.3.2 Privately Expandable Multi-key HE based on LTV Encryption Scheme

In [LTV12], 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 be privately expandable by constructing $\widetilde{\mathsf{Expand}}, \widetilde{\mathsf{Eval}}, \widetilde{\mathsf{Dec}}$ without extra 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 [LTV12] defined in the previous section.

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 bootstrappable somewhat homomorphic version instead of 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 $\mathcal{O}(\log t)$. 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 [AJL12]:

Lemma 4.3.2. *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 $|a_1 - a_2| \leq 2B$. Thus, $\Delta(a_1 + \mathcal{U}_t, a_2 + \mathcal{U}_t) \leq 4B/t$. 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 4.3.3. *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* $t = O(\frac{q}{N(nB)^N})$. *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 4.3.2. 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 $E := 2nBt + 2nB(2t + 1) = 2nB(3t + 1) = O(nBt)$. Then for $f = f_1 \ldots f_N$,

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

is bounded by $(nB)^N + N(nB)^{N-1}E = O(N(nB)^N t)$, which can be decrypted if it is less than $q/2$. Thus, for $t = O(\frac{q}{N(nB)^N})$, 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 4.3.4 (implied from [LTV12]). *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.*

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 $pk = (pk_0, pk_{\mathcal{E}}, f_s k)$ and $sk = sk_{\mathcal{E}}$.

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

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

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

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

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

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

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

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 4.3.5. *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 privately expandable multi-key compact somewhat homomorphic scheme that can evaluate circuit up to depth* $2d_0 + 2$.

Proof. The security and compactness of $\mathcal{F}$ follows directly from that of $\mathcal{E}$. By Lemma 4.3.3, 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 4.3.3, $\hat{c} \simeq^s \widetilde{\mathsf{Expand}}^t(pk_1, \ldots, pk_N, j, \mathsf{Enc}_{\mathcal{E}}(pk_j, \mu))$. 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 circuit up to depth $2d_0 + 2$ as required. □

4.4 Circuit-Private Multi-key HE for Branching Programs

In this section, we construct a circuit-private multi-key HE for a class 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 depended on its corresponding input bit, which in turn depended on which public key it is encrypted with. Using such encryption may reveal bit position of the path it takes to reach a terminal node. Using privately expandable multi-key HE scheme from the 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 secure against semi-honest adversary where each pair of public key and ciphertext is correctly generated. We show that, in this case, the output can be simulated knowing the public keys, ciphertext and the output, thus independent of the program being evaluated. We then show that we can augment it with a single-key circuit-private FHE as the evaluated output do not depend on the branching program unlike in the general case.

4.4.1 Semi-Honest Case

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 $\mathsf{Dec}^2_{\mathcal{F}}(sk_1, \ldots, sk_N, c) = \mathsf{Dec}_{\mathcal{F}}(sk_1, \ldots, sk_N, \mathsf{Dec}_{\mathcal{F}}(sk_1, \ldots, sk_N, c))$. We describe $\mathcal{E}_S = (\mathsf{KeyGen}_S, \mathsf{Enc}_S, \mathsf{Eval}_S, \mathsf{Dec}_S)$ together with Expand and $\widetilde{\mathsf{Enc}}$.

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

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. For $j = 1, \ldots, N$, parse $pk_j = (pk_{\mathcal{F},j}, f_{sk,j})$.
2. Let $i \leftarrow [N]$ and compute $c \leftarrow \mathsf{Enc}(pk_i, \mu)$.
3. Output $\tilde{c} = \mathsf{Expand}_{\mathcal{F}}(pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, i, c)$.

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

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) $a_v = a_1 \dots a_s$; if $h = 1$, $label(v) \leftarrow a_v$

(c) otherwise, $label(v) \leftarrow \mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, pk_{\mathcal{F},1}, \dots, pk_{\mathcal{F},N}, \tilde{f}_{sk,1}, \dots, \tilde{f}_{sk,N}, a_v)$

6. Output $\tilde{c} = label(root)$

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

1. Parse $sk_i = sk_{\mathcal{F},i}$.

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

4.4.2 Correctness and Security against Semi-Honest Adversaries

The correctness is a direct result of the following lemma:

Lemma 4.4.1. *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 encryption

of 1 and encryptions 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 encryption of 1 and encryption of 0 in the second case and the last case. All of which are correct with respect to r_0, r_1. Thus, $\mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \ldots, sk_{\mathcal{F},N}, a_v) = label(u_{x_i})$.

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

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

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

□

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

Lemma 4.4.2. *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 program.*

Proof. We construct a simulator Sim_S as follows:

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

1. For $i = 1, \ldots, N$, parse $pk_i = (pk_{\mathcal{F},i}, f_{sk,i})$.
2. Let $out_0 = b$.
3. 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 $\mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, pk_{\mathcal{F},1}, \ldots, pk_{\mathcal{F},N}, \tilde{f}_{sk,1}, \ldots, \tilde{f}_{sk,N}, out_h)$

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

4.4.3 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 (such as one constructed in [OPP14]) to homomorphically check 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 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 $r_0 = label(u_0)[t]$ 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 $\hat{c} = (label(root) \oplus (S_1 \oplus \ldots \oplus S_N), v_1, \ldots, v_N)$

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

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 $\mu' := \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F},1}, \ldots, sk_{\mathcal{F},N}, \tilde{c}')$

4.4.4 Security against Malicious Adversary

Theorem 4.4.4. *Assuming $\mathcal{F}$ is 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 branching program.*

Proof. Let $\mathsf{Ext}_{\mathcal{P}}$ and $\mathsf{Sim}_{\mathcal{P}}$ be the extractor and the simulator guaranteed by circuit privacy of $\mathcal{P}$. We construct a pair of algorithms Ext and Sim 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 $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) \neq \mathsf{KeyGen}_{\mathcal{F}}(params^*_{\mathcal{F}}; r^*_e)$, 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 $\mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}^2_{\mathcal{F}}, pk^*_{\mathcal{F},1}, \ldots, pk^*_{\mathcal{F},N}, \tilde{f}^*_{sk,1}, \ldots, \tilde{f}^*_{sk,N}, out_h)$

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 4.2.5.

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.

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_S, the result follows from Lemma 4.4.2. □

Combining the above result with Lemma 4.3.5 results in the following theorem:

Theorem 4.4.5. *Let $\mathcal{F}$ be a privately expandable multi-hop multi-key compact somewhat homomorphic encryption scheme that can evaluate 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 program.*

Corollary 4.4.6. *Assuming RLWE and DSPR assumptions, and circular security for* $\mathcal{E}$*, there exists a maliciously circuit-private multi-key HE scheme for branching program.*

4.5 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 [OPP14]. As we discuss 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 circuit-private scheme to verify the input. Finally, we switch it back to the original scheme for compactness. Unlike in single-key case, we cannot verify all public keys and ciphertexts at once as it would lead to larger verification circuit. We rely on 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

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

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

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

4.5.1 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 4.5.1. *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.*

Proof. Correctness, semantic security and compactness follow from scheme $\mathcal{F}$ and $\mathcal{P}$. Let $\mathsf{Ext}_{\mathcal{P}}$ and $\mathsf{Sim}_{\mathcal{P}}$ be as in Definition 4.2.5 for circuit private scheme $\mathcal{P}$. We describe a pair of algorithms

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

1. Parse $pk^* = (pk^*_{\mathcal{P}}, pk^*_{\mathcal{F}}, p^*_{sk_{\mathcal{F}}}, p^*_{r_{\mathcal{F}K}}, f^*_{sk_{\mathcal{P}}})$ and $c^* = (c^*_{\mathcal{F}}, p^*_{r_{\mathcal{F}E}})$.

2. Let $sk^*_{\mathcal{F}} = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk^*_{\mathcal{P}}, p^*_{sk_{\mathcal{F}}})$, $r^*_{\mathcal{F}K} = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk^*_{\mathcal{P}}, p^*_{r_{\mathcal{F}K}})$ and $r^*_{\mathcal{F}E} = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk^*_{\mathcal{P}}, p^*_{r_{\mathcal{F}E}})$.

3. if $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) \neq \mathsf{KeyGen}_{\mathcal{F}}(1^\lambda; r^*_{\mathcal{F}K})$, output 0.

4. if $c^*_{\mathcal{F}} = \mathsf{Enc}_{\mathcal{F}}(pk^*_{\mathcal{F}}, b; r^*_{\mathcal{F}E})$ for some $b \in \{0, 1\}$, output b; otherwise, output 0.

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

1. For $i = 1, \ldots, N$,
   (a) Parse $pk_i^* = (pk_{\mathcal{P},i}^*, pk_{\mathcal{F},i}^*, p_{sk_{\mathcal{F}},i}^*, p_{r_{\mathcal{F}K},i}^*, f_{sk_{\mathcal{P}},i}^*)$.
   (b) Let $sk_{\mathcal{F},i}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},i}^*, p_{sk_{\mathcal{F}},i}^*)$ and $r_{\mathcal{F}K,i}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},i}^*, p_{r_{\mathcal{F}K},i}^*)$.
2. Let $out_{\mathcal{P}}^* = \mathsf{Sim}_{\mathcal{P}}(1^\lambda, (pk_{\mathcal{P},1}^*, \ldots, pk_{\mathcal{P},N}^*), (1, p_{sk_{\mathcal{F}},1}^*), \ldots, (N, p_{sk_{\mathcal{F}},N}^*), b)$.
3. For $i = 1, \ldots, n$,
   (a) Parse $c_i^* = (c_{\mathcal{F},i}^*, p_{r_{\mathcal{F}E},i}^*)$.
   (b) Let $r_{\mathcal{F}E,i}^* = \mathsf{Ext}_{\mathcal{P}}(1^\lambda, pk_{\mathcal{P},i}^*, p_{r_{\mathcal{F}E},i}^*)$.
   (c) If $c_{\mathcal{F},i}^* \neq \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F},I_i}^*, b_i; r_{\mathcal{F}E,i}^*)$ for any $b_i \in \{0,1\}$, let $out_{C,i}^* = \mathsf{Sim}_{\mathcal{P}}(1^\lambda, (pk_{\mathcal{P},1}^*, \ldots, pk_{\mathcal{P},N}^*), (I_i, p_{r_{\mathcal{F}E},i}^*), 0)$.
   (d) Otherwise, let $out_{C,i}^* = \mathsf{Sim}_{\mathcal{P}}(1^\lambda, (pk_{\mathcal{P},1}^*, \ldots, pk_{\mathcal{P},N}^*), (I_i, p_{r_{\mathcal{F}E},i}^*), out_{\mathcal{P}}^*)$.
4. For $i = 1, \ldots, N$,
   (a) If $(pk_{\mathcal{F},i}^*, sk_{\mathcal{F},i}^*) \neq \mathsf{KeyGen}_{\mathcal{F}}(1^\lambda; r_{\mathcal{F}K,i}^*)$, let $out_{K,i}^* = \mathsf{Sim}_{\mathcal{P}}(1^\lambda, (pk_{\mathcal{P},1}^*, \ldots, pk_{\mathcal{P},N}^*), (i, p_{sk_{\mathcal{F}},i}^*), (i, p_{r_{\mathcal{F}K},i}^*), 0)$.
   (b) Otherwise, let $out_{K,i}^* = \mathsf{Sim}_{\mathcal{P}}(1^\lambda, (pk_{\mathcal{P},1}^*, \ldots, pk_{\mathcal{P},N}^*), (i, p_{sk_{\mathcal{F}},i}^*), (i, p_{r_{\mathcal{F}K},i}^*), out_{\mathcal{P}}^*)$.
5. Output $\hat{c}^* = \mathsf{Eval}_{\mathcal{F}}(\mathsf{Dec}_{\mathcal{P}}(\cdot, \mathsf{CombineDec}(\cdot, out_{K,1}^*, \ldots, out_{K,N}^*, out_{C,1}^*, \ldots, out_{C,n}^*)), (pk_{\mathcal{F},1}^*, \ldots, pk_{\mathcal{F},N}^*), (1, f_{sk_{\mathcal{P}},1}^*), \ldots, (N, f_{sk_{\mathcal{P}},N}^*))$.

For each $i = 1, \ldots, N$, if $V_{K,i} = \mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F},i}^*, out_{\mathcal{P}}^*}(sk_{\mathcal{F},i}^*, r_{\mathcal{F}K,i}) = 0$, then the test in step 4 of Sim fail. By circuit privacy of $\mathcal{P}$, $out_{K,i} \simeq^s out_{K,i}^*$. Otherwise, $V_{K,i} = out_{\mathcal{P}}^*$. Then the test in step 3 passes. Thus, by circuit privacy of $\mathcal{P}$, $out_{\mathcal{P}} \simeq^s out_{\mathcal{P}}^*$. Then $out_{K,i} \simeq^s \mathsf{Eval}_{\mathcal{P}}(\mathsf{KValidate}^{\lambda,d}_{pk_{\mathcal{F},i}^*, out_{\mathcal{P}}^*}, (i, p_{sk_{\mathcal{F}},i}^*), (i, p_{r_{\mathcal{F}K},i}^*)) \simeq^s out_{K,i}^*$.

Similar argument can be made for $out_{C,i} \simeq^s out_{C,i}^*$ for each $i = 1, \ldots, n$. Therefore $\hat{c} \simeq^s \hat{c}^*$ as the last step of Sim is the same as the last step of Eval. □

4.5.2 Instantiation

Finally, we instantiate the result of Theorem 4.5.1 by our construction in Theorem 4.4.5, we get the following results:

Corollary 4.5.2. *Assuming there exists a privately expandable multi-hop multi-key compact somewhat homomorphic encryption scheme that can evaluate circuit 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 4.5.3. *Assuming RLWE and DSPR assumptions, and circular security for* $\mathcal{E}$*, there exists a maliciously circuit-private multi-key fully homomorphic encryption scheme.*

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

In this section, we consider one application of circuit-private MFHE scheme: on-the-fly MPC protocol. In this setting, 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 [MW15] as follows:

Definition 4.6.1. *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 $P_1, \ldots, P_U$ is 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 security and circuit privacy of on-the-fly MPC protocol in plain model against malicious adversary.

Definition 4.6.2. *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 $\mathsf{View}_{\Pi,S}(F, V, \vec{x})$ 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 $\vec{x}$. Let $\mathsf{View}_{\Pi,\mathcal{A}}(F, V, \vec{x})$ denote the joint collection of messages $\mathcal{A}$ receives through corrupted parties in an execution of protocol Π on V, F and $\vec{x}$.

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 $\vec{x}, \vec{x}'$ such that $x_i = x'_i$ for any $i \in T$,*

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

- *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 $\vec{x}, \vec{x}'$,*

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

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 $\mathcal{F}$, who 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 $\vec{x}$, $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}}(F, V, x) \simeq^s \mathsf{REAL}_{\Pi,\mathcal{A}}(F, V, x)$.*

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 adversary without using setup. This is because the evaluated output only depend 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 possible, to verify against malicious server. Therefore, we assume that the server is only honest-but-curious, who follows the protocol, but may try to learn clients' input data.

Naturally, MFHE scheme leads to server-assisted MPC by having each client generates keys, encrypts its input 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 [LTV12], 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 projective garbling scheme.

After the server runs 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 1-out-of-2 oblivious transfer (OT). In order to minimize the round complexity of our MPC protocol, we would like to use one-round variant of OT. However, the standard 1-round 1-out-of-2 OT protocols known are only secure against semi-honest receiver.

Before we describe the protocol, we first show that we can use a circuit-private single-key FHE to construct a 1-round 1-out-of-2 OT protocol that secure against malicious receiver. This protocol serves as a building block we will use in the construction of 3-round on-the-fly MPC protocol.

4.6.1 1-round 1-out-of-2 OT against Malicious Receiver

Let $\mathcal{F} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Enc}_{\mathcal{F}}, \mathsf{Eval}_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$ be maliciously circuit-private (single-key) FHE. We define a circuit $C_{b_0,b_1}(x) = b_x$. We define $\mathsf{OT}_M = (G_{\mathsf{OT}}, Q_{\mathsf{OT}}, A_{\mathsf{OT}}, D_{\mathsf{OT}})$ as follows:

$G_{\mathsf{OT}}(1^\lambda)$**:**

1. Let $(pk_{\mathcal{F}}, sk_{\mathcal{F}}) \leftarrow \mathsf{KeyGen}_{\mathcal{F}}(1^\lambda)$.
2. Output $pk = pk_{\mathcal{F}}$ and $sk = sk_{\mathcal{F}}$.

$Q_{\mathsf{OT}}(pk, b)$**:**

1. Output $q = \mathsf{Enc}_{\mathcal{F}}(pk, b)$.

$A_{\mathsf{OT}}(s_0, s_1, pk, q)$**:**

1. For $i = 1, \ldots, \tau$, let $a_i = \mathsf{Eval}_{\mathcal{F}}(C_{s_0[i],s_1[i]}, pk, q)$.
2. Output $a = (a_1, \ldots, a_\tau)$.

$D_{\mathsf{OT}}(sk, a)$**:**

1. Parse $a = (a_1, \ldots, a_\tau)$.
2. For $i = 1, \ldots, \tau$, let $s[i] = \mathsf{Dec}_{\mathcal{F}}(sk, a_i)$.
3. Output $s = (s[1], \ldots, s[\tau])$.

Lemma 4.6.3. *There exists an unbounded algorithm* Sim *such that for any* $s_0, s_1 \in \{0,1\}^\tau$*,* pk^*, q^* *of appropriate length, there exists* $b^* = b(pk^*, q^*) \in \{0,1\}$ *such that*

$$\mathsf{Sim}(s_{b^*}, pk^*, q^*) \simeq^s A_{\mathsf{OT}}(s_0, s_1, pk^*, q^*).$$

Proof. Let $\mathsf{Ext}_{\mathcal{F}}$, $\mathsf{Sim}_{\mathcal{F}}$ be the extractor and the simulator guaranteed by circuit privacy. We define Sim as follows:

$\mathsf{Sim}(s, pk^*, q^*)$**:**

1. Let $b^* = \mathsf{Ext}_{\mathcal{F}}(pk^*_{\mathcal{F}}, q^*)$.
2. For $i = 1, \ldots, \tau$, let $a^*_i = \mathsf{Ext}_{\mathcal{F}}(pk^*, q^*, s_{b^*[i]})$.
3. Output $a^* = (a^*_1, \ldots, a^*_\tau)$.

For each $i = 1, \ldots, \tau$, we have $a^*_i \simeq^s \mathsf{Eval}_{\mathcal{F}}(C_{s_0[i], s_1[i]}, pk^*, q^*)$ by circuit privacy of $\mathcal{F}$. Note that for $i = 1, \ldots, \tau$, for fixed pk^*, q^*, the distributions on each side are independent (only depend on randomness used for $\mathsf{Eval}_{\mathcal{F}}$ and $\mathsf{Ext}_{\mathcal{F}}$). Thus, the joint distributions $a^* \simeq^s A_{\mathsf{OT}}(s_0, s_1, pk^*, q^*)$. □

Theorem 4.6.4. *Assuming a circuit-private single-key FHE, there exists a one-round 1-out-of-2 oblivious transfer protocol that secure against malicious receiver.*

4.6.2 Construction of On-the-fly MPC

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 1-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 $y = \mathsf{GarbEval}(G, X_0, \ldots, X_{N(r+s)-1})$.

Figure 4.1: On-the-fly multi-party computation protocol with circuit privacy

Remark

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 López-Alt *et al.*'s multi-key homomorphic encryption scheme that we use to construct MFHE. It is also the case in [LTV12] on-the-fly MPC construction.

2. Private channel (from the server) between clients is required to prevent the server learning clients' secret keys. This requirement can be done by 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 encrypted 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 assumption by using OT constructed from the same circuit-private homomorphic scheme in section 4.

Theorem 4.6.5. *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 $Gb = (GarbCircuit, Garb\text{-}Eval)$ be a projective garbling scheme. If $\mathcal{E}$ is maliciously circuit-private, OT is secure against malicious receiver, and Gb is secure garbling scheme, then the protocol Π_N is a 3-round secure on-the-fly MPC protocol with circuit privacy.*

Proof. First, we will show the privacy for honest clients against malicious adversaries corrupting $t < N$ clients. Let $T \subsetneq [N]$ be the set of corrupted clients.

Lemma 4.6.6. *For any $y \in \{0,1\}$, $x, x' \in \{0,1\}^N$, $F : \{0,1\}^N \to \{0,1\}$ and $T \subsetneq [N]$ such that $x_i = x'_i$ for any $i \in T$,*

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

Proof. Let G and G' be garbled circuits $\mathcal{A}$ receives in step 2 with input x and x', respectively. Note also that by the sender security of OT against malicious adversary, $\mathcal{A}$ can

receive at most one garbled input for each client it controls. Since both circuits evaluate to y on garbled inputs corresponding to valid secret keys and 0 otherwise, $(G, X) \simeq^c (G', X')$. Since a_i and garbled inputs do not depend on x, x', $[\mathsf{View}_{\Pi,\mathcal{A}}(F, x)|y = F(x)] \simeq^c [\mathsf{View}_{\Pi,\mathcal{A}}(F, x')|y = F(x')]$. □

Now we show the privacy for clients against honest-but-curious server. By the sender security of OT, we may replace each bit of secret keys and random coins in queries $\widetilde{\vec{q}}_i$ by a random bit, assuming the circular security. Then by the security of $\mathcal{E}$, we may replace clients' input in $\tilde{c}_i$ by random bits. Thus, the honest-but-curious server learns nothing about $\{x_i\}_{i\in[U]}$. Since this indistinguishability argument can be one separately for each client, the security for honest clients can be achieved regardless of corrupted clients' messages.

Lastly, we will show the privacy for the server against unbounded adversaries. Since we do not guarantee client privacy here, we may assume w.l.o.g. that the adversary corrupts all clients in V. Let $T = V = [N]$ be the set of corrupted clients.

Let Ext and Sim be the extractor and the simulator in Definition 4.2.5. We construct an unbounded simulator $\mathcal{S}_s$ as follows:

Step 1: The simulator receives $\{(\tilde{pk}_i, \tilde{c}_i, \tilde{pk}_{\mathsf{OT},i}, \widetilde{\vec{q}}_i)\}_{i\in T}$ from $\mathcal{A}_s$, and runs Ext for $\mathcal{E}$ to compute corrupted input $\tilde{x} = \mathsf{Ext}(1^\lambda, \tilde{pk}_i, \tilde{c}_i)$. It then submits to the ideal functionality $\mathcal{F}$ to obtain $b = F(\tilde{x}_1, \ldots, \tilde{x}_N)$.

Step 2: For $i \notin T$, the simulator generates a key pair $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(1^\lambda)$. The simulator runs Sim for $\mathcal{E}$ to compute $\tilde{c} = \mathsf{Sim}(1^\lambda, (\tilde{pk}_1, \ldots, \tilde{pk}_N), (1, \tilde{c}_1), \ldots, (N, \tilde{c}_N), b)$. It then computes a garbled circuit $(G', e') = \mathit{GarbCircuit}(1^\lambda, g_{\tilde{c},\tilde{pk}_1,\ldots,\tilde{pk}_N})$ where

$$g_{\tilde{c},\tilde{pk}_1,\ldots,\tilde{pk}_N}((sk_1, r_1), \ldots, (sk_N, r_N)) = \begin{cases} \mathsf{Dec}(sk_1, \ldots, sk_N, \tilde{c}) & \text{if } (\tilde{pk}_i, sk_i) = \\ & \mathsf{KeyGen}(1^\lambda; r_i) \\ & \text{for all } i \in [N]; \\ \bot & \text{otherwise} \end{cases}$$

and $e' = (X_0^0, X_0^1, \ldots, X_{N(r+s)-1}^0, X_{N(r+s)-1}^1)$. For each $i \in [N]$ and $j = 0, \ldots, r+s-1$, it computes $a_i^j = A_{\mathsf{OT}}(pk_{\mathsf{OT},i}, q_i^j, X_{i(r+s)+j}^0, X_{i(r+s)+j}^1)$. It sends $(G', a_i^0, \ldots, a_i^{r+s-1})$ (and V) to $\mathcal{A}_s$ for each $i \in T$.

Step 3: The simulator receives garbled inputs $X_{i(r+s)+j}$ for $j = 0, \ldots, r+s-1$ and $i \in T$ from $\mathcal{A}$ and retrieves $(\tilde{sk}_i, \tilde{r}_i)$ using e'. It then verifies if $(\tilde{pk}_i, \tilde{sk}_i) = \mathsf{KeyGen}(1^\lambda; \tilde{r}_i))$ for all $i \in T$. If not, it outputs $\perp$ and aborts.

Output: The simulator receives the output of the corrupted parties from $\mathcal{A}_s$ and returns it as its output.

By Definition 4.2.5, $\tilde{c} \simeq^s c$. Thus, $G' \simeq^s G$ and $e' \simeq^s e$. We have $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S}_s}(F, V, x) \simeq^s \mathsf{REAL}_{\Pi,\mathcal{A}_s}(F, V, x)$ by circuit privacy of $\mathcal{E}$. □

4.7 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 López-Alt *et al.*'s construction of MFHE 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 plain model?

Since our technique only relies on properties that exist in many single-key construction, 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 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

[AIR01] Bill Aiello, Yuval Ishai, and Omer Reingold. "Priced oblivious transfer: How to sell digital goods." In *Advances in CryptologyEUROCRYPT 2001*, pp. 119–135. Springer, 2001.

[AJL12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. "Multiparty computation with low communication, computation and interaction via threshold FHE." In *Advances in Cryptology–EUROCRYPT 2012*, pp. 483–501. Springer, 2012.

[BFK09] Mauro Barni, Pierluigi Failla, Vladimir Kolesnikov, Riccardo Lazzeretti, Ahmad-Reza Sadeghi, and Thomas Schneider. "Secure evaluation of private linear branching programs with medical applications." In *Computer Security–ESORICS 2009*, pp. 424–439. Springer, 2009.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. "Non-interactive zero-knowledge and its applications." In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pp. 103–112. ACM, 1988.

[BGT13] Elette Boyle, Shafi Goldwasser, and Stefano Tessaro. "Communication locality in secure multi-party computation." In *Theory of Cryptography*, pp. 356–376. Springer, 2013.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping." In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pp. 309–325. ACM, 2012.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. "Completeness theorems for non-cryptographic fault-tolerant distributed computation." In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pp. 1–10. ACM, 1988.

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. "The round complexity of secure protocols." In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pp. 503–513. ACM, 1990.

[BPS07] Justin Brickell, Donald E Porter, Vitaly Shmatikov, and Emmett Witchel. "Privacy-preserving remote diagnostics." In *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 498–507. ACM, 2007.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. "Fully homomorphic encryption from ring-LWE and security for key dependent messages." In *Advances in Cryptology–CRYPTO 2011*, pp. 505–524. Springer, 2011.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. "Efficient fully homomorphic encryption from (standard) LWE." *SIAM Journal on Computing*, **43**(2):831–871, 2014.

[Can00] Ran Canetti. "Security and composition of multiparty cryptographic protocols." *Journal of CRYPTOLOGY*, **13**(1):143–202, 2000.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgard. "Multiparty unconditionally secure protocols." In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pp. 11–19. ACM, 1988.

[CFG96] Ran Canetti, Uri Feige, Oded Goldreich, and Moni Naor. "Adaptively Secure Multi-party Computation." In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pp. 639–648, New York, NY, USA, 1996. ACM.

[CGO10] Nishanth Chandran, Juan Garay, and Rafail Ostrovsky. "Improved fault tolerance and secure computation on sparse networks." In *Automata, Languages and Programming*, pp. 249–260. Springer, 2010.

[CGO12] Nishanth Chandran, Juan Garay, and Rafail Ostrovsky. "Edge fault tolerance on sparse networks." In *Automata, Languages, and Programming*, pp. 452–463. Springer, 2012.

[Cle86] Richard Cleve. "Limits on the security of coin flips when half the processors are faulty." In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 364–369. ACM, 1986.

[CM15] Michael Clear and Ciarán McGoldrick. "Multi-identity and multi-key leveled FHE from learning with errors." In *Advances in Cryptology–CRYPTO 2015*, pp. 630–656. Springer, 2015.

[DJ01] Ivan Damgård and Mats Jurik. "A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System." In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, PKC '01, pp. 119–136, London, UK, UK, 2001. Springer-Verlag.

[DN00] Ivan Damgård and Jesper Buus Nielsen. "Improved non-committing encryption schemes based on a general complexity assumption." In *Advances in Cryptology-Crypto 2000*, pp. 432–450. Springer, 2000.

[DPP88] Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. "Fault tolerance in networks of bounded degree." *SIAM Journal on Computing*, **17**(5):975–988, 1988.

[DS83] Danny Dolev and H. Raymond Strong. "Authenticated algorithms for Byzantine agreement." *SIAM Journal on Computing*, **12**(4):656–666, 1983.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. "Multiple noninteractive zero knowledge proofs under general assumptions." *SIAM Journal on Computing*, **29**(1):1–28, 1999.

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. "i-hop homomorphic encryption and rerandomizable Yao circuits." In *Advances in Cryptology–CRYPTO 2010*, pp. 155–172. Springer, 2010.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to play any mental game." In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pp. 218–229. ACM, 1987.

[GO08] Juan A Garay and Rafail Ostrovsky. "Almost-everywhere secure computation." In *Advances in Cryptology–EUROCRYPT 2008*, pp. 307–323. Springer, 2008.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based." In *Advances in Cryptology–CRYPTO 2013*, pp. 75–92. Springer, 2013.

[HK12] Shai Halevi and Yael Tauman Kalai. "Smooth projective hashing and two-message oblivious transfer." *Journal of Cryptology*, **25**(1):158–193, 2012.

[HLP11] Shai Halevi, Yehuda Lindell, and Benny Pinkas. "Secure Computation on the Web: Computing Without Simultaneous Interaction." In *Proceedings of the 31st Annual Conference on Advances in Cryptology*, CRYPTO'11, pp. 132–150, Berlin, Heidelberg, 2011. Springer-Verlag.

[Hoe63] Wassily Hoeffding. "Probability inequalities for sums of bounded random variables." *Journal of the American statistical association*, **58**(301):13–30, 1963.

[IP07] Yuval Ishai and Anat Paskin. "Evaluating Branching Programs on Encrypted Data." In *Proceedings of the 4th Conference on Theory of Cryptography*, TCC'07, pp. 575–594, Berlin, Heidelberg, 2007. Springer-Verlag.

[KK06] Jonathan Katz and Chiu-Yuen Koo. "On expected constant-round protocols for Byzantine agreement." In *Advances in Cryptology-CRYPTO 2006*, pp. 445–462. Springer, 2006.

[KMR11] Seny Kamara, Payman Mohassel, and Mariana Raykova. "Outsourcing Multi-Party Computation." *IACR Cryptology ePrint Archive*, **2011**:272, 2011.

[KSS06a] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. "Scalable leader election." In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pp. 990–999. Society for Industrial and Applied Mathematics, 2006.

[KSS06b] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. "Towards secure and scalable computation in peer-to-peer networks." In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pp. 87–98. IEEE, 2006.

[LLR06] Yehuda Lindell, Anna Lysyanskaya, and Tal Rabin. "On the composition of authenticated byzantine agreement." *Journal of the ACM (JACM)*, **53**(6):881–917, 2006.

[LOS06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. "Sequential aggregate signatures and multisignatures without random oracles." In *Advances in Cryptology-EUROCRYPT 2006*, pp. 465–485. Springer, 2006.

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. "The Byzantine generals problem." *ACM Transactions on Programming Languages and Systems (TOPLAS)*, **4**(3):382–401, 1982.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. "On-the-fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption." In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pp. 1219–1234, New York, NY, USA, 2012. ACM.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. "Accountable-subgroup multisignatures." In *Proceedings of the 8th ACM conference on Computer and Communications Security*, pp. 245–254. ACM, 2001.

[MS13] Payman Mohassel and Saeed Sadeghian. "How to hide circuits in MPC an efficient framework for private function evaluation." In *Advances in Cryptology–EUROCRYPT 2013*, pp. 557–574. Springer, 2013.

[MSS14] Payman Mohassel, Saeed Sadeghian, and Nigel P Smart. "Actively secure private function evaluation." In *Advances in Cryptology–ASIACRYPT 2014*, pp. 486–505. Springer, 2014.

[MW15] Pratyay Mukherjee and Daniel Wichs. "Two Round Mutliparty Computation via Multi-Key FHE." Cryptology ePrint Archive, Report 2015/345, 2015. http://eprint.iacr.org/.

[NSM13] Salman Niksefat, Babak Sadeghiyan, Payman Mohassel, and Saeed Sadeghian. "Zids: A privacy-preserving intrusion detection system using secure two-party computation protocols." *The Computer Journal*, p. bxt019, 2013.

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. "Maliciously circuit-private FHE." In *Advances in Cryptology–CRYPTO 2014*, pp. 536–553. Springer, 2014.

[Pip79] Nicholas Pippenger. "On Simultaneous Resource Bounds." In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science*, SFCS '79, pp. 307–311, Washington, DC, USA, 1979. IEEE Computer Society.

[PSL80] Marshall Pease, Robert Shostak, and Leslie Lamport. "Reaching agreement in the presence of faults." *Journal of the ACM (JACM)*, **27**(2):228–234, 1980.

[Raz08] Ran Raz. "Elusive Functions and Lower Bounds for Arithmetic Circuits." In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pp. 711–720, New York, NY, USA, 2008. ACM.

[Upf94] Eli Upfal. "Tolerating a linear number of faults in networks of bounded degree." *Information and Computation*, **115**(2):312–320, 1994.

[Val76] Leslie G. Valiant. "Universal Circuits (Preliminary Report)." In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, STOC '76, pp. 196–203, New York, NY, USA, 1976. ACM.

[VGH10] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. "Fully homomorphic encryption over the integers." In *Advances in cryptology–EUROCRYPT 2010*, pp. 24–43. Springer, 2010.

[Yao86] Andrew Yao. "How to generate and exchange secrets." In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pp. 162–167. IEEE, 1986.