Maliciously Circuit-Private FHE

Rafail Ostrovsky[1], Anat Paskin-Cherniavsky[2], and Beni Paskin-Cherniavsky[3]

[1] Department of Computer Science and Mathematics, UCLA, rafail@cs.ucla.edu*
[2] Department of Computer Science, UCLA, anpc@cs.ucla.edu**
[3] cben@users.sf.net

Abstract. We present a framework for transforming FHE (fully homomorphic encryption) schemes with no circuit privacy requirements into maliciously circuit-private FHE. That is, even if both maliciously formed public key and ciphertext are used, encrypted outputs only reveal the evaluation of the circuit on some well-formed input x^*. Previous literature on FHE only considered semi-honest circuit privacy. Circuit-private FHE schemes have direct applications to computing on encrypted data. In that setting, one party (a receiver) holding an input x wishes to learn the evaluation of a circuit C held by another party (a sender). The goal is to make receiver's work sublinear (and ideally independent) of $|C|$, using a 2-message protocol. The transformation technique may be of independent interest, and have various additional applications. The framework uses techniques akin to Gentry's bootstrapping and conditional disclosure of secrets (CDS [AIR01]) combining a non circuit private FHE scheme, with a homomorphic encryption (HE) scheme for a smaller class of circuits which is maliciously circuit-private. We devise the first known circuit private FHE, by instantiating our framework by various (standard) FHE schemes from the literature.
Keywords: Fully homomorphic encryption, computing on encrypted data, privacy, malicious setting.

* Work supported in part by NSF grants 09165174, 1065276, 1118126 and 1136174, US-Israel BSF grant 2008411, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014 -11 -1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

** Work supported in part by NSF grants 09165174, 1065276, 1118126 and 1136174. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014 -11 -1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

1 Introduction

In this paper, we devise a first fully homomorphic encryption scheme (FHE) [Gen09] that satisfies (a meaningful form of) circuit privacy in the malicious setting—a setting where the public key and ciphertext input to Eval are not guaranteed to be well-formed. We present a framework for transforming FHE schemes with no circuit privacy requirements into maliciously circuit-private FHE. The transformation technique may be of independent interest, and have various additional applications. The framework uses techniques akin to Gentry's bootstrapping and conditional disclosure of secrets (CDS [AIR01]) combining a non circuit private FHE scheme, with a homomorphic encryption (HE) scheme for a smaller class of circuits which is maliciously circuit-private. We then demonstrate an instantiation of this framework using schemes from the literature.

The notion of FHE does not require circuit privacy even in the semi-honest setting (but rather standard IND-CPA security, the ability to evaluate arbitrary circuits on encrypted inputs and encrypted outputs being "compact"). In [Gen09] and [vDGHV09, appendix C], the authors show how to make their FHE schemes circuit-private in the semi-honest setting.

One natural application of (compact) FHE is the induced 2-message, 2-party protocol, where a receiver holds an input x, and a sender holds a circuit C; the receiver learns $C(x)$, while the sender learns nothing. In the first round the receiver generates a public-key pk, encrypts x to obtain c, and sends (pk, c). The sender evaluates C on (pk, c) using the schemes' homomorphism, and sends back the result. An essential requirement is that receiver's work (and overall communication) is $poly(k, n, o(|C|))$, where k is a security parameter, ideally independent of $|C|$ altogether. This application of homomorphic encryption, termed *computing on encrypted data*, was studied both in several works [IP07,BKOI07] predating Gentry's first fully homomorphic scheme, and mentioned in [Gen09].

The underlying scheme's IND-CPA security translates into the standard simulation-based notion of *privacy* against a malicious sender in the stand-alone model[4] (but not any form of correctness against a malicious sender). The circuit privacy of the scheme translates into a privacy guarantee against a malicious receiver (of the same "flavor"). While standard FHE (without extra requirements) does not imply any security guarantees against malicious receivers, the semi-honestly circuit-private schemes from (e.g.) [vDGHV09] imply standard simulation-based security against semi-honest receivers. Thus, a maliciously circuit-private scheme induces a protocol which is private against malicious corruptions in the stand-alone model.

Let us now define maliciously circuit-privacy of FHE more precisely. We say a scheme is circuit-private if it satisfies the following privacy notion ala [IP07], stating that any (pk, c) pair induces some "effective" encrypted input x^*:

Definition 1. *(informal). We say a $\mathcal{C}$-homomorphic[5] encryption scheme* (KeyGen, Enc, Eval, Dec) *is (maliciously) circuit-private if there exists an unbounded algorithm* Sim*, such that for all security parameters k, and all pk^*, c^* there exists x^*, such that for all circuits $C \in \mathcal{C}$ over $|x^*|$ variables* $\mathsf{Sim}(1^k, C(x^*)) =^s \mathsf{Eval}(1^k, pk^*, C, c^*)$ *(statistically indistinguishable). We say the scheme is semi-honestly circuit-private if the above holds only for* well-formed pk^*, c^* *pairs.*

An FHE satisfying Definition 1 induces a protocol private against a malicious sender (by IND-CPA security of the FHE), but private against unbounded malicious receivers with unbounded simulation.[6]

On one hand, this privacy notion is weaker compared to full security as the simulation is not efficient; on the other hand, it is stronger in the sense that it holds against unbounded adversaries as well.

Due to impossibility results for general 2-round sender-receiver computation in the plain model (e.g. [BLV04]), this notion has become standard in the (non-interactive, plain model) setting of computing on encrypted data [NP01,AIR01,HK12,IP07] as a plausible relaxation.

It is important to note that we only consider the plain model. If preprocessing, such as CRS was allowed, the malicious case could be easily reduced to the semi-honest case. That is, given CRS, Enc could have added

[4] Privacy against a malicious sender comes "for free", as the protocol is 2-round, and the client speaks first.

[5] In particular, for fully homomorphic schemes, $\mathcal{C}$ is the class of all circuits.

[6] Jumping ahead, settling for computational indistinguishability with unbounded simulation would allow for somewhat simplified constructions. However, we shoot for the best achievable privacy notion.

a NIZK proving that the key is well-formed, and that the ciphertext is a valid ciphertext under that public key. Then, Eval could explicitly check that the proof is valid, if not return $\perp$, otherwise run Eval as for the semi-honest setting (for that scheme). Some care needs to be taken even in this setting, so that the scheme for the semi-honest setting used has somewhat enhanced privacy. More specifically, it needs to hold assuming (pk, c) are in the support of valid pk and $c \in Enc_{pk}(\cdot)$ respectively, but not that the *distribution* of (pk, c) is identical to the honestly generated one. Indeed, such a semi-honestly circuit-private scheme has been put forward in [GHV10] (when applied to a perfectly correct FHE scheme). On the other hand, the semi-honestly private scheme suggested in [vDGHV09] needs that pk has the proper distribution ($\mathsf{KeyGen}(1^k)$), rather than just being in the support of that distribution.

To summarize, our main "take home" theorem is as follows.

Theorem 1. *(informal) Assume an FHE scheme $\mathcal{F}$ with decryption circuits in NC^1 exists. Assume further there exists a maliciously circuit private HE $\mathcal{B}$ that supports bit OT exists. Then, there exists a maliciously circuit private multi-hop FHE scheme.*

There exist several instantiations of the theorem by ingredients from the literature. All known FHE schemes from the literature, such as [Gen09,vDGHV09,BV11] have efficient decryption circuits as required by the Theorem. Some candidates for $\mathcal{B}$ are [NP01,AIR01,HK12].

In terms of implications to MPC, our result can be interpreted as non-interactive 2PC protocols with asymmetric inputs as follows.

Theorem 2. *(informal) Assume the preconditions of Theorem 1 hold. Then, there exist 2-message client-server MPC protocols where the client holds x, and the server holds $n = 1^{|x|}$ a circuit C with n inputs (which may be much larger then x), and 1^k a security parameter. The client learns $C(x)$ (but nothing else, not even $|C|$), and the server learns nothing about x (but $|x|$). The privacy guarantee for the client is standard simulation-based computational privacy. The privacy guarantee for the server is based on unbounded simulation (against possibly unbounded clients). The protocols's communication is at most $poly(n, k)$ (as the client is efficient in its own input).* [7]

Multi-hop circuit-private FH. It is a desirable property of an FHE scheme that the outputs of Eval applied to any given circuit C mapping $\{0,1\}^n$ to $\{0,1\}^m$ can be fed again into Eval running on a circuit taking $\{0,1\}^m$ as input, and so on - an unbounded number of times. In the terminology of [GHV10], this property of a HE scheme is referred to as multi-hop. If only upto some i such iterations are supported, the scheme is called i-hop. The standard definition of FHE, thus corresponds to 1-hop encryption. However, there exists a simple transformation for any compact FHE into multi-hop. This is done by including an encryption of the secret key in the public key, and homomorphically decrypting the encrypted outputs received using the encrypted key bits (as in Gentry's bootstrapping theorem [Gen09,GHV10]). The maliciously circuit-private FHE resulting from our construction is also designed to be only 1-hop, but the standard transformation does not make it multi-hop, as it does not preserve malicious circuit-privacy.

In Section 3.3, we define multi-hop maliciously circuit-private HE, and sketch a modification to our 1-hop scheme, making it multi-hop. Our transformation starts with the above transformation, and adds some validation of the added key bits.

1.1 Previous Work

Circuit private FHE implicit in work on MPC. As explained above, (compact) HE naturally gives rise to non-interactive client-server protocols for computing on encrypted data (and the privacy level of the protocol

[7] In fact, the above result can be interpreted as general "size hiding" 2PC with asymmetric inputs. The case in Theorem 2 is a special case with $F(x, y)$ being the universal function of evaluating a circuit y on input x. In the general case, F is some polynomial-time computable function. The client learns $F(x, y)$ (but not even $|y|$), and the server learns only $|x|$. This can be implemented by letting the server set $C = F_y(x)$, and run the protocol from Theorem with input $C, |x|$ for the server and input x for the client.

depends on the notion of circuit privacy of the FHE). An essential requirement is that client's work in these protocols is sub-linear in $|C|$ (ideally independent of $|C|$).

In the other direction, standard two-message client-server protocols (inputs of similar length, only client learns output), robust against malicious receivers, induce circuit-private (not necessarily compact) HE, when applied with the universal function [CCKM00,GHV10]. Roughly the round-1 message of the client is viewed as an encryption for his input, and the senders' reply as Eval's output. This is still not an encryption scheme, as in the protocol, the client needs to "remember" the randomness generating a round-1 message in order to "decode" the reply. This is solved by setting KeyGen output a public-key, private-key pair of some public key encryption scheme as (pk, sk) respectively. Enc is augmented concatenate an encryption c_r of its randomness under pk. Eval is augmented to pass c_r as is. Intuitively, if the protocol was private against malicious clients, then so is the encryption scheme (as the randomness was known to the client anyway).

One specific construction of HE from 2PC is by combining an information-theoretic version of Yao's garbled circuits with oblivious transfer (OT) secure against malicious receivers [NP01,AIR01,HK12,IP07]. As information-theoretic Yao is only efficient for NC^1, the resulting HE scheme captures only functions in NC^1. Also, this generic construction, even in the semi-honest setting (for all known 2PC protocols from then literature) has encrypted output size at least $|C|$, while the crux of HE is having compact encrypted outputs—ideally $poly(k)$, as modern FHE schemes achieve.

Another relevant work is a recent work on 2-message 2-party evaluation of a public function $f(x, y)$, where the work of the party holding y (wlog.) is $poly(k, n, \log|f|)$, where $|f|$ is the size of f's circuit representation [DFH12]. Their protocol is maliciously UC-secure [Can01]. Instantiating f with the universal function for evaluating circuits, results in HE with good compactness properties for circuits of certain size. However, this protocol requires CRS, and thus does not translate into (circuit private) HE in the plain model.

Circuit privacy in HE literature. Without the circuit privacy requirement, FHE candidates have been proposed in a line of work following the seminal work of Gentry [Gen09,vDGHV09,BV11], to mention a few. However, these works typically do not have circuit privacy as a goal.

Circuit privacy in the semi-honest setting, for properly generated pk and c has been addressed in [Gen09,vDGHV09]. In both works, the solution method is akin to that used in additively homomorphic cryptosystems [GM84,DJ01]. The idea in the latter is to (homomorphically) add a fresh encryption of 0. In FHE, the situations is a bit more complicated, as the output of Eval typically has a different domain than "fresh" encryptions, so adding a 0 is not straightforward. However, a generalization of this technique often works (see e.g. [vDGHV09]).

Another approach suggested in [vDGHV09] for the semi-honest setting is replacing the encrypted output c with a Yao garbled circuit for decryption (with sk, c as inputs), thereby transforming any scheme into a semi-honestly circuit-private one.

The work of [GHV10] considers a generalization of circuit privacy of HE (referred there as function privacy) to a setting with multiple evaluators and single encryptor (and decryptor), where all but a single evaluator E_i can collude to learn extra information about E_i's circuit. Among other contributions, in [GHV10], the authors further abstract the Yao-based approach from [vDGHV09] as a combination of two HE scheme, one compact but not private, the other (semi-honestly) private but not compact, so that the result is both compact and (semi-honestly) private. We use this transformation as is as a first step in our transformation (along with some additional ideas formulated in [GHV10]).

As mentioned above, the malicious setting (with compact encrypted outputs) has been addressed in the context of Oblivious Transfer (OT) [NP01,AIR01,HK12] (these works can be viewed as HE for the limited class of Oblivious Transfer functions). For broader classes of functions [IP07] devise maliciously circuit-private HE for depth-bounded *branching programs* (with partial compactness).

All of the above schemes use the Conditional Disclosure of Secrets (CDS) methodology [GIKM98]. CDS is a light-weight alternative to zero-knowledge proofs, that receives a secret string, an encryption c of some x in an HE, and the corresponding pk. It discloses the secret iff. x satisfies a certain condition. CDS was originally defined for well-formed pk, c, leading to semi-honestly circuit private HE constructions [AIR01]. The CDS from [AIR01] works for additive HE with ciphertexts over groups of a prime order, and [Lip05] generalized it to groups of sufficiently "rough" composite order. For specific groups, the technique of [Lip05] turned out to generalize to situations where (pk, c) may not be well-formed. Roughly, the secret luckily remains hidden

even if the CDS was obliviously performed on the (possibly malformed) encryptions and pk as if they were proper. Such CDS was used in [HK12,IP07] to obtain maliciously circuit-private HE.

1.2 Our techniques

We devise a framework for transforming FHE schemes with no circuit privacy requirements into maliciously circuit-private FHE. We use 2 ingredients which have implementations in the literature: powerful (evaluate circuits) compact FHE without privacy $\mathcal{F}$, and weak (evaluate formulas) non-compact maliciously circuit-private HE $\mathcal{P}$ (by "compact" we refer to the strong requirement that encrypted outputs have size $poly(k)$, where k is the security parameter). Our construction proceeds in three steps:

Lemma 1 ([GHV10]). *A compact FHE without privacy can be upgraded to (compact) semi-honestly circuit private by decrypting its encrypted output under a (possibly non-compact) enhanced semi-honestly private HE, capable of evaluating the decryption circuit. The resulting scheme has enhanced semi-honest circuit privacy, assuming only that pk, c are in the support of honestly generated pairs, rather then being distributed as honestly generated pairs.*

Lemma 2 (this paper). *An enhanced semi-honestly circuit-private FHE can be upgraded to maliciously circuit-private by homomorphically validating its keys and inputs under a (possibly non-compact) maliciously circuit-private FHE capable of evaluating a circuit validating that (pk, c) are well-formed (provided a suitable witness as additional input).*

The output resulting from composing these two steps is not compact—but fortunately:

Lemma 3 (this paper,same construction as [GHV10] for semi-honest setting). *Any circuit-private FHE can be upgraded to compact by homomorphically decrypting its output under a compact (F)HE (while preserving circuit-privacy).*

Let us now elaborate on each of the steps.

Step 1. The first step transforms a "main" (compact) FHE scheme $\mathcal{M}_1$ into a semi-honestly circuit-private scheme ($\mathcal{M}_2$). An output encrypted via $\mathsf{Eval}_{\mathcal{M}_1}$ may contain extra information about the structure of C (though limited to $poly(k)$ since $\mathcal{M}_1$ is compact). An easy way to strip all information beyond the value of the function is to decrypt it already during Eval under an "auxiliary" scheme $\mathcal{A}_1$ which is semi-honestly circuit-private. To make sure the evaluator learns no secret information $(sk_{\mathcal{M}_1}, x)$, the decryption is done "blindly", using $\mathcal{A}_1$'s homomorphic properties. Details follow.

KeyGen generates a public key $pk = (pk_{\mathcal{M}_1}, pk_{\mathcal{A}_1}, a_{sk_{\mathcal{M}_1}} = \mathsf{Enc}_{\mathcal{A}_1}(sk_{\mathcal{M}_1}))$ and secret key $sk_{\mathcal{A}_1}$. Enc simply outputs $c_{\mathcal{M}_1} = \mathsf{Enc}_{\mathcal{M}_1}(x)$. Now, Eval first computes $out_{\mathcal{M}_1} = \mathsf{Eval}_{\mathcal{M}_1}(C, c_{\mathcal{M}_1})$, and outputs $\mathsf{Eval}_{\mathcal{A}_1}(\mathsf{Dec}_{\mathcal{M}_1}, (out_{\mathcal{M}_1}, a_{sk_{\mathcal{M}_1}}))$.[8] Dec simply applies $\mathsf{Dec}_{\mathcal{A}_1}$ to Eval's output.

Enhanced semi-honest circuit privacy of the resulting scheme follows by (semi-honest) circuit privacy of $\mathcal{A}_1$, and the correctness of $\mathcal{M}_1$. For enhanced circuit privacy, we assume perfect correctness of $\mathcal{M}_1$ rather then allowing for negligible decryption error, as is common in the FHE literature. The reason is that otherwise, the receiver could pick pairs (pk, c) for which the output of Eval is not $C(x)$ with high over Eval's randomness, and potentially reveal a bit about the circuit not consistent with x.

Note that since $\mathcal{M}_1$ is compact $|out_{\mathcal{A}_1}| = \mathsf{poly}(k)$ even if $\mathcal{A}_1$ is not compact. $\mathcal{M}_1, \mathcal{A}_1$ can be instantiated via almost any FHE from the literature, and (non-compact) semi-honestly circuit-private HE obtained from non-interactive 2PC protocols, such as Yao-based protocols (see above). In particular, although most FHE schemes from the literature have (negligible) decryption errors, they can be modified to have perfect correctness, while maintaining security.[9]

[8] The trick of "re-encrypting" under $\mathcal{A}_1$ using $\mathcal{A}_1$'s own Eval procedure to perform the decryption is similar in Gentry's bootstrapping technique in [Gen09] for transforming a "somewhat homomorphic" scheme into (unleveled) FHE. One difference is that here we can use two different schemes. Another difference is that for the purpose of reducing noise via Gentry's bootstrapping theorem, it is important to hardwire c as a string into the decryption circuit, rather then supplying an encryption of it. In step 1, we can afford introducing c into Dec in either way.

[9] Consider for example the [BV11] scheme can be modified to use Gaussian noise truncated to a value which still does not incur decryption errors. It is easy to prove that the new scheme remains secure under the same (LWE)

Step 2. The above approach generally fails in the malicious setting, even with stronger ingredients. Let $\mathcal{A}_2$ be a maliciously circuit-private scheme, and $\mathcal{M}_2$ be the semi-honestly circuit-private FHE resulting from step 1. An obvious attempt is using $\mathcal{A}_2$ instead of $\mathcal{A}_1$ in the first decryption; another is repeating the construction, taking $\mathcal{M}_2$ and additionally decrypting its output under $\mathcal{A}_2$. Neither is enough.

On a high level, in a maliciously circuit-private $\mathcal{A}_2$, any $pk_{\mathcal{A}_2}, a_{sk_M}$ "induce" an encryption of *some* sk_M^* under $\mathcal{A}_2$, so, we may think of them as being well-formed. However, the following potential attack exists. Assume even that pk_M is well-formed, but c_M is arbitrarily malformed. Thus out_M is not guaranteed to be a valid encryption of some x, and may instead carry some other arbitrary (upto $poly(k)$) bits of information about C. In turn, $\mathsf{Dec}_M(c_M, sk_M^*)$ may leak some of this information (even if sk_M^* is the right key corresponding to pk_M).

To fix the above potential attack (and other similar ones) and achieve malicious circuit privacy, we validate (pk, c) of $\mathcal{M}_2$. (In particular, for $\mathcal{M}_2$ resulting from step 1, we need to also check that $a_{sk_{\mathcal{M}_1}}$ encrypts an $sk_{\mathcal{M}_1}$ that corresponds to $pk_{\mathcal{M}_1}$ as part of validating $pk_{\mathcal{M}_2}$ is well-formed.) Such validation is generally hard, so we augment $\mathsf{KeyGen}_{\mathcal{M}_2}$ and $\mathsf{Enc}_{\mathcal{M}_2}$ to supply a helpful witness (encrypted under $\mathcal{A}_2$). For starters we want the full *randomness* used by them. However known $\mathcal{A}_2$ instantiations with malicious circuit privacy against unbounded adversaries, as we require can only evaluate functions in NC^1 and $\mathsf{KeyGen}_{\mathcal{M}_2}, \mathsf{Enc}_{\mathcal{M}_2}$ need not be in NC^1. [10] But surely they are in P, and we can use the standard transform to validate polynomial work in parallel we require the witness to also include values of *all intermediate wires* of $\mathsf{KeyGen}_{\mathcal{M}_2}, \mathsf{Enc}_{\mathcal{M}_2}$, validate all gates in parallel, and have a log-depth AND tree. A similar issue arises already in step 1, where $\mathcal{A}_1$ needs to evaluate the decryption circuit of $\mathcal{M}_1$. The same trick can not by applied there, as the values to decrypt are not known to the receiver. Thus we need to assume $\mathsf{Dec}_{\mathcal{M}_1}$ is in NC^1, which is fortunately satisfied by all known schemes from the literature.

Somewhat more precisely, we transform $\mathcal{M}_2$ in the following non-blackbox way.

- $\mathsf{Enc}(x)$ outputs $c_{\mathcal{M}_2} = \mathsf{Enc}_{\mathcal{M}_2}(r', pk_{\mathcal{M}_2}, x)$ along with $a_{r_{\mathcal{M}_2}} = \mathsf{Enc}_{\mathcal{A}_2}(r)$ - a witness that $c_{\mathcal{M}_2}$ is a proper encryption (derived from r').
- $\mathsf{Eval}(C, c_{\mathcal{M}_2})$: Let $Validate(pk_{\mathcal{M}_2}, c_{\mathcal{M}_2}, out, rk_{\mathcal{M}_2}, r_{\mathcal{M}_2})$ denote a circuit where $rk_{\mathcal{M}_2}, r_{\mathcal{M}_2}$ are purported witnesses for the well-formedness of $pk_{\mathcal{M}_2}, c_{\mathcal{M}_2}$ respectively. It outputs out if $rk_{\mathcal{M}_2}, r_{\mathcal{M}_2}$ certify well-formedness of $pk_{\mathcal{M}_2}, c_{\mathcal{M}_2}$, and the all-zero vector otherwise.
  - Compute $out_{\mathcal{M}_2} = \mathsf{Eval}_{\mathcal{M}_2}(pk_{\mathcal{M}_2}, C, c_{\mathcal{M}_2})$.
  - Output $out = \mathsf{Eval}_{\mathcal{A}_2}(Validate_{pk_{\mathcal{M}_2}, c_{\mathcal{M}_2}, out_{\mathcal{M}_2}}(a_{rk_{\mathcal{M}_2}}, a_{r_{\mathcal{M}_2}}))$ (that is, fixing the suitable variables in $Validate$ to the values at the subscript).
- Dec outputs $\mathsf{Dec}_{\mathcal{M}_2}(sk_{\mathcal{M}_2}, \mathsf{Dec}_{\mathcal{A}_2, out}(sk_{\mathcal{A}_2}))$.

Note that $pk_{\mathcal{M}_2}, c_{\mathcal{M}_2}, out_{\mathcal{M}_2}$ are hardwired into $Validate$, rather then encrypted via $\mathcal{A}_2$ (in Eval). The subtle reason for this, is that if $pk_{\mathcal{M}_2}$ is malformed, "encrypting" values under it can yield effective encryptions of different values, possibly making us perform the "wrong" validation.

In a nutshell, the construction works by enhanced semi-honest circuit privacy of $\mathcal{M}_2$ if $(pk_{\mathcal{M}_2}, c_{\mathcal{M}_2})$ is well-formed, and the validation procedure takes care of the fact that it is indeed well-formed (otherwise, no information whatsoever is revealed about C).

Merging steps 1 and 2 (Section 3.1). Steps 1+2 provide a clear blueprint for transforming a (compact) FHE $\mathcal{F}$ into maliciously circuit-private FHE by combining it with a maliciously circuit-private HE $\mathcal{P}$ capable of evaluating $\mathcal{F}$'s decryption and validation circuits. That is, the same maliciously circuit-private $\mathcal{P}$ may instantiate both $\mathcal{A}_1$ and $\mathcal{A}_2$. $\mathcal{M}_2$ resulting from step 1 is fed into step 2. In section 3.1 we describe the natural composition of the two steps in a single protocol, making a small shortcut that exploits the structure of $\mathcal{M}_2$ output by step 1, and the fact that $\mathcal{A}_1 = \mathcal{A}_2$. Namely, we do not have to check the well-formedness of $a_{sk_{\mathcal{M}_2}}$ as an encryption under $\mathcal{A}_2$.

---

assumption - essentailly because samples larger then some bound hae negligible probability of being sampled. A similar trick can be applied to other LWE-based FHE schemes [BV11,Bra12,GSW13].

[10] Settling for unbounded simulation against bounded distinguishers, would allow to evaluate arbitrary circuits under the scheme without further complications, however, we are shooting for the strongest possible privacy guarantee.

Step 3. Let us look more closely at the compactness of the scheme achieved from steps 1+2. The validate circuit that needs to be evaluated has input of size $m = poly(k, n)$ (and polynomial size $|C|$). Thus, if $\mathcal{P}$ is not compact, the encrypted output size is some $poly(|C|, m, k)$ – also $poly(k, n)$.

This is acceptable for our main application of computing on encrypted data, as receiver's input is of size n. However, it would be nice to meet the current standard for FHE where encrypted output size is independent of n.

A more complicated setting is that of leveled FHE. In such schemes, $\mathsf{KeyGen}(1^k, 1^d)$ generates an additional key pk_{Eval} received by Eval (all the rest remains the same), which may grow with the bound d on depth of circuits to be evaluated. In a nutshell, such schemes are often considered in the FHE literature in order to make the underlying assumptions more plausible, avoiding so called (weak) circular security assumptions.

The encrypted output size is $poly(k, n, d)$—quite undesirable!

The idea is to combine the circuit-private (non-compact) HE $\mathcal{M}_3$ resulting from steps 1+2 with a compact FHE $\mathcal{A}_3$ with no circuit privacy "in the opposite direction" from step 1. That is, use $\mathcal{A}_3$ to homomorphically decrypt the output of $\mathcal{M}_3$ to "compress" it. Intuitively, even though the FHE $\mathcal{A}_3$ used for decrypting is not circuit-private, the resulting scheme is, because $\mathsf{Eval}_{\mathcal{A}_3}$ merely acts upon a string that we originally were willing to output "in the plain", so there is no need to protect it.

2 Preliminaries

Notation. We use $\overrightarrow{\text{arrow}}$ to denote vectors, though not always—we tend to use it to stress element-wise handling, e.g. bit-by-bit encryptions. For a function $f(a, b, c, \ldots)$, we write $\overrightarrow{f}(\overrightarrow{a}, b, \overrightarrow{c}, \ldots)$ as a shorthand for element-wise application $(f(a_1, b, c_1, \ldots), f(a_2, b, c_2, \ldots), \ldots)$. When considering function vectors, all inputs which are the same in all executions appear without an arrow (even if they are vectors by themselves).

For a pair of vectors u, v (u, v) denotes the vector resulting from concatenating u, v. For vectors u, v over some U^t, V^t, $(u; v)$ denotes $((u_1, v_1), \ldots, (u_t, v_t))$.

For a function $f(a, b, c, \ldots)$, we denote the set of functions fixing some of its parameters (here b, c) as follows $f|_{b,c}(a, \ldots)$. $f|_{b=B,c=C}$ denotes a function fixing the parameters to particular values B, C respectively.

For randomized algorithms $A(x, r)$, we sometimes write $out \in A(x)$ as a shorthand for $out \in support(A(x, r))$.

By $\mathrm{negl}(k)$ we refer to a function that for all polynomials $p(k)$, $\mathrm{negl}(k) < \frac{1}{p(k)}$ for all $k > K$, where K is a constant determined by p.

Usually an encryption of x under scheme $\mathcal{Y}$ will be named y_x.

Definition 2 (Indistinguishability of distributions). *Let $\{X_k(x)\}_{k\in\mathbb{N},x\in\{0,1\}^*}, \{Y_k(x)\}_{k\in\mathbb{N},x\in\{0,1\}^*}$ be two distribution ensembles. We say that the ensembles are statistically indistinguishable, if for all circuit families $\{C_n(x_1, \ldots, x_n)\}_n$ $x \in \{0,1\}^n$, $|Pr[C_n(X_k(x))] - Pr[C_n(Y_k(x))]| = \mathrm{negl}(k)$. We denote $X_k =^s Y_k$. If the above holds for all circuit families of polynomially bounded size, we say X_k, Y_k are computationally indistinguishable, denoted by $X_k =^c Y_k$.*

Representation Models When we say a HE scheme is $\mathcal{C}$-homomorphic for a class of functions, we actually mean functions having programs C from the set $\mathcal{C}$ of programs. By a program C, we mean a string representing a function $f : \{0,1\}^n \to \{0,1\}$. The correspondence between programs C and the function it represents is determined by an underlying representation model U. A *representation model* $U : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is an efficient algorithm taking an input (C, x), and returning $f(x)$, where f is the function represented by C. By $|C|$ we simply refer to the length of the string C (as opposed to size(C), which is a related measure depending on U, such as the number of gates in a boolean circuit). For completeness, for circuits and other models we let $U(C, x) = 0$ whenever the input (C, x) is syntactically malformed.

As typical in the FHE literature, our default representation model is boolean circuits, unless stated otherwise. We use circuits over some complete set of gates, such as $\{AND, XOR, NOT\}$. Another model we will consider is boolean formulas, which are circuits with fanout 1. We assume the underlying DAGs of circuits are connected. For formulas, we assume wlog. that $depth(C) \leq c \log size(C)$ for a global constant c (that is, that they are "balanced"). For a circuit C, $size(C)$ denotes the number of wires in C's underlying

graph, and $depth(C)$ the number of gates on the longest path between an input wire and the output wire of the circuit. By NC^1 we refer to the class of function (families) with uniform formulas of size $poly(n)$.

2.1 Homomorphic encryption

Throughout the paper k denotes the security parameter taken by HE schemes. A (public-key) homomorphic encryption scheme (HE) $\mathcal{E} = (\mathsf{KeyGen}_E, \mathsf{Enc}_E, \mathsf{Eval}_E, \mathsf{Dec}_E)$ is a quadruple of PPT algorithms as follows.

$\mathsf{KeyGen}(1^k)$**:** Outputs a public key, secret key pair (pk, sk).
$\mathsf{Enc}(pk, b)$**:** Takes a public key and a bit b to encrypt, and returns an encryption c of the bit under pk.
$\mathsf{Eval}(pk, C, c = (c_1, \ldots, c_n))$**:** Takes a public key pk, a bit-by-bit encryption c of a bit vector $x \in \{0,1\}^n$, a function represented by a program C (encoded in a pre-fixed representation model U) and outputs an encryption *out* of bit $U(C, x)$. We assume wlog. that pk includes 1^k (intuitively, this is intended to handle maliciously generated public keys). We refer to outputs of Eval as "encrypted outputs".
$\mathsf{Dec}(sk, out)$**:** Takes a secret key sk, and a purported output *out* of Eval, outputs a bit.

Throughout the paper, HE is semantically secure if $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ satisfies standard IND-CPA security for public key encryption schemes as in [GM84]. An HE scheme is *weakly circular-secure* if even knowing a bit-by-bit encryption of the schemes' secret key sk, the adversary still has negligible advantage in the IND-CPA experiment.In this paper, we consider a general notion of homomorphism, under various program classes, rather then just circuits (to express weaker homomorphism properties then FHE).

Definition 3 ($(U, \mathcal{C})$-homomorphic encryption). *Let $\mathcal{C} = \bigcup \mathcal{C}_k$. We say a scheme $\mathcal{E}$ is $(U, \mathcal{C})$-homomorphic if for every $k > 0$ and every program $C \in \mathcal{C}_k$ on inputs $x \in \{0,1\}^n$, the experiment*

$$(pk, sk) \xleftarrow{\$} \mathsf{KeyGen}(1^k)$$
$$out \xleftarrow{\$} \mathsf{Dec}(sk, \mathsf{Eval}(pk, C, \overrightarrow{\mathsf{Enc}}(pk, \overrightarrow{x})))$$

outputs $out = U(C, x)$ with probability 1 for all $x \in \{0,1\}$ and all random choices of the algorithms involved. We say the scheme is k-independently homomorphic if $\mathcal{C}_k = \mathcal{C}$ for all k.[11]

By default our schemes are k-independently homomorphic (in particular the $\mathcal{C}_k$'s are not explicitly defined).

If a scheme satisfies that $\mathcal{C}$ equals the set of all circuits, and is k-independetly homomorphic, we refer to it as "fully homomorphic" (FHE).

Definition 4. *We say a $(U, \mathcal{C})$-homomorphic scheme $\mathcal{E}$ is* compact *if there exists an output bound $B(k, n, |C|) = \mathsf{poly}(k)$ on the output of Eval on all $1^k, n$ and $C \in \mathcal{C}_k$ on n bits.*

Another standard variant of HE we consider is *leveled* HE. In this variant, KeyGen is modified to take another parameter 1^d. KeyGen outputs keys (pk, sk), where pk includes a fixed-size part pk_{Enc}, which depends only on k; likewise, sk depends only on k. Enc is modified to accept pk_{Enc} as the public key, and only Eval receives the entire public key pk. In particular, Enc is the same for all d. The notions of compact HE is as for non-leveled schemes ($B(k, n, |C|) = poly(k)$).For compact schemes, the algorithm Dec is also independent of d. We say such a leveled compact scheme is an FHE, if for all D, the (standard) HE $\mathcal{E}^D$ induced by fixing $d = D$ when calling $\mathsf{KeyGen}(1^d, \cdot)$ induces a k-independently $\mathcal{C}$-homomorphic scheme $\mathcal{E}^D$, where C is the set of all depth-D circuits. The encrypted outputs' size is still $poly(k)$ (for a global polynomial independent of d).

Standard FHE schemes can be thought of as a special case of leveled FHE schemes where KeyGen simply ignores d. Thus, all schemes $\mathcal{E}^d$ are the same (standard) FHE scheme. We refer to this special case as *unleveled* FHE. A HE scheme is maliciously circuit private if every (pk, c) (even arbitrarily malformed) induce some "effective" input x^*.

[11] For instance, the notion of "somewhat homomorphic" schemes in the FHE literature corresponds to non k-independent schemes, where C_k is a set of circuits with depth bounded by some function of k.

Definition 5. *Let* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *denote a* $(U, \mathcal{C})$*-homomorphic scheme. We say* $\mathcal{E}$ *is (maliciously)* circuit private *if there exist unbounded algorithms* $\mathsf{Sim}(1^k, pk^*, c^*, b)$*, and deterministic* $\mathsf{Ext}(1^k, pk^*, e^*)$*, such that for all* k*, and all* $pk^*, c^* = (c_1^*, \ldots, c_n^*)$ *and all programs* $C : \{0,1\}^n \to \{0,1\} \in (U, \mathcal{C})$ *in* $\mathcal{C}_k$ *the following holds:*

- $\overrightarrow{x^*} = \overrightarrow{\mathsf{Ext}}(1^k, pk^*, \overrightarrow{c^*})$.
- $\mathsf{Sim}(1^k, pk^*, c^*, U(C, x^*)) =^s \mathsf{Eval}(1^k, pk^*, C, c^*)$.

In particular, for circuits $C(x_1, \ldots, x_n) \in \mathcal{C}_k$ *the output distribution of* Eval *(including length) depends only on* n, k*. For leveled schemes,* Sim *and* Ext *also take a depth parameter* 1^d*. We say a scheme is semi-honestly circuit-private if the above holds, where* pk^*, c^* *belong to the set of* well-formed *public-key, ciphertext pairs.*

On leveled HE. As mentioned before, the reason this relaxation of compact FHE is considered, is that so far, all (compact) FHE schemes were obtained using Gentry's bootstrapping theorem, which assumes circular security of an underlying *bootstrappable* HE scheme.

Definition 6 (Bootstrappable homomorphic encryption). *For a* $\mathcal{C}$*-homomorphic scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$*, let* $\mathsf{Dec}_k^{\oplus}(c_1, c_2, sk), \mathsf{Dec}_k^{\wedge}(c_1, c_2, sk), \mathsf{Dec}_k^{\neg}(c, sk)$ *denote the augmented decryption circuits of the scheme, taking two encrypted outputs* out_1, out_2*, decrypting them to obtain bits* b_1, b_2 *or* b *and returning* $b_1 \oplus b_2$ *and* $b_1 \wedge b_2$ *or* $\neg b$ *respectively. We say the scheme is* bootstrappable *if for all* k*,* $\mathsf{Dec}_k^{\oplus}|_{c_1,c_2}$, $\mathsf{Dec}_k^{\wedge}|_{c_1,c_2}$, $\mathsf{Dec}_k^{\neg}|_c$ *are in* $\mathcal{C}_k$*.*

Theorem 3 ([Gen09]). *Assume a bootstrappable,* $\mathcal{C}$*-homomorphic HE scheme* $\mathcal{E} = (\mathsf{KeyGen}_E, \mathsf{Enc}_E, \mathsf{Eval}_E, \mathsf{Dec}_E)$ *exists. Then there exists a compact leveled FHE scheme* $\mathcal{BS}$ *with the following properties:*

- $\mathsf{KeyGen}_{BS}(1^d, 1^k)$ *outputs* $(pk_{BS}, sk_\ell$*, for* $pk_{BS} = (p_1, \ldots, p_\ell, e_1, \ldots, e_\ell)$*, where the* (pk_i, sk_i)*'s are output by* $\mathsf{KeyGen}_E(1^k)$*; for all* $i < \ell$ $e_i \in \mathsf{Enc}(pk_{i+1}, sk_i)$ *where* $\ell = O(d)$*.*
- $\mathsf{Enc}, \mathsf{Dec}$ *apply* $\mathsf{Enc}, \mathsf{Dec}$ *to* pk_1 *and* sk_ℓ *respectively.*

If $\mathcal{E}$ *is weakly circular secure, one can set* $pk_i = pk_1$ *and* $sk_i = sk_1$*, where* (pk_1, sk_1) *are sampled from* $\mathsf{KeyGen}(1^k)$ *(which results in a standard FHE scheme).*

Constructions of bootstrappable HE schemes from (more) standard assumptions (rather than just assuming circular security) are currently unknown.[12] On a high level, bootstrapping-based leveled FHE schemes allow to circumvent this difficulty by encrypting each circuit level under a different key, according to an acyclic sequence of independently generated keys. The transformation involves homomorphic decryption (and reincryption) of the encrypted outputs under the same scheme on each level (much as we do in step 3, for instance). The goal of this operation is to reduce the noise level of the ciphertext (which is low for outputs of Enc, and grows with homomorphic operations, especially multiplications), as decryption correctness is lost when it becomes too high. Setting all keys to be the same results in a compact key – an unleveled scheme. On the other hand, this involves encrypting sk under its corresponding pk, which requires the circular security assumption) All these keys are published as pk.

We define leveled FHE as having $\mathsf{Enc}, \mathsf{Dec}$ algorithms which are independent of d. The requirement of Enc being independent of d is not always made in the literature on leveled FHE. Consider for instance "second generation" constructions of leveled FHE that use modulus switching techniques to improve noise growth [BV11,BGV12,Bra12]. In [BGV12], for instance, Enc depends on circuit depth as well. For our application of computing on encrypted data, we want to minimize the dependence of receiver's work on circuit parameters, and thus prefer schemes where receiver's work depends on circuit depth only in KeyGen.[13]

[12] Furthermore, contrary to common belief, it was recently shown that IND-CPA security does not imply weak circular security [Rot13].

[13] If we additionally apply bootstrapping every $d(k)$ levels, the resulting scheme becomes compact in the sense we need. "Squashing" using sparse subset sum is not required for a proper setting of $d(k)$.

3 Framework

In this section we spell out the construction outlined in the introduction in more detail, but combining steps 1 and 2 for simplification. We will need the representation model $(U_{SI}, \mathcal{C})$ (from "split-input") and the weaker notion of "input-privacy" for $\mathcal{P}$:

Programs in the model are represented by a pair (C_p, C_s), where C_p is a circuit on some m variables, and $C_s \in \{0,1\}^t$ for some $t \leq m$. A program (C_p, C_s) is interpreted as a function f over $n = m - |C_s|$ variables via $U_{SI}((C_p, C_s), x) = C_p(C_s, x)$. Typically, we will consider $(U_{SI}, \mathcal{C})$-homomorphic schemes where for each (C_p, C_s), all (C_p, Z) for $Z \in \{0,1\}^*$ of length $t \leq m$ are in $\mathcal{C}$. In this case, we specify $\mathcal{C}$ as just a set of circuits.

We say a $(U_{SI}, \mathcal{C})$-homomorphic scheme is *input-private* if it satisfies Definition 5, with the only modification that Sim receives C_p as an input. (This is exactly the guarantee from converting any 2PC protocol where both parties have private input but the function is public to an HE.)

The purpose of introducing this seemingly unnatural representation model and relaxed circuit privacy definition is to allow for simpler implementations of auxiliary HE schemes we use, and overall presentation of our result. The implementation we use for $\mathcal{P}$ is based on Yao's garbled circuits, which only protect the inputs but leak quite a lot about the circuit. We could evaluate a universal circuit under Yao to get real circuit privacy but it would be overkill—in Construction 7 we'll want to evaluate under $\mathcal{P}$ functions that are essentially public (Dec, $Validate$). But we have private inputs from both the encryptor and the evaluator[14] to be fed into the non-secret function—that's why we complicate the representation model to have a secret portion C_s.

3.1 From compact FHE to circuit-private (somewhat compact) FHE

In this section we spell out the combination of steps 1 and 2 as described in the introduction. The schemes $\mathcal{F}, \mathcal{P}$ are a compact FHE, and maliciously circuit private HE respectively. Here $\mathcal{P}$ is for the split input model, and is input-private rather then circuit-private. Although the construction would go through with standard circuit privacy of $\mathcal{P}$, this simplifies the presentation and instantiation of the framework.

Given a leveled FHE $\mathcal{F}$ we define a set of programs $\mathcal{C}_{\mathcal{F}}$ (to be interpreted via U_{SI}) as follows.

1. Let $Dec_{\mathcal{F},k}(sk_{\mathcal{F}}, out_{\mathcal{F}})$ denote the decryption circuit of $\mathcal{F}$ instantiated with security parameter k (recall $\mathsf{Dec}, \mathsf{Enc}$ are independent of d).
2. Let $Validate_{k,d,n}(pk_{\mathcal{F}}, c_{\mathcal{F}}, sk'_{\mathcal{F}}, r_{FE}, out_{\mathcal{F}})$ be the circuit computing

$$Validate_{k,d,n}(\ldots) = \begin{cases} out_{\mathcal{F}} & \text{if } (pk_{\mathcal{F}}, sk_{\mathcal{F}}) \in \mathsf{KeyGen}_{\mathcal{F}}(1^k, 1^d), \text{and} \\ & \forall i \ (c_{\mathcal{F},i} \in \mathsf{Enc}_{\mathcal{F}}(b_i) \text{ for some bit } b_i \in \{0,1\}) \\ & \text{in both evaluations randomness and intermediate} \\ & \text{values equal } r_{FK}, \overrightarrow{r}_{FE} \text{ respectively.} \\ \overrightarrow{0} & \text{otherwise.} \end{cases}$$

   where:
   - $(pk_{\mathcal{F}}, sk_{\mathcal{F}})$ is a purported public-key private-key pair output by $\mathsf{KeyGen}_{\mathcal{F}}(1^k, 1^d)$. $sk'_{\mathcal{F}} = (sk_{\mathcal{F}}, r_{FK})$ where r_{FK} is the purported random string used in the generation of $(pk_{\mathcal{F}}, sk_{\mathcal{F}})$, along with all the intermediate outputs of gates in the circuit for KeyGen on input r_{FK}.[15]

[14] This situation comes up many times in this paper. The formalism of circuit-private FHE only allows one party to provide inputs but when composing it out of itself, we repeatedly needed extra private inputs from the evaluator. Usually we handle this by hard-wiring the input into the circuit. (Creating a fresh encryption works similarly but is less efficient and sometimes problematic because a malicious public key could produce malformed encryptions.) But here the circuit is going to be public, so we must keep the private input separate.

[15] As explained in the intro, the goal of the intermediate values is to put the validation in NC^1, making it implementable by known circuit-private $\mathcal{P}$ with the required notion of privacy.

- $c_{\mathcal{F}} = (c_{\mathcal{F},1}, \ldots, c_{\mathcal{F},n})$ is a purported encryption under $pk_{\mathcal{F}}$ of the input bit vector and r_{FE} is a purported vector of randomness used when generating $c_{\mathcal{F}}$, along with intermediate values for the circuit (where $r_{FK,i}$ corresponds to bit x_i).

3. Let $\mathcal{C}_{\mathcal{F}}$ include all pairs of the forms $C = (Validate_{k,d,n}(\ldots), (pk_{\mathcal{F}}, c_{\mathcal{F}}, out_{\mathcal{F}}))$ and $(\mathsf{Dec}_{\mathcal{F},k}(sk_{\mathcal{F}}), out_{\mathcal{F}})$.

Construction 7 *Let $\mathcal{F}$,$\mathcal{P}$ be schemes as above. We construct the following scheme $\mathcal{M}_3$.*

$\mathsf{KeyGen}_{\mathcal{M}_3}(1^k)$**:** *let $(pk_{\mathcal{P}}, sk_{\mathcal{P}}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{P}}(1^k)$, $(pk_{\mathcal{F}}, sk_{\mathcal{F}}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{F}}(1^k, 1^d)$, and let $sk'_{\mathcal{F}} = (sk_{\mathcal{F}}, r_{KF})$, where r_{KF} is induced by the randomness used by $\mathsf{KeyGen}_{\mathcal{F}}$ as specified in $Validate$; $\overrightarrow{p_{sk'_{\mathcal{F}}}} = \overrightarrow{\mathsf{Enc}_{\mathcal{P}}}(pk_{\mathcal{P}}, \overrightarrow{sk'_{\mathcal{F}}})$. Return $(pk_{\mathcal{M}_3}, sk_{\mathcal{M}_3}) = ((pk_{\mathcal{P}}, pk_{\mathcal{F}}, \overrightarrow{p_{sk'_{\mathcal{F}}}}), (sk_{\mathcal{P}}, sk_{\mathcal{F}}))$. Here $pk_{\mathcal{M}_3,\mathsf{Enc}} = (pk_{\mathcal{P}}, pk_{\mathcal{F},\mathsf{Enc}}, p_{sk_{\mathcal{F}}})$.*

$\mathsf{Enc}_{\mathcal{M}_3}(pk_{\mathcal{M}_3} = (pk_{\mathcal{P}}, pk_{\mathcal{F},\mathsf{Enc}}, p_{sk_{\mathcal{F}}}), b \in \{0,1\})$**:** *Return $(c, \overrightarrow{p_{r_{FE}}}) = (\mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, b), \overrightarrow{\mathsf{Enc}_{\mathcal{P}}}(pk_{\mathcal{P}}, \overrightarrow{r_{FE}}))$, where r_{FE} is derived from the randomness used by $\mathsf{Enc}_{\mathcal{F}}$ as in $Validate$.*

$\mathsf{Eval}_{\mathcal{M}_3}(1^k, pk_P = (pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk'_{\mathcal{F}}}), C, c = (c_{\mathcal{F}}; p_{r_{FE}}))$**:**

1. *If C is syntactically malformed, or $|x|$ does not match the number of inputs to C, replace C with the circuit returning $x_1 \wedge \overline{x_1}$.*
2. *Set $out_{\mathcal{F}} = \mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, C, c_{\mathcal{F}}))$, $out_{\mathcal{P}} = \mathsf{Eval}_{\mathcal{P}}(pk_{\mathcal{P}}, (\mathsf{Dec}_{\mathcal{F}}, out_{\mathcal{F}}), p_{sk_{\mathcal{F}}})$*
3. *Let $(C_p, C_s) = (Validate_{k,d,n}, (out_{\mathcal{P}}, pk_{\mathcal{F}}, c_{\mathcal{F}}))$*
4. *Compute and output $out = \mathsf{Eval}_{\mathcal{P}}(pk_{\mathcal{P}}, (C_p, C_s), p_{sk'_{\mathcal{F}}}, p_{r_{FE}})$.*

$\mathsf{Dec}_{\mathcal{M}_3}(sk_{\mathcal{M}_3}, out)$**:** *Output $y = \mathsf{Dec}_{\mathcal{F}}(sk_{\mathcal{F}}, \overrightarrow{\mathsf{Dec}_{\mathcal{P}}}(sk_{\mathcal{P}}, \overrightarrow{out}))$.*

Theorem 4. *Assume a compact leveled FHE scheme $\mathcal{F} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Enc}_{\mathcal{F}}, \mathsf{Eval}_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$ and $\mathcal{P}$ a $(U_{SI}, \mathcal{C}_{\mathcal{F}})$-homomorphic, input-private scheme, exist. Consider the resulting scheme $\mathcal{M}_3$ as specified in Construction 7 above when instantiating with $\mathcal{F}, \mathcal{P}$. Then $\mathcal{M}_3$ is a circuit-private FHE. It is unleveled iff $\mathcal{M}_3$ is unleveled, and is compact iff $\mathcal{P}$ is compact. If $\mathcal{P}$ is not compact, $\mathcal{M}_3$'s output complexity is $poly(k, d, n)$ ($poly(k, n)$ if $\mathcal{F}$ is unleveled).*

3.2 Compactization of circuit-private FHE

When instantiated by the best known constructions from the literature, Theorem 4 only yields $poly(n, k), poly(n, d, k)$ encrypted output complexity for unleveled and leveled $\mathcal{F}$ respectively. This is so, since all circuit-private $\mathcal{C}_{\mathcal{F}}$-homomorphic $\mathcal{P}$ for some FHE $\mathcal{F}$ we know of are not compact.

In this section, we devise a simple transformation (corresponding to Lemma 3 in the introduction) for making a (leveled) scheme's output compact (only $poly(k)$), while preserving circuit privacy. This will yield leveled circuit-private FHE with optimal ($poly(k)$) compactness.

The idea is to use bootstrapping similar to that of step 1 but "in reverse" order. Namely, we take a main scheme $\mathcal{M}_3$ which is circuit-private but not compact, and "decrypt" it under a scheme $\mathcal{A}_3$ which is compact but has no circuit privacy guarantees.

Theorem 5 (Compaction theorem). *Assume a leveled $\mathcal{C}$-homomorphic circuit-private scheme $\mathcal{M}_3$ and a compact FHE scheme $\mathcal{A}_3$ exist.*[16] *Then the scheme $\mathcal{M}_4$ in the following construction is a compact $\mathcal{C}$-homomorphic circuit-private scheme.*

Construction 8 *Let $\mathcal{M}_3, \mathcal{A}_3$ be HE schemes as in Theorem 5.*

$\mathsf{KeyGen}_{\mathcal{M}_4}(1^k, 1^d)$**:** *Sample $(pk_{\mathcal{M}_3}, sk_{\mathcal{M}_3}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{M}_3}(1^k, 1^d)$; let $sk'_{\mathcal{M}_3} = (sk_{\mathcal{M}_3}, r_k)$; $(pk_{\mathcal{A}_3}, sk_{\mathcal{A}_3}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{A}_3}(1^k)$; Output $(pk, sk) = ((pk_{\mathcal{M}_3}, pk_{\mathcal{A}_3}, a_{sk_{\mathcal{M}_3}}), sk_{\mathcal{A}_3})$. Here $pk_{\mathsf{Enc}} = (pk_{\mathcal{M}_3,\mathsf{Enc}}, pk_{\mathcal{A}_3}, a_{sk_{\mathcal{M}_3}})$.*

$\mathsf{Enc}_{\mathcal{M}_4}(pk, b)$**:** *Output $\mathsf{Enc}_{\mathcal{M}_3}(pk_{\mathcal{M}_3,\mathsf{Enc}}, b)$.* [17]

[16] In fact, $\mathcal{A}_3$ should only be compact and homomorphic for the circuit family it is used for. It does not need to be an FHE.

[17] Here and elsewhere, we do not distinguish between the parts of pk used in Eval and Enc, and refer to both as pk. The distinction is implied by the context.

$\mathsf{Eval}_{\mathcal{M}_4}(1^k, pk, C, c)$**:**

- $out_{\mathcal{M}_3} \xleftarrow{\$} \mathsf{Eval}_{\mathcal{M}_3}(1^k, pk_{\mathcal{M}_3}, C, c)$.
- *Let* $Dec_{\mathcal{M}_3,k}$ *denote the decryption circuit of* $\mathcal{M}_3$ *with parameter* k. *Then* $Dec_{\mathcal{M}_3,k}|_{out=out_{\mathcal{M}_3}}(sk_{\mathcal{M}_3})$ *is a circuit for decrypting (hard-wired)* $out_{\mathcal{M}_3}$ *under secret keys generated by* $\mathsf{KeyGen}_{\mathcal{M}_3}$.
- *Compute and output* $out_{\mathcal{A}_3} = \mathsf{Eval}_{\mathcal{A}_3}(1^k, pk_{\mathcal{A}_3}, Dec_{\mathcal{M}_3,k}, a_{sk_{\mathcal{M}_3}})$.

$\mathsf{Dec}_{\mathcal{M}_4}(sk = sk_{\mathcal{A}_3}, out)$**:** *Output* $\mathsf{Dec}_{\mathcal{A}_3}(sk_{\mathcal{A}_3}, out)$.

Combining Theorem 4 and Theorem 5, we get:

Theorem 6. *Assume a compact unleveled FHE scheme* $\mathcal{F}$ *and a* $(U_{SI}, \mathcal{C}_{\mathcal{F}})$*-homomorphic maliciously input-private scheme* $\mathcal{P}$ *exist. Then there exists a maliciously circuit-private compact unleveled scheme* $\mathcal{M}_4$.

Getting rid of circular security Theorem 5 still leaves open the question of obtaining compact leveled circuit-private FHE. We show that posing some mild additional efficiency requirements on $\mathcal{M}_3, \mathcal{A}_3$ in Theorem 5, we are able to modify Construction 8 to allow for a leveled $\mathcal{A}_3$.

Theorem 7. *Assume schemes* $\mathcal{F}, \mathcal{P}$ *as Theorem 6 exist. Additionally, assume* $\mathsf{Dec}_{\mathcal{P},k}(out, sk)$ *has depth* $poly(depth(C_p), k)$*, where* $out = \mathsf{Eval}_{\mathcal{P}}(C_p, C_s)$ *for some* C_s.[18] *Assume also that* $\{Dec_{\mathcal{F},k}\}$ *induced by* $\mathcal{F}$ *is in* NC^1. *Then there exists a compact leveled circuit-private scheme* $\mathcal{M}_4$.

3.3 Multi-hop circuit-private FHE

In this chapter, we focus on unleveled FHE schemes and show how to upgrade Theorem 6 to yield a multi-hop $\mathcal{M}_4$ (under the same assumptions). A multi-hop scheme is an FHE scheme, where Eval is modified to support (pk, C, c), where the c_i's are either outputs of Enc *or of previous* Eval*s*. More formally, if c_i is a purported output of Enc we label it as a level-0 execution of Eval. We recursively define an execution of Eval to be of level l, if the highest level input c_i to it is of level $l-1$. Syntactically, we modify Eval to take an encrypted output $c = (c_1, \ldots, c_n)$, where each c_i is a (purported) output of either Enc or Eval. Indeed, we may assume wlog. that we need not include the level of c_i, or even whether it is an output of Enc or of Eval, as it is possible to construct FHE schemes where the supports of level-l executions of Eval come from the same finite set for all $l \geq 0$. We say that a level-0 ciphertext is well-formed under pk, encrypting b, if it is a valid output of $\mathsf{Enc}_{pk}(b)$ under a well-formed key pk. Recursively, we say a level $l > 0$ ciphertext c' is well-formed under pk, if it is an output of Eval on a valid key pk, some circuit C, and all its input c_i's are well-formed level$< l$ ciphertexts under the same pk. If $\overrightarrow{c}$ encrypts $\overrightarrow{b}$, we say that c' encrypts $C(b)$.

Perfectly correct multi-hop is a natural recursive extension of Definition 3. Namely, we require that if a level-i ($i > 1$) ciphertext encrypts a bit b, then $\mathsf{Dec}(sk, c)$ outputs b with probability 1 over the random choices of KeyGen (when outputting (pk, sk)), and of Dec. Similarly, a scheme is i-hop if it satisfies the above correctness requirements only for level-j executions of Eval, where $j \leq i$. Thus standard FHE corresponds to 1-hop. The definitions of IND-CPA security and compactness extend for multi-hop schemes in the natural way. The definition of maliciously circuit-private multi-hop FHE is precisely Definition 5. The notion of semi-honest encryption requires that pk is a valid public key, and all c_i's are well-formed ciphertexts under pk of some level $l_i \geq 0$ respectively. [19]

The construction induced by Theorem 6 is 1-hop, but not multi-hop. There exists a straightforward transformation from compact FHE schemes into multi-hop schemes (see discussion in Section 1), but it does not necessarily preserve malicious circuit privacy. However, it can be modified to work here. We start from the scheme $\mathcal{M}_4$ resulting from our construction, instantiated with an FHE $\mathcal{F}$, and a maliciously circuit private $\mathcal{P}$, where $\mathcal{M}_1 = \mathcal{A}_3 = \mathcal{F}$, and $\mathcal{A}_1 = \mathcal{A}_2 = \mathcal{P}$, reusing keys for the same scheme. Thus, both encrypted

[18] The circuit for *Dec* can be efficiently computed from *out*.

[19] Observe that even our semi-honest privacy requirement combined with perfect correctness implies that the levels of the input encryptions involved are not revealed by the output of Eval.

inputs and encrypted output of $\mathcal{M}_4$ are encryptions under $\mathcal{F}$ and same key. We can thus include encryptions of the bits of $sk_{\mathcal{F}}$ in pk_{MH}. In subsequent executions of Eval using this one as input, one can first decrypt the out_i's using these key bits, and plug the decrypted out_i's into the original scheme $\mathcal{M}_4$ as the c_i's input to Eval. The main caveat is that in our construction the well-formedness of the c_i's fed to a subsequent instance of Eval needs to be certified (under $\mathcal{P}$) as part of the output of the previous Eval. The key observation is that there is no need to certify the well-formedness of the out_i's, but rather only that of the secret key bits used for decryption![20] Moreover, we only need to prove that $sk_{\mathcal{F}}$ constitute valid encryptions of some bits under $pk_{\mathcal{F}}$ specified in $pk_{\mathcal{M}_4}$ (not even correspondence to the $pk_{\mathcal{F}}$ published as part of pk!). This is ok because $sk_{\mathcal{F}}$ is short and independent of the circuit being evaluated. As these are decrypted under the specified key bits in subsequent Eval's, the result would be an enryption of some value independent of the (subsequent) C, which is what we need (as in $\mathcal{M}_4$, if validation fails, nothing is learned about C).

In the next section, we describe the multi-hop construction in full detail. For convenience, here we describe a (variant of) the combination of all 3 steps in a single construction.

A formal construction The schemes $\mathcal{F}, \mathcal{P}$ are a compact FHE, and maliciously circuit private HE respectively. Here $\mathcal{P}$ is for the split input model, and is input-private rather then circuit-private. Although the construction would go through with standard circuit privacy of $\mathcal{P}$, this simplifies the presentation and instantiation of the framework.

Given a leveled FHE $\mathcal{F}$ we define a set of programs $\mathcal{C}_{\mathcal{F}}$ (to be interpreted via U_{SI}) as follows.

1. Let $Dec_{\mathcal{F},k}(sk_{\mathcal{F}}, out_{\mathcal{F}})$ denote the decryption circuit of $\mathcal{F}$ instantiated with security parameter k.
2. Let $Validate_k(pk_{\mathcal{F}}, sk'_{\mathcal{F}}, f_{FK}, r_{FKE}, out_{\mathcal{F}})$ be the circuit computing

$$Validate_k(\ldots) = \begin{cases} out_{\mathcal{F}} & \text{if } (pk_{\mathcal{F}}, sk_{\mathcal{F}}) \in \mathsf{KeyGen}_{\mathcal{F}}(1^k), \text{and} \\ & f_{FKE} \text{ is an encryption of a string of bits (the length of secret keys of } \mathcal{F}) \\ & \text{ with randomness and intermediate values } \overrightarrow{r_{FK}}, \overrightarrow{r_{FKE}} \text{ respectively.} \\ \overrightarrow{0} & \text{otherwise.} \end{cases}$$

   where:
   - $(pk_{\mathcal{F}}, sk_{\mathcal{F}})$ is a purported public-key private-key pair output by $\mathsf{KeyGen}_{\mathcal{F}}(1^k)$. $sk'_{\mathcal{F}} = (sk_{\mathcal{F}}, r_{FK})$ where r_{FK} is the purported random string used in the generation of $(pk_{\mathcal{F}}, sk_{\mathcal{F}})$, along with all the intermediate outputs of gates in the circuit for KeyGen on input r_{FK}.
   - f_{FK} is a purported encryption of sk that corresponds to pk under pk. $\overrightarrow{r_{FKE}}$ is the vector of randomness and intermediate values used by $\mathsf{Enc}_{\mathcal{F}}$.[21]
3. Let $\mathcal{C}_{\mathcal{F}}$ include all pairs of the forms $C = (Validate_k(\ldots), (pk_{\mathcal{F}}, f_{FK}, out_{\mathcal{F}}))$ and $(\mathsf{Dec}_{\mathcal{F},k}(sk_{\mathcal{F}}), out_{\mathcal{F}})$.

Theorem 8. *Assume a compact unleveled FHE scheme $\mathcal{F}$ and a $(U_{SI}, \mathcal{C}_{\mathcal{F}})$-homomorphic maliciously input-private scheme $\mathcal{P}$ exist (both 1-hop). Then the scheme $\mathcal{M}_4$ in Construction 9 a maliciously circuit-private multi-hop compact (unleveled) scheme.*

Construction 9 *Let $\mathcal{F}$,$\mathcal{P}$ be (1-hop) schemes as in Theorem 8. Then, the following construction is a compact, unleveled multi-hop FHE scheme.*

$\mathsf{KeyGen}_{\mathcal{MH}}(1^k)$:

1. *Let* $(pk_{\mathcal{P}}, sk_{\mathcal{P}}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{P}}(1^k)$.
2. $(pk_{\mathcal{F}}, sk_{\mathcal{F}}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathcal{F}}(1^k)$.
3. *Let* $sk'_{\mathcal{F}} = (sk_{\mathcal{F}}, r_{FK})$, *where* r_{KF} *is induced by the randomness used by* $\mathsf{KeyGen}_{\mathcal{F}}$ *in Item 2 as specified in* $Validate$.

[20] If we had to certify them, seemingly, we would need to give a proof on the validity of the execution of Eval, referring to its inputs. It is not clear how to make it short and protect the privacy of that Eval's circuit.

[21] Note that we do not check that the encrypted values correspond to an sk that matches pk.

4. $p_{sk'_{\mathcal{F}}} \xleftarrow{\$} \overrightarrow{\mathsf{Enc}_{\mathcal{P}}}(pk_{\mathcal{P}}, \overrightarrow{sk'_{\mathcal{F}}})$, $f_{sk_{\mathcal{F}}} \xleftarrow{\$} \overrightarrow{\mathsf{Enc}_{\mathcal{P}}}(pk_{\mathcal{F}}, \overrightarrow{sk_{\mathcal{F}}})$.
5. $p_{r_{FKE}} \xleftarrow{\$} \mathsf{Enc}(pk_{\mathcal{P}}, \overrightarrow{r_{FKE}})$, *where* r_{FKE} *is induced by the randomness of* $\mathsf{Enc}_{\mathcal{F}}$ *generating* $f_{sk_{\mathcal{F}}}$, *as specified in* $Validate$.
6. $f_{sk_{\mathcal{P}}} \xleftarrow{\$} \overrightarrow{\mathsf{Enc}_{\mathcal{P}}}(pk_{\mathcal{F}}, \overrightarrow{sk_{\mathcal{P}}})$
7. *Return* $(pk_{\mathcal{MH}}, sk_{\mathcal{MH}}) = ((pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk'_{\mathcal{F}}}, f_{sk_{\mathcal{F}}}, p_{FKE}), (sk_{\mathcal{P}}, sk_{\mathcal{F}}))$.

$\mathsf{Enc}_{\mathcal{MH}}(pk_{\mathcal{MH}} = (pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk_{\mathcal{F}}}), b \in \{0,1\})$**:** *Set* $c = \mathsf{Enc}_{\mathcal{F}}(pk_{\mathcal{F}}, b)$. [22] *Let* $c' = \mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F}})$

$\mathsf{Eval}_{\mathcal{MH}}(1^k, pk_P = (pk_{\mathcal{P}}, pk_{\mathcal{F}}, p_{sk'_{\mathcal{F}}}, f_{sk_{\mathcal{F}}}, f_{sk_{\mathcal{P}}}, p_{r_{FKE}}), C, c_{\mathcal{F}})$**:**

1. *If* C *is syntactically malformed, or* $|x|$ *does not match the number of inputs to* C, *replace* C *with the circuit returning* $x_1 \wedge \overline{x_1}$.
2. $c'_{\mathcal{F}} = \mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, \overrightarrow{\mathsf{Dec}_{\mathcal{F},k|\overrightarrow{c_{\mathcal{F}}}}}, f_{sk_{\mathcal{F}}})$ [23]
3. *Set* $out_{\mathcal{F}} = \mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, C, c'_{\mathcal{F}}))$, $out_{\mathcal{P}} = \mathsf{Eval}_{\mathcal{P}}(pk_{\mathcal{P}}, (\mathsf{Dec}_{\mathcal{F},k}, out_{\mathcal{F}}), p_{sk_{\mathcal{F}}})$
4. *Let* $(C_p, C_s) = (Validate_k, (out_{\mathcal{P}}, pk_{\mathcal{F}}, f_{sk_{\mathcal{F}}}))$
5. *Let* $out_{\mathcal{P}} = \mathsf{Eval}_{\mathcal{P}}(pk_{\mathcal{P}}, (C_p, C_s), p_{sk'_{\mathcal{F}}}, p_{r_{FKE}})$
6. *Output* $out = \mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{P},k|out=out_{\mathcal{P}}}, f_{sk_{\mathcal{P}}})$

$\mathsf{Dec}_{\mathcal{MH}}(sk_{\mathcal{MH}}, out)$**:** *Output* $y = \mathsf{Dec}_{\mathcal{F},k}(sk_{\mathcal{F}}, out)$.

4 Instantiations of the framework

We devise instantiations of schemes $\mathcal{F}, \mathcal{P}$ as required in Theorem 7. As these requirements are strictly stronger then the requirements in Theorem 6, they immediately yield an instantiation of Theorem 6 as well. The component $\mathcal{F}$ has many instantiations from the literature.

For $\mathcal{P}$ we use the following instantiation, induced by the specific construction combining an information theoretic variant of Yao's garbled circuits [IK02] with maliciously circuit-private OT-homomorphic schemes.

Lemma 4. *Assume the existence of circuit-private schemes which are homomorphic for (bit) OT. In particular, the DDH,QR,Paillier or the DCRA assumptions yield such OT schemes [AIR01,HK12]. Then there exists a circuit-private* $(U_{SI}, \mathcal{C})$*-homomorphic scheme* $\mathcal{P}$ *where* $\mathcal{C}$ *consists of all (balanced) formulas. Furthermore,* $\mathcal{P}$ *has decryption circuits* $\mathsf{Dec}_{\mathcal{P},k}(sk, out)$ *of depth* $\mathrm{depth}(C_p)poly(k)$, *where* $out = \mathsf{Eval}_{\mathcal{P}}(C_p, C_s)$.

See Section B.2 for proof of the lemma.

The following corollary of Theorem 7 (6) and Lemma 4 is our working, "take-home", instantiation of the framework.

Corollary 1. *Assume a leveled FHE* $\mathcal{F}$ *with decryption circuits in* NC^1 *exists. Assume further that there exist (bit) OT-homomorphic circuit-private HE* $\mathcal{B}$. *Then there exists a circuit-private compact FHE* $\mathcal{M}_4$. $\mathcal{M}_4$ *is unleveled if* $\mathcal{F}$ *is.*

See section B for proof sketch (almost immediate).

As mentioned above, $\mathcal{F}$ has many "efficient enough" instantiations. Namely, almost all schemes from the literature fit (after an augmentation to achieve perfect correctness, truncating the noise used in encryptions), such as various recent LWE-based constructions [BV11,Bra12].

5 Future work

The "work horse" of our bootstrapping-based transformation 1 for transforming FHE into circuit-private is a circuit-private bit-OT-homomorphic HE. The known constructions from the literature we are aware of can not base circuit-private OT on some assumption the implies FHE (such as LWE, approximate GCD etc.). We do not see good reasons, except for historical ones to why this is the case. Such a construction would give an example of compact FHE which can be made circuit-private without additional assumptions.

[22] The subsequent procedure is meant for making sure that valid level-0 encryptions have the same support as level-i ciphertexts for other $i \geq 1$.

[23] "Sanitize" c. This is the key modification allowing for the multi-hop construction.

References

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

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS*, pages 309–325. ACM, 2012.

BKOI07. Dan Boneh, Eyal Kushilevitz, Rafail Ostrovsky, and William E. Skeith III. Public key encryption that allows pir queries. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 50–67. Springer, 2007.

BLV04. Boaz Barak, Yehuda Lindell, and Salil P. Vadhan. Lower bounds for non-black-box zero knowledge. *Electronic Colloquium on Computational Complexity (ECCC)*, (083), 2004.

Bra12. Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012.

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. Cryptology ePrint Archive, Report 2011/344, 2011. http://eprint.iacr.org/2011/344.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001.

DFH12. Ivan Damgård, Sebastian Faust, and Carmit Hazay. Secure two-party computation with low communication. In Ronald Cramer, editor, *TCC*, volume 7194 of *Lecture Notes in Computer Science*, pages 54–74. Springer, 2012.

DJ01. Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136. Springer, 2001.

Gen09. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. http://crypto.stanford.edu/craig.

GHV10. Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-hop homomorphic encryption and rerandomizable yao circuits. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172. Springer, 2010.

GIKM98. Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. In Jeffrey Scott Vitter, editor, *STOC*, pages 151–160. ACM, 1998.

GM84. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO (1)*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013.

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

IK02. Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials, 2002.

IP07. Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594. Springer, 2007. Full version in http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-info.cgi/2012/PHD/PHD-2012-16, chapter 5.

Lip05. Helger Lipmaa. An oblivious transfer protocol with log-squared communication. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 2005.

NP01. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *SODA*, pages 448–457. ACM/SIAM, 2001.

Rot13. Ron Rothblum. On the circular security of bit-encryption. In *TCC*, pages 579–598, 2013.

vDGHV09. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2009/616, 2009. http://eprint.iacr.org/2009/616.

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167. IEEE Computer Society, 1986.

A Details on framework

A.1 Proof of Theorem 4

Correctness. It is easy to see that the proposed scheme remains correct if all keys and ciphertexts are well-formed. Namely, as all validations pass, $out_{\mathcal{P}}$ encrypts $out_{\mathcal{F}}$ (under $\mathcal{P}$), which in turn encrypts $C(x)$ (under $\mathcal{F}$), which is properly decrypted by perfect correctness of both $\mathcal{P}, \mathcal{F}$.

Efficiency. We run $\mathsf{Eval}_{\mathcal{P}}$ on some $Validate_{k,d,n}$ on input $(pk, sk', \overrightarrow{c_{\mathcal{F}}}, \overrightarrow{r_{\mathcal{F}}}, out_{\mathcal{F}})$. By definition of leveled schemes, this input to $Validate_{k,d,n}$ is of size $m = poly(k, d, n)$. The dependence on d is only because $|pk|$ may depend on d in leveled schemes. Thus, if $\mathcal{P}$ is compact, the encrypted output size $|out_{\mathcal{P}}|$ is $poly(k)$ (a compact scheme). Otherwise, if $\mathcal{F}$ is standard (not leveled), $m = poly(k, n)$, thus encrypted output size is also $poly(k, n)$.

Semantic security. Follows by standard techniques. In particular, the analysis is similar to that of semantic security of *leveled* FHE schemes [Gen09] (as we avoid "cycles" in the graph of encryptions under the various keys).

Circuit privacy. We describe a pair of algorithms $\mathsf{Ext}_l, \mathsf{Sim}_P$, as required.

$\mathsf{Ext}_{\mathcal{M}_3}(1^k, pk^* = (pk^*_{\mathcal{P}}, pk^*_{\mathcal{F}}, p^*_{sk'_{\mathcal{F}}}), c^* = (c^*_{\mathcal{F}}, p^*_{r_{FE}}))$**:**

1. Let $\mathsf{Sim}_{\mathcal{P}}, \mathsf{Ext}_{\mathcal{P}}$ as guaranteed by Definition 5, and let $\overrightarrow{x^*_{\mathcal{P}}} = (r^*_{FE}, sk'^*_{\mathcal{F}}) = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \overrightarrow{p^*_{r_{\mathcal{F}}}}, \overrightarrow{p^*_{sk'_{\mathcal{F}}}})$.
2. If $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) \neq (\mathsf{KeyGen}_{\mathcal{F}}(1^k, 1^d, r^*_k))$ (r^*_k is the randomness implied by r^*_{FK}), return 0. Here, the "secret" parts $r^*_{FK}, r^*_{FE}, sk^*_{\mathcal{F}}$ are taken from $\mathsf{Ext}_{\mathcal{P}}$'s output, and the rest are taken from the input.
3. Otherwise, If $c^*_{\mathcal{F}} = \mathsf{Enc}_{\mathcal{F}}(pk^*_{\mathcal{F}}, r^*_{FE}, b)$ for some bit b, return b.
4. Otherwise, return 0.

$\mathsf{Sim}_{\mathcal{M}_3}(1^k, pk^* = (pk^*_{\mathcal{P}}, pk^*_{\mathcal{F}}, p^*_{sk'_{\mathcal{F}}}), \overrightarrow{c^*} = ((c^*_{F,1}, p^*_{r_{FE,1}}), \ldots, (c^*_{F,n}, p^*_{r_{FE,n}})), b)$**:**

1. Let $\mathsf{Sim}_{\mathcal{P}}, \mathsf{Ext}_{\mathcal{P}}$ as guaranteed by Definition 5, and let $\overrightarrow{x^*_{\mathcal{P}}} = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \overrightarrow{(p^*_{r_{\mathcal{F}}}, p^*_{sk'_{\mathcal{F}}})})$.
2. If the check in $\mathsf{Ext}(1^k, pk^*, c^*_i)$ fails for $(pk^*_{\mathcal{F}}, sk'^*_{\mathcal{F}})$ or for some $i \in [n]$:
   Output $\mathsf{Sim}_{\mathcal{P}}(1^k, pk^*_{\mathcal{P}}, Validate_{k,d,n}, p^*_{sk'_{\mathcal{F}}}, \overrightarrow{p^*_{r_{FE}}}, \overrightarrow{0})$.
3. Otherwise: Compute $out_{\mathcal{P}} = \mathsf{Sim}_{\mathcal{P}}(\mathsf{Dec}_{F,k}, \overrightarrow{pk^*_{\mathcal{P}}}, p^*_{sk_{\mathcal{F}}}, b)$,
   output $out = \mathsf{Sim}_{\mathcal{P}}(1^k, pk^*_{\mathcal{P}}, Validate_{k,d,n}, p^*_{sk'_{\mathcal{F}}}, \overrightarrow{p^*_{r_{FE}}}, out_{\mathcal{P}})$.

We have specified $\mathsf{Ext}_{\mathcal{P}}$ of the proper form, so it remains to analyze $\mathsf{Sim}_{\mathcal{P}}$. Let $x^*_{\mathcal{P}} = (sk'^*_F, r^*_{FE}) = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \mathsf{Dec}_{F,k}, \overrightarrow{(p^*_{sk'_F}, p^*_{r_{FE}})}))$. There are several cases.

- Assume $Validate_{k,d,n}|_{c=c^*_{\mathcal{F}}, pk=pk^*_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{FE}) = 0$. Then by definition $Validate_{k,d,n|c^*_{\mathcal{F}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{FE}) = \overrightarrow{0}$ *for all* $out_{\mathcal{P}}$. By construction of $\mathsf{Sim}_{\mathcal{M}_3}, \mathsf{Ext}_{\mathcal{M}_3}$, this condition is always correctly detected (line 2 in $\mathsf{Sim}_{\mathcal{M}_3}$). By circuit privacy of $\mathcal{P}$, and as the value passed to $\mathsf{Sim}_{\mathcal{P}}$ is $Validate_{k,d,n}(C_s, sk'^*_{\mathcal{F}}, r^*_{\mathcal{F}})$ for $C_s = (c^*_{\mathcal{F}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}})$, which exactly equals $Validate_{k,d,n|c^*_{\mathcal{F}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{\mathcal{F}}) = \overrightarrow{0}$ (as computed in $\mathsf{Eval}_{\mathcal{M}_3}$). Thus, $\mathsf{Sim}_{\mathcal{P}}$ (in line 2) statistically simulates $\mathsf{Eval}_{\mathcal{M}_3}$'s output in this case.
- Otherwise, $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) = \mathsf{KeyGen}(1^k, 1^d, r^*_k)$, and $\overrightarrow{c^*_{\mathcal{F}}} = \overrightarrow{\mathsf{Enc}_{\mathcal{F}}}(pk^*_{\mathcal{F}}, \overrightarrow{x_{\mathcal{F}}}, \overrightarrow{r^*_{\mathcal{F}}})$ for some $\overrightarrow{x_{\mathcal{F}}} \in \{0,1\}^n$. In other words, $pk^*_{\mathcal{F}}, c^*_{\mathcal{F}}$ are a valid public-key, encryption-vector pair. Thus the output of $Validate_{k,d,n}$ in $\mathsf{Eval}_{\mathcal{M}_3}$ is distributed as $\mathsf{Eval}_{\mathcal{F}}(pk_{\mathcal{F}}, C, x_{\mathcal{F}})$. Then by perfect correctness of $\mathcal{F}$, this always decrypts to $C(x_{\mathcal{F}})$. Assume we show that $\overrightarrow{x_{\mathcal{F}}}$ is exactly the vector returned by $\mathsf{Ext}_{\mathcal{M}_3}$. Then, when computing $out_{\mathcal{P}}$ by $\mathsf{Sim}_{\mathcal{M}_3}$, its distribution is as the $\mathsf{Eval}_{\mathcal{M}_3}$, as $b = C(x_{\mathcal{F}})$ is passed to $\mathsf{Sim}_{\mathcal{P}}$ when extracting $out_{\mathcal{P}}$. Consequently, the simulated out is also statistically indistinguishable from out in $\mathsf{Eval}_{\mathcal{M}_3}$, again by validity of $\mathsf{Sim}_{\mathcal{P}}$.

It remains to prove $\overrightarrow{x_{\mathcal{F}}}$ above was indeed extract by $\mathsf{Ext}_{\mathcal{M}_3}$. Let $\overrightarrow{x^*_{\mathcal{F}}}$ denote a vector as extracted in the series of executions of $\mathsf{Ext}_{\mathcal{M}_3}$ on $\overrightarrow{c^*}$. This is so, since $\mathsf{Ext}_{\mathcal{M}_3}$ exits in line 3 on all c^*_i, and it can not return a different bit $x^*_i \neq x_{\mathcal{F},i}$, or we obtain two valid decryptions of some encryption c^*_i, contradicting the correctness $\mathcal{F}$ (Plug the identity function $f(x_1) = x_1$ into Definition 3). Thus, $\mathsf{Sim}_{\mathcal{M}_3}$ receives $b = C(x_{\mathcal{F}})$ as its last input.

A.2 More details on compaction (step 3)

Proof of Theorem 5 The main observation is that Eval computes the output of $\mathsf{Eval}_{\mathcal{P}}$ on the input $pk^*, \overrightarrow{c}^*$, which reveals no "redundant" information, and perform some randomized computation on it (homomorphically decrypt via $\mathcal{F}$), using randomness which is independent of $out_{\mathcal{P}}$. More precisely, we can define:

$\mathsf{Ext}_{\mathcal{M}_4}(1^k, 1^d, pk^*, c^*)$**:** return $x^* = \mathsf{Ext}_{\mathcal{M}_3}(1^k, pk^*, c^*)$.
$\mathsf{Sim}_{\mathcal{M}_4}(1^k, 1^d, pk^* = (pk^*_{\mathcal{M}_3}, pk^*_{\mathcal{A}_3}, s^*_{sk_{\mathcal{P}}}), c^*, b = C(\mathsf{Ext}_{\mathcal{M}_3}(1^k, pk^*, c^*)))$**:**
- Let $out'_{\mathcal{M}_3} = \mathsf{Sim}_{\mathcal{M}_3}(1^k, pk^*, c^*, b)$.
- Output $out'_{\mathcal{A}_3} = \mathsf{Eval}_{\mathcal{A}_3}(1^k, pk^*_{\mathcal{F}}, Dec_{S,k|out=out'_{\mathcal{M}_3}}, \overrightarrow{f_{sk_{\mathcal{M}_3}}})$.

Let $C \in \mathcal{C}$, and set some pk^*, c^*. Conditioned on $out'_{\mathcal{M}_3} = out_{\mathcal{M}_3}$, $out'_{\mathcal{A}_3}$ is distributed *exactly* like $out_{\mathcal{A}_3}$, as both $\mathsf{Sim}_{\mathcal{M}_4}$ and $\mathsf{Eval}_{\mathcal{M}_4}$ run the same process on the same input distribution. Now, $out'_{\mathcal{M}_3}$ is statistically close to $out_{\mathcal{M}_3}$ by circuit-privacy of $\mathcal{M}_3$, so $out_{\mathcal{A}_3}, out'_{\mathcal{A}_3}$ are also statistically close, and validity of $\mathsf{Sim}_{\mathcal{M}_4}$ follows.

Proof of Theorem 7

High level intuition We follow a similar path of combining Theorem 4 with Theorem 5. Here we use the structure of $\mathcal{M}_3$ in Theorem 5, rather then just the existence of $\mathcal{M}_3$. Naturally, we pick a leveled scheme $\mathcal{F}$ in Theorem 4 (and a suitable non-compact $\mathcal{P}$), resulting in a circuit-private scheme $\mathcal{M}_3$. To eliminate the need of assuming circular security, we modify Theorem 5 to use a leveled compact $\mathcal{F}$ as well. The modification is straightforward, by passing the right bound d' on the (decryption) circuit depth to evaluate by $\mathcal{C}$. However, estimating d' poses a technical problem. The encrypted output of $\mathcal{P}$ is of size $m = \mathsf{poly}(k, d, n)$, where d, n is the bound on circuits C the scheme $\mathcal{M}_3$ is to evaluate, and n is the number of variables of the concrete circuit to evaluate. The depth of the decryption circuit $\mathcal{A}_3$ needs to evaluate is then generally some $poly(m)$ as well.

However, n is only known upon Eval, while d' should be estimated at KeyGen! The first observation is that we can bound $n \leq 2^d$, as circuits are assumed to have connected underlying DAGs. Still, this leaves us with circuits of size (and possibly depth) $poly(m) = poly(k, 2^d)$. Thus, $\mathsf{KeyGen}_{\mathcal{A}_3}$ may run in exponential time in d, making $\mathsf{KeyGen}_{\mathcal{M}_3}$ inefficient as well. However, if $\mathsf{Dec}_{\mathcal{P},k,d}$ has efficient enough circuits, we could hope to overcome this caveat, as we indeed manage to.

With a particular instantiation in mind, in Theorem 7, we impose some additional efficiency requirements on both $\mathcal{A}_3, \mathcal{M}_3$, that allows the approach of combining Theorem 4 and Theorem 5 go through.

Filling in the details. We create $\mathcal{M}_4$ by using the output $\mathcal{M}_3$ of Theorem 4 and $\mathcal{F}$ as $\mathcal{M}_3, \mathcal{A}_3$ in Theorem 5 respectively, when augmented to accept a (compact) leveled $\mathcal{A}_3$. As explained before, the main technical difficulty arises in $\mathsf{KeyGen}_{\mathcal{M}_4}$, when bounding d' passed to $\mathsf{KeyGen}_{\mathcal{A}_3}$ by some $poly(k, d)$. By construction, the inputs to both $\mathsf{Dec}_{\mathcal{F},k}$ and $Validate_{k,d,n}$ is of size $m = poly(k, n, d)$ (for some global polynomial independent of d). As $Validate_{k,d,n}$ is in NC^1, and $\mathsf{Dec}_{\mathcal{F},k}$ is assumed to be as well, the depth of each is bounded by $c \log m$ for some (global) constant c. By construction, the decryption algorithm for $\mathcal{P}$ is the decryption algorithm for $\mathcal{M}_3$ (recall that the circuit is computed based on out). By assumption on $\mathcal{M}_3$, the size of this (one such) circuit is $poly(d)$, for $d' = depth(Validate_{k,d,n}, Dec_{\mathcal{F},k})$. Since they are both in NC^1, $d' = poly(c \log m, k)$, where c is a constant known to $\mathsf{KeyGen}_{\mathcal{M}_4}$ (depending on the concrete $\mathcal{M}_3, \mathcal{A}_3$). This is in turn $\leq poly(d, k)$, since $n \leq 2^d$.

Another minor issue to note is that the resulting scheme's decryption algorithm is indeed independent of d, since it applies $\mathsf{Dec}_{\mathcal{F},k}$ (on an output of $\mathsf{Eval}_{\mathcal{F}}$), where $\mathcal{F}$ is a compact leveled scheme.

A.3 Proof of Theorem 8

Correctness. Given perfectly correct $\mathcal{F}, \mathcal{P}$, perfect correctness of $\mathcal{MH}$ follows by simple induction. That is, by construction an application of Eval_{pk} on C and a well-formed ciphertext vector $\overrightarrow{c}$ encrypting $\overrightarrow{b}$ under pk results in c' that decrypts to $C(b)$ with probability 1 (in particular, all validations pass).

Efficiency. The output of the scheme is an output on an $\mathsf{Eval}_{\mathcal{F}}$, which is a compact scheme, thus it is of size $poly(k)$.

Semantic security. Follows by standard techniques from semantic security of $\mathcal{F}, \mathcal{P}$. We also need to assume weak circular security of $\mathcal{F}, \mathcal{P}$, as we are reusing keys (unlike in our leveled constructions).

Circuit privacy. We describe a pair of algorithms $\mathsf{Ext}_{\mathcal{MH}}, \mathsf{Sim}_{\mathcal{MH}}$, as required.

$\mathsf{Ext}_{\mathcal{MH}}(1^k, pk^* = (pk^*_{\mathcal{P}}, pk^*_{\mathcal{F}}, p^*_{sk'_{\mathcal{F}}}, f^*_{sk_{\mathcal{F}}}, f^*_{sk_{\mathcal{P}}}), c^* = c^*_{\mathcal{F}})$:

1. Let $\mathsf{Sim}_{\mathcal{P}}, \mathsf{Ext}_{\mathcal{P}}$ as guaranteed by Definition 5, and let $\overrightarrow{x^*_{\mathcal{F}}} = (r^*_{FKE}, sk'^*_{\mathcal{F}}) = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \overrightarrow{p^*_{r_{FKE}}, p^*_{sk'_{\mathcal{F}}}})$.
2. (a) If $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}})$ is not generated by $(\mathsf{KeyGen}_{\mathcal{F}}(1^k))$ with randomness and intermediate values as specified by $r^*_{sk'_{\mathcal{F}}}$, return 0. Here and in the following the "secret" parts $(r^*_{FKE}, sk'^*_{\mathcal{F}})$ are taken from $\mathsf{Ext}_{\mathcal{P}}$'s output, and the rest are taken from the input.
   (b) Otherwise, If $f^*_{sk_{\mathcal{F}}}$ is not generated by $\overrightarrow{\mathsf{Enc}_{\mathcal{F}}}(pk^*_{\mathcal{F}}, \overrightarrow{b})$ with randomness and intermediate values as specified by r^*_{FKE}, return 0.
3. Otherwise, return $b = Dec_{\mathcal{F},k}(\mathsf{Eval}_{\mathcal{F}}(pk^*_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{F},k|c=c^*_{\mathcal{F}}}, f^*_{sk_{\mathcal{F}}}), sk^*_{\mathcal{F}})$.

$\mathsf{Sim}_{\mathcal{MH}}(1^k, pk^* = (pk^*_{\mathcal{P}}, pk^*_{\mathcal{F}}, p^*_{sk'_{\mathcal{F}}}, f^*_{sk_{\mathcal{F}}}, f^*_{sk_{\mathcal{P}}}), \overrightarrow{c^*} = (c^*_{\mathcal{F},1}, \ldots, c^*_{\mathcal{F},n}), b)$:

1. Let $\mathsf{Sim}_{\mathcal{P}}, \mathsf{Ext}_{\mathcal{P}}$ as guaranteed by Definition 5, and let $x^*_{\mathcal{P}} = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \overrightarrow{p^*_{r_{FKE}}, p^*_{sk'_{\mathcal{F}}}})$.
2. If the check in $\mathsf{Ext}_{\mathcal{F}}(1^k, pk^*, c^*_i)$, 2.a or 2.b fails for some $i \in [n]$:
   (a) Let $out_{\mathcal{P}} = \overrightarrow{\mathsf{Sim}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, Validate_{k,n}, (p^*_{sk'_{\mathcal{F}}}, p^*_{r_{FKE}}), \overrightarrow{0})$.
   (b) Otherwise: let $out_{\mathcal{P}} = \mathsf{Sim}_{\mathcal{P}}(\mathsf{Dec}_{\mathcal{F},k}, pk^*_{\mathcal{P}}, p^*_{sk_{\mathcal{F}}}, b)$
   then set $out'_{\mathcal{P}} = \overrightarrow{\mathsf{Sim}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, Validate_{k,n}, (p^*_{sk'_{\mathcal{F}}}, p^*_{r_{FKE}}), \overrightarrow{out_{\mathcal{P}}})$.
3. Output $\mathsf{Eval}_{\mathcal{F}}(pk^*_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{P},k|out=out'_{\mathcal{P}}}, f^*_{sk_{\mathcal{P}}})$

We have specified $\mathsf{Ext}_{\mathcal{MH}}$ of the proper form. It remains to analyze $\mathsf{Sim}_{\mathcal{MH}}$. Let $x^*_{\mathcal{P}} = (sk'^*_{\mathcal{F}}, r^*_{FKE}) = \overrightarrow{\mathsf{Ext}_{\mathcal{P}}}(1^k, pk^*_{\mathcal{P}}, \mathsf{Dec}_{\mathcal{F},k}, \overrightarrow{(p^*_{sk'_{\mathcal{F}}}, p^*_{r_{FKE}})}))$. There are several cases.

– Assume $Validate_{k,n}|_{f_{sk_{\mathcal{F}}}=f^*_{sk_{\mathcal{F}}}, pk=pk^*_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{FKE}) = 0$. Then by definition

$$Validate_{k,n|f^*_{sk_{\mathcal{F}}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{FKE}) = \overrightarrow{0} \text{ for all } out_{\mathcal{F}}$$

By construction of $\mathsf{Sim}_{\mathcal{MH}}, \mathsf{Ext}_{\mathcal{MH}}$, this condition is always correctly detected (Item 2.a in $\mathsf{Sim}_{\mathcal{P}}$). By circuit privacy of $\mathcal{P}$, and as the value passed to $\mathsf{Sim}_{\mathcal{P}}$ is $Validate_{k,n}(C_s, sk'^*_{\mathcal{F}}, r^*_{FKE})$ for $C_s = (f^*_{sk_{\mathcal{F}}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}})$, which exactly equals $Validate_{k,n|f^*_{sk_{\mathcal{F}}}, pk^*_{\mathcal{F}}, out_{\mathcal{F}}}(sk'^*_{\mathcal{F}}, r^*_{FKE}) = \overrightarrow{0}$ (as computed in $\mathsf{Eval}_{\mathcal{MH}}$). Finally, we apply exactly the same procedure $\mathsf{Eval}_{\mathcal{F}}(pk^*_{\mathcal{F}}, \mathsf{Dec}_{\mathcal{P}}, \ldots)$, with the same inputs taken from $pk^*_{\mathcal{MH}}$. That is, we apply the same function to negligibly-far random variables $out_{\mathcal{F}}$ (real and simulated), so the distance remains negligible (can not grow). Thus, $\mathsf{Eval}_{\mathcal{F}}$ (in line 3) statistically simulates $\mathsf{Eval}_{\mathcal{MH}}$'s output in this case.

– Otherwise, $(pk^*_{\mathcal{F}}, sk^*_{\mathcal{F}}) = \mathsf{KeyGen}(1^k; r^*_{FKE})$, and $\overrightarrow{f^*_{sk_{\mathcal{F}}}} = \overrightarrow{\mathsf{Enc}_{\mathcal{F}}}(pk^*_{\mathcal{F}}, \overrightarrow{sk_{\mathcal{F}}}; \overrightarrow{r^*_{FKE}})$ for some $x^*_{\mathcal{F}} \in \{0,1\}^h$, for $\{0,1\}^h$ being the domain of secret keys generated by $\mathsf{KeyGen}_{\mathcal{F},k}$.In other words, $f^*_{sk_{\mathcal{F}}}$ is a well-formed encryption vector under $pk^*_{\mathcal{F}}$ (same for all coordinates (not necessarily a secret key corresponding to $pk^*_{\mathcal{F}}$, and even if so, not necessarily the one encrypted in $p^*_{r_{KE}}$).

  Assume we show that $\overrightarrow{b} = \overrightarrow{\mathsf{Dec}_{\mathcal{F},k|\vec{c^*}}}(x^*_{\mathcal{F}})$ is exactly the vector returned by $\mathsf{Ext}_{\mathcal{P}}$. Then, when computing $out'_{\mathcal{F}}$ by $\mathsf{Sim}_{\mathcal{P}}$, its distribution is as the $\mathsf{Eval}_{\mathcal{MH}}$, as $y = C(x^*_{\mathcal{F}})$ is passed to $\mathsf{Sim}_{\mathcal{MH}}$ as last input (bit), and used in line 2.b. Consequently, $out'_{\mathcal{F}}$ is also statistically indistinguishable from out in $\mathsf{Eval}_{\mathcal{MH}}$, again by validity of $\mathsf{Sim}_{\mathcal{P}}$. Finally, as the same transformation is applied to the output of $Validate_k$ both in $\mathsf{Eval}_{\mathcal{MH}}$, and in the simulation, the validity of $\mathsf{Sim}_{\mathcal{MH}}$ would follow.

  It remains to prove that $\overrightarrow{b} = \overrightarrow{\mathsf{Dec}_{\mathcal{F},k|\vec{c^*}}}(x^*_{\mathcal{F}})$ as above is indeed extracted by $\overrightarrow{\mathsf{Ext}_{\mathcal{P}}}$. Let $\overrightarrow{b^*_{\mathcal{F}}}$ denote a vector as extracted in the series of executions of $\mathsf{Ext}_{\mathcal{P}}$ on $\vec{c^*}$. This is so, since $\mathsf{Ext}_{\mathcal{MH}}$ exits in line 3 on all c^*_i (all checks pass). Recall that $f^*_{sk_{\mathcal{P}}}$ is a well-formed encryption, under the well-formed key $pk^*_{\mathcal{F}}$ (as follows from passing both tests in 2). Thus, applying $\mathsf{Dec}_{\mathcal{F},k}$ in line 3 will result in the corresponding b_i bit used in $\mathsf{Eval}_{\mathcal{MH}}$, by perfect correctness of $\mathcal{F}$.

B Details on instantiations of the framework.

Our "take-home", working version of our framework is given in Corollary 1. It can transform almost any FHE from the literature into circuit-private FHE with only mild additional assumptions (namely, circuit-private bit-OT-homomorphic HE). In fact, most instantiations of FHE nowadays have decryption circuits in NC^1. For instance, LWE-based constructions, such as [BV11] have decryption based on linear algebra over $\mathbb{Z}_p$ for large integers p, and verifying some $c \mod p$ (noise) is not too high. These operations are typically in NC^1, allowing various instantiations of $\mathcal{F}$. FHE schemes from the "first generation", obtained by using Gentry's original blueprint, applying "squashing" and then bootstrapping to a somewhat homomorphic scheme, are also good candidates for applying Theorem 1. In particular, squashing of somewhat homomorphic encryption using the sparse subset sum assumption does not compromise efficiency of decryption. See the following section for a detailed efficiency analysis of decryption of the (bootstrappable) DGHV scheme. In fact, we show that the scheme has validation circuits in NC^1 as well. Although not required for our construction, this results in improved concrete efficiency, allowing to supply only the randomness of $\mathsf{Enc}_{\mathcal{F}}$ as validity witness in Construction 7. In the general case, if validation is carried by a circuit of size $poly(k)$, then encryption size blows up by an additive factor of $poly(k)$. In fact, this optimization is in fact possible for instantiations with all known FHE's from the literature, which have validation circuits in NC^1.

By Lemma 4, the existence of $\mathcal{B}$ implies the existence of $\mathcal{F}$ as required in Theorem **??**. The lemma also gives several possibly instantiations of $\mathcal{B}$ from the literature.

Proof Sketch of Corollary 1. The only non-trivial point that remains to observe here is that $\mathcal{M}_4$ satisfies the efficiency requirements of Theorem 4. $\mathcal{F}$ (as implied by the existence of $\mathcal{B}$) is $\mathcal{C}$-homomorphic for $\mathcal{C}$ consisting of all (balanced) formulas. Since $Validate_{k,d,n}, Dec_{\mathcal{F},k} \in \mathsf{NC}^1$, both have formulas of size polynomial in their input, and are balanced (c is as required by our representation of formulas). As the inputs to both are in turn polynomial in $\mathsf{Eval}_{\mathcal{M}_4}$'s input (in fact, it depends on $pk, \vec{c}$), the corollary follows.

B.1 Setting $\mathcal{F}$ = DGHV in Corollary 1

Lemma 5. *Consider the compact leveled FHE $\mathcal{F}$ obtained by applying the bootstrapping theorem 3 to [vDGHV09]'s bootstrappable somewhat homomorphic scheme (the variant without privacy). It is semantically secure assuming the approximate GCD and succinct subset sum assumptions hold. For this scheme, the function family $\mathcal{C}_{\mathcal{F}}$ induced by $\mathcal{F}$ is in NC^1.*

Proof. In the following $\lfloor v \rceil$ denotes the closest integer to v, and $a \mod b$ denotes $a - \lfloor a/b \rceil b$ (which falls within $(-b/2, b/2]$).

For completeness, we present DGHV's bootstrappable scheme (after squashing [vDGHV09, Section 6]). For simplicity we describe the slightly simplified variant where $p \mid v_0$ [vDGHV09, Section 3.3.2]; we could have used the original scheme as well. (We do not use their re-randomized semi-honestly circuit-private variant from [vDGHV09, Appendix C].)

Construction 10

$\mathsf{KeyGen}_{DGHV}(1^k)$**:**

1. *Let* $p \xleftarrow{\$} (2\mathbb{Z}+1) \cap [2^{\eta-1}, 2^\eta)$*;* $r_0 = 0$*;* $(r_1, \ldots, r_\tau) \xleftarrow{\$} \mathbb{Z} \cap [-2^\rho, 2^\rho]^\tau$*.*
   $q_0, \ldots, q_\tau \xleftarrow{\$} [0, 2^\gamma)^\tau$*; relabel so that* q_0 *is the largest. Set* $v_i = p\lfloor z_i/p \rceil + 2r_i$*.*
   Restart unless v_0 *is odd.*
2. *Pick a random vector* $\vec{s} \in \{0,1\}^\Theta$ *of Hamming weight* θ*. For* $i \in [\Theta]$ *choose at random integers* $u_i \in \mathbb{Z} \cap [0, 2^{\kappa+1})$ *subject to the condition that* $\sum_{i=1}^{\theta} s_i u_i = \lfloor 2^\kappa/p \rceil \mod 2^{\kappa+1}$*.*
   Let $\vec{w} = (u_1/2^\kappa, \ldots, u_\Theta/2^\kappa)$*, where the* w_i*'s are computed with precision* κ *bits.*
3. *Output* $(pk, sk) = ((\vec{v}, \vec{w}), (p, \vec{s}))$*.*

$\mathsf{Enc}_{DGHV}(pk, b)$**:** $r \xleftarrow{\$} \mathbb{Z} \cap (-2^{\rho'}, 2^\rho)$*;* $s \xleftarrow{\$} \{0,1\}^\tau$ *; set* $c' = (b + 2r + 2\sum_i s_i v_i) \mod v_0$*.*
For $i \in [\Theta]$*, set* $z_i = (c' \cdot w_i) \mod 2$*, keeping only* $\lceil \log \theta \rceil + 3$ *bits of precision after the binary point. Output* $c = (c', \vec{z})$*.*

$\mathsf{Eval}_{DGHV}(pk, C, (c_1, \ldots, c_n))$**:** *Proceed from the bottom up, labeling input wires by the corresponding* c_i*'s; output wires of XOR gates by* $(a+b) \mod v_0$*, of AND gates by* $(a \wedge b) \mod v_0$*, and of NOT gates by* $(a+1) \mod v_0$ *where* a, b *are labels of the input wires to that gate.*
Let out'*, denote the label of the output wire. Compute a vector* $\vec{z}$ *from* $out', \vec{w}$ *as in* Enc*, and output* $out = (out', \vec{z})$*.*

$\mathsf{Dec}_{DGHV}(sk = (p, \vec{s}), out = (out', \vec{z}))$**:** *Output* $out' - \lfloor \sum_i s_i z_i \rceil \mod 2$*.*

The parameters are set as in [vDGHV09] (and for the same reasons) in the following way.

- $\rho = \omega(\log k)$*;* $\rho' = \rho + \omega(\log k)$*.*
- $\eta \geq \rho \cdot \Theta(k \log^2 k)$*.*
- $\gamma = \omega(\eta^2 \log k)$*.*
- $\tau = \gamma + \omega(k)$*.*
- $\kappa = \gamma\eta/\rho', \theta = k, \Theta = \omega(\kappa \log k)$*: squashing related parameters.*

In [vDGHV09], the authors prove that the scheme is indeed bootstrappable. Let $\mathcal{DGHV}_{\mathcal{BS}}$ be a leveled scheme obtained by applying the transformation in Theorem 3 to the above scheme. We prove that the relevant $\{Validate_{k,d,n}(pk_{\mathcal{F}}, \vec{c_{\mathcal{F}}}, sk'_{\mathcal{F}}, \vec{r_{\mathcal{F}}})\}_{k,d,n}$, and $\{\mathsf{Dec}_k(out, sk_{\mathcal{F}})\}_k$ function families are in NC^1.

Construction validity proof.

- Let us first understand the complexity of relevant operations in *DGHV* (Construction 10). Given a purported key pair $(pk = (\vec{v}, \vec{w}), sk = (p, s))$, it suffices to check:
  - parity of p, v_0.
  - Check that $p \mid v_0$, and that $v_i \mod p$ is not too high for all i.
  - Verify the u_i's against p.

  To verify encryption of a bit $(out, \vec{w})$, given randomness $\vec{w}, r$ verifying $out = \sum_i v_i w_i + 2r + b$, and that $\vec{z} = out' \cdot \vec{u}$ with the right precision again involves only integer arithmetics, and is in NC^1. Similarly, decryption involves integer arithmetics in NC^1.

– By the bootstrapping construction, $\mathcal{DGHV}_{\mathcal{BS}}$'s public key has $O(d)$ purported public-key private-key pairs of $DGHV$ to be verified. The private keys are sk_i's corresponding to the pk_i's are specified as part of the randomness $\overrightarrow{r_k}$ (except for sk_ℓ, which is included as the private key). One thing to check is that the (pk_i, sk_i) pairs are valid $DGHV$ keys. As noted before, there are $O(\log \cdot)$ formulas for this task. Then we need to check that each e_i is in $\overrightarrow{\mathsf{Enc}}(pk_i, \overrightarrow{sk_{i-1}})$. We accomplish this using the parpoted randomness for each encryption (part of r_k) to check validity of encryptions. Then we decrypt using sk_i to obtain sk'_i, and check that $sk_i = sk'_i$. As noted above, each of these individual operations (key validation, decryption and encryption validation for $DGHV$) is in NC^1. There are $poly(k) \cdot (d + n)$ such conditions to verify, that is, a conjunction of the conditions is to be computed. The input size to each individual check is $poly(lambda)$. Performing the conjunction using a tree of $\wedge$'s, the depth of the formula is $O(k + \log n + \log d)$ which is $O(\log \cdot)$ depth in its input size.

B.2 Proof of Lemma 4

Let $\mathcal{OT} = (\mathsf{KeyGen}_{OT}, \mathsf{Enc}_{OT}, \mathsf{Eval}_{OT}, \mathsf{Dec}_{OT})$ denote a bit-OT homomorphic scheme. That is, "programs" are bit pairs (s_0, s_1) and the input is always a single bit x_1. $OT((s_0, s_1), b) = s_b$. Such schemes can be instantiated using one of the following results [AIR01,HK12,IP07]. Fix such a scheme.

We define our scheme using an information theoretic version of Yao's garbled circuits as in [IK02].

Theorem 9 (information theoretic Yao). *Let $\mathcal{C}$ denote the set of formulas (recall that wlog. formulas satisfy $\mathrm{depth}(C) \leq D \log(\mathrm{size}(C))$, for some global constant D). For each $C(x_1, \ldots, x_n) \in \mathcal{C}$, there exists a randomized, efficiently computable, functionality $p_C(\overrightarrow{x}, \overrightarrow{r}) : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^t$ of the form $p_C = (p_1(x_1, r), \ldots, p_n(x_n, r), p_{n+1}(r) = (mask, rest))$, where each $p_i(\cdot)$ outputs strings of length t_i, and $mask \in \{0,1\}$. It satisfies:*

1. *There exists a uniform efficient algorithm A, such that $A(C) = p_C$. In particular, $m, t \leq q(\mathrm{size}(C))$ for a global polynomial q.*
2. *Privacy: For a random $r \in \{0,1\}^m$, the distribution $p_C(x, r)$ depends only on $C(x)$. More precisely, for every C (of proper depth) there exists an (efficient) simulator Sim_C, such that for all $\overrightarrow{x}$, $\mathsf{Sim}_C(C(x))$ is distributed identically to $p_C(x)$. Moreover, all but joint distribution of all but mask is independent of x.*
3. *Correctness: There exists a decoding circuit Dec_C of depth $O(\mathrm{depth}(C))$ that for all $x \in \{0,1\}^n, r \in \{0,1\}^m$, outputs $f(x)$ given $p_C(x, r)$.*

(Such a functionality is referred in the literature as a "randomized encoding" of the function f_C).

Proof sketch. The statement and proof of this theorem already appear (in a slightly different form) in [IK02]. The only aspect that is not explicitly mentioned is the decryption complexity of the randomized functionality p_C. Roughly, evaluating p_C involves an evaluation of a garbled circuit for C gate-by gate, to compute a relevant output key, as in "standard" computational Yao [Yao86]. More precisely, every input wire is assigned a key-index pair (k, c) for every input value, where both are just portions of the randomness r. The perform the evaluation, one learns the pair (k, c) corresponding to the value of x_i for each input variable x_i (one of two options). Now, every gate g ($g \in \{\oplus, \wedge\}$) has a table of four strings of the form (k_{c_1,c_2}, c), indexed by bits (c_1, c_2). Entry (c_1, c_2) is encrypted via the keys from (k_i, c_1),(k_j, c_2) corresponding to its input wires i, j. The encryption here is simply bitwise XOR with $k_i \oplus k_j$. To complete the decryption, the final gate includes a value r_g, so that $c \oplus r_g$ is the output.

Thus, having keys corresponding to the input wires, choosing and decrypting the right entry in a gate's table is a degree-3 vector of polynomials in the $c_i, c_j, k_{i,c_1}, k_{j,c_2}$ and g's plain $(k_{g,\cdot,\cdot})$ (decryption is done by XORing once more). Overall, the resulting depth of the decryption circuit is $O(\mathrm{depth}(C))$.

Thus, we suggest the following scheme for $(U_{SI}, \mathcal{C})$ where $\mathcal{C}$ includes formulas of depth $\leq D \log(\mathrm{size}(C))$ (D is a constant as in theorem 9).

Construction 11

$\mathsf{KeyGen}_P(1^k)$: *Let* $(pk_{OT}, sk_{OT}) \xleftarrow{\$} \mathsf{KeyGen}_{OT}(1^k)$.
$\mathsf{Enc}_P(pk, b)$: *output* $\mathsf{Enc}_{OT}(pk_{OT}, b)$.
$\mathsf{Eval}_P(pk, (C_p(C_s, x), \overrightarrow{C_s}), \overrightarrow{c})$: *Denote* $\ell = |C_s| + |x|$.
Let $p_{C_p}(x, r) = p_1, \ldots, p_{\ell+1}$ *be as in Theorem 9.*
- *Let* $r \xleftarrow{\$} \{0,1\}^m$, *(*$m$ *is as in* p_{C_p}*).*
- *For* $i \leq |C_s|$, *set* $out_i = p_i(C_{s,i}, r)$.
- *For* $|C_s| + 1 \leq i \leq \ell$*:*
  - *Let* $v_0 = p_i(0, r), v_1 = p_i(1, r)$, *both of length* t_i.
  - *Set* $\overrightarrow{out_i} = \overrightarrow{\mathsf{Eval}_{OT}}(pk_{OT}, (\overrightarrow{v_0; v_1}), c_i)$.
- *Let* $v_r = p_{\ell+1}(r)$.
- *Output* $(C_p, \overrightarrow{out}, v_r)$).

$\mathsf{Dec}_P(sk_{OT}, (C_p, \overrightarrow{out}, v_r))$: *For* $i \leq |C_s|$, *recover a key* $k_i = out_i$; *for* $|C_s| + 1 \leq i \leq \ell$, *recover a key* $k_i = \mathsf{Dec}_{OT}(sk_{OT}, out_i)$. *Let* $\mathsf{Dec}_{C_p,k}(\cdot)$ *be a circuit that recovers the output bit from* $p_{C_p}(x, r)$. *Output* $\mathsf{Dec}_{C_p,k}(k_1, k_2, \ldots, k_\ell, v_r)$.

Theorem 10. *Assume Construction 11 is instantiated with a circuit-private bit OT-homomorphic scheme* $\mathcal{OT}$. *Then the scheme in Construction 11 is* $(U_{SI}, \mathcal{C})$*-homomorphic and input-private for* $\mathcal{C}$ *consisting of formulas (of depth* $\leq D \log |C|$*) where* D *is the global constant in our definition of the formulas representation model). The depth of its encrypted circuits is* $poly(k, \log(\text{size}(C)))$.

As to decryption circuit complexity of the functions Dec is indeed bounded by $poly(k, \log \text{size}(C))$, since the circuit constructed runs the OT decryption circuit on the x_i's in parallel, and then runs Yao's decryption on v_r and the resulting (k_i, c_i) pairs. This computation has logarithmic depth in $|C|$.[24]

Circuit privacy of Construction 11. The extractor $\mathsf{Ext}_P(1^k, pk^*, \overrightarrow{c}^*)$ outputs $\overrightarrow{x} = \overrightarrow{\mathsf{Ext}_{OT}}(1^k, \overrightarrow{c}^*)$. A simulator $\mathsf{Sim}_P(1^k, C_p(z, x), pk^*, \overrightarrow{c}, out = C_p(C_s, x^*))$, where x^* is the output of $\mathsf{Ext}(1^k, pk^*, \overrightarrow{c^*})$ proceeds by:

1. Let $\ell = |(z, x)|$. Sample $(p_1, \ldots, p_\ell, p_{rest})$ at random according to the partial distribution of p_{C_p} with mask omitted (recall this distribution is independent of x).
2. For $i \leq |z|$ let $out_i = p_i$.
3. For $|z| + 1 \leq i \leq \ell$, let $out_i = \overrightarrow{\mathsf{Sim}_{OT}}(1^k, \overrightarrow{p_i}, c^*{}_i))$.
4. Set $mask$ so that $(p_1, \ldots, p_\ell, (mask, p_{rest})$ decrypts to out. Set $p_{\ell+1} = (mask, p_{rest})$. Output $(out_1, \ldots, out_\ell, p_{\ell+1})$.

We now show that the above simulation is a perfect one. First observe that the sampling in item 1 is a perfect simulation since $p^{(-)}_{C_p} = (p_1, \ldots, p_\ell, p_{rest})$ is independent of C_p's input. Also, the recovery of $mask$ in item 4 results in a perfect simulation of $p_{C_p}(out, r)$, as it is uniquely determined by $p^{(-)}_{C_p}$ by correctness of p_{C_p}, and the fact that $p^{(-)}$ is independent of x.

Now, consider the distribution of out in Eval_P. By circuit-privacy of $\mathcal{E}_{OT}$, each OT call corresponding to some x_i determines a single query bit b_i. Thus, sampling $p_{C_p}(C_s, x)$, where p_i for $i \leq |z|$ corresponds to input bit $C_{s,i}$, and to x_i for $|z| + 1 \leq i \leq \ell$, Eval_P's output on pk^*, out^* is statistically indistinguishable from the following "hybrid" distribution (where OT replies are simulated using the bits queried for, but the bits "not asked for" are not given to the simulator):

- Let $x^* = \overrightarrow{\mathsf{Ext}_{OT}}(pk^*_{OT}, \overrightarrow{c^*})$.
- Sample $p_{C_p}(\overrightarrow{C_s}, x, r)$ to obtain $p_1, \ldots, p_\ell, p_{\ell+1}$.

[24] That is, the decryption algorithm computes the decryption circuit that extracts the encrypted output bit somewhat indirectly. It does not plug C_p into a universal Yao decryption circuit, but rather preprocesses it to be "directly embedded" into the circuit's structure, using C_p included in the encrypted output. Although a universal Yao decryptor of the proper complexity exists, this construction is more straightforward. The simplicity advantage is obtained since the decryption circuit can be precomputed using the encrypted output in polynomial time, without the extra efficiency requirements.

– The output of Eval is statistically close to

$$(p_1, \ldots, p_{|z|}, \overrightarrow{\mathsf{Sim}_{OT}}(pk^*_{OT}, \overrightarrow{p_{|z|+1}}, c^*_{|z|+1}, \ldots, p_{\ell+1}(r)).$$

By the privacy property in Theorem 9, the simulation of $p_{C_p}(C_s, x)$ in Sim_P is perfect. Thus, the simulated OT replies in Sim_P are statistically indistinguishable from the ones in the hybrid distribution, and the result follows (through comparing the real and simulated distributions to the hybrid distribution).