Maliciously Circuit-Private FHE

Rafail Ostrovsky[*] Anat Paskin-Cherniavsky[†] Beni Paskin-Cherniavsky[‡]

May 21, 2013

Abstract

We present a framework for constructing compact FHE (fully homomorphic encryption) which is circuit-private in the malicious setting. 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 $\mathcal{C}$, using a 2-message protocol. Maliciously circuit-private FHE immediately gives rise to such a protocol which is secure against a malicious receiver.

Keywords: Fully homomorphic encryption, computing on encrypted data, malicious setting.

[*]Department of Computer Science and Mathematics, UCLA, Email: rafail@cs.ucla.edu. Research supported in part by NSF grants CNS-1118126; CNS-1136174; and 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 authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

[†]Department of Computer Science, University of California, Los Angeles. E-mail: anpc@cs.ucla.edu The work is supported in part 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 authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

[‡]E-mail: cben@users.sf.net

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 HE scheme for a smaller class of circuts which is 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 $poly(\lambda, n, o(|C|))$, where λ is a security parameter, ideally independent of $|C|$ altogether. This application of homomorphic encryption was studied both in several works predating Gentry's first fully homomorphic scheme [IP07, BKOI07], and mentioned in [Gen09], and termed *computing on encrypted data*.

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 (but not any form of correctness against a malicious sender). The circuit privacy of the scheme translates into a privacy guarantee against malicious receiver for that protocol. While standard FHE (without extra requirements) does not imply any security against malicious receiver, the semi-honestly circuit-private schemes from (e.g.) [vDGHV09] implies standard simulation-based security against semi-honest receivers. Thus, such a scheme induces a protocol which is private against malicious corruptions in the stand-alone model.

Back to maliciously circuit-private FHE, we say a scheme is circuit-private if it satisfies the following privacy notion ala [IP07]:

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

An FHE satisfying Definition 1 induces a protocol private against a malicious sender as before, but private against a malicious receiver with unbounded simulation (as usual, no correctness guarantees against a malicious sender are given).

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

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 [AIR01, Kal05, IP07] as a plausible relaxation.

It is important to note that we only consider the plain model. If preprocessing, such as CRS was allowed, then Enc could have added a NIZK proving that the key is well-formed, and that the ciphertext is a valid ciphertext under that public key. However, this straight-forward approach does not provide an immediate solution, as it does not guarantee the public key (for example) has the proper distribution. In particular, the (ad-hoc) DGHV [vDGHV09] scheme variant which is circuit-private in the semi-honest setting (where (pk, c) is generated as prescribed by the scheme), needs that pk has the proper distribution ($\mathsf{KeyGen}(1^k)$), rather than just being in the support of that distribution.

1.1 Previous Work

Two-message two party protocols in a setting of computing on encrypted data, robust against malicious receivers, can be often interpreted as circuit-private (not necessarily compact) HE for that class of circuits. This is roughly by viewing the receivers' randomness as the secret key, round-1 message as an encryption for his input, and the senders' reply as Eval's output (also, the encryption is not necessarily applied bit-by-bit, as done in contemporary definitions of FHE). Such circuit-private (non-compact) FHE can be obtained by combining an information-theoretic version of Yao's garbled circuits with oblivious transfer (OT) secure against malicious receivers [AIR01, Kal05, IP07]. As information-theoretic Yao is only efficient for NC^1, the resulting HE scheme captures only functions in NC^1. Also, this construction has encrypted output size at least $|C|$, while the crux of HE is having compact encrypted outputs. Ideally polynomial only in the security parameter λ. Ciphertext size $poly(\lambda, n)$ (yet independent of $|C|$) is also reasonable in the setting of computing on encrypted data, since the receiver is at least linear in its input size n. 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 $poly(\lambda, n, \log|f|)$, where $|f|$ is the size of f's circuit representation [DFH12]. The protocol is UC-secure against malicious parties under a new type of assumption they call "extractable" collision resistant hash functions along with some additional (more standard) assumptions. Instantiating f with the universal function for evaluating circuits, would have resulted 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.

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 ciphertext 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, appendix C] for the semi-honest setting is adding a Yao circuit for decryption, and homomorphically performing the decryption operation during Eval, thereby transforming any scheme into a semi-honestly circuit-private one (as the decryption circuit is of size $poly(\lambda)$, the lack of compactness of Yao-based schemes is not an issue). As mentioned before, the malicious setting (with compact encrypted outputs) has been addressed in the context of OT [AIR01, Kal05]. For broader classes of functions [IP07] addressed the question for depth-bounded *branching programs*. Many of the above protocols rely on the Conditional Disclosure of Secrets (CDS) methodology [GIKM98]. This is a light-weight alternative to Zero-knowledge proofs, allowing to disclose a secret string s conditioned on some encrypted input c satisfying some condition. The idea is that instead of making the client prove that its queries are well formed, it suffices for the server to disclose its answer c' to the receiver and only under the condition that the queries are well formed. Using the homomorphic property of the encryptions, the latter conditional disclosure can be done without the server even knowing whether the condition is satisfied. The original CDS solutions from [AIR01] relies on additively homomorphic encryption over groups of a prime order. An efficient extension to groups of a composite order was suggested in [Lip05], assuming that the order of the group is sufficiently "rough". This technique turned out to generalize to a certain extent to situations where the key and ciphertexts may not be well-formed. Roughly, for malformed the secret luckily remained hidden even if the CDS was obliviously performed on the (possibly malformed) encryptions and pk as if they were proper.

1.2 Our result

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 $\mathcal{M}$, and weak (evaluate formulas) non-compact maliciously circuit-private HE $\mathcal{A}$ (here, by "compact" we refer to the strong requirement that encrypted outputs have size $poly(\lambda)$, where λ is the security parameter). Our construction proceeds in three steps:

Lemma 1 (folklore)**.** *A (compact) FHE without privacy can be upgraded to (compact) semi-honestly circuit private by decrypting it under a (possibly weaker and non-compact) semi-honestly private HE.*

Lemma 2 (this paper)**.** *A semi-honestly private FHE can be upgraded to maliciously circuit-private by homomorphically validating its keys and inputs under a (possibly weaker and non-compact) maliciously circuit-private FHE.*

Unfortunately, for known instantiations the scheme resulting from Lemma 1 doesn't satisfy the requirements of lemma 2 (validation becomes hard) but we show that combining the two steps in a simple way does work. The resulting output is not compact but fortunately:

Lemma 3 (this paper)**.** *A non-compact circuit-private FHE can be upgraded to compact by homomorphically decrypting it under a compact (F)HE.*

Let us now elaborate on each of the steps.

Step 1. The first step transforms a (compact) FHE scheme $\mathcal{M}$ into a semi-honestly circuit-private scheme. For that purpose, it uses an auxiliary circuit-private scheme $\mathcal{A}$ to re-randomize $\mathcal{M}$'s output, which may in principle contain structural information about C, beyond $C(x)$ on some input vector x. Eval produces an encrypted output via Eval_M - denoted by out_M. It then homomorphically evaluates the decryption circuit of $\mathcal{M}$ using $\mathcal{A}$ with out_M hardwired, and an encryption of sk_M, a_{sk_M} used as input, and outputs the result out_A. The encryption a_{sk_M} and the public key for $\mathcal{A}$ are provided as part of the public key pk. This process in fact re-encrypts the output under sk_A, so The decryption (of the result) is done via sk_A (sk_M is not needed).

Indeed, in the above solution, since decryption outputs exactly the information (a single bit - $C(x)$) we are supposed to learn, and since $\mathcal{A}$ is circuit-private, it is all the information we learn. It can also be viewed as applying a re-encryption via $\mathcal{A}$ gadget to Eval_M's output.

Note that since $\mathcal{M}$ is compact $|out_A| = \mathsf{poly}(\lambda)$ even if $\mathcal{A}$ is not compact. In particular, [vDGHV09]'s solution of using Yao for evaluating the decryption circuit can be viewed as a special case of that approach. This is a general form of "re-randomization", independent of the structure of the ciphertext domain.

Step 2. However, this approach generally fails in the malicious setting, even if $\mathcal{A}$ is maliciously circuit-private. One problem is that the input to Eval_M may be arbitrarily malformed, and thus out_M is not guaranteed to be a valid encryption of (the right, or even) a bit, and applying decryption to it might reveal extra information about the circuit. Moreover, even if the input vector encryption c_M and public key pk_M are well formed, sk_M "reported" via a_{sk_M} may be mal-formed in a way that the decryption circuit outputs some invalid structural information about C, rather than a decryption of the input ciphertext.

Therefor, we devise a second step which transforms a semi-honestly circuit-private scheme $\mathcal{M}$ into a maliciously circuit-private scheme. The idea is to apply (during Eval) a form of CDS that discloses the encrypted output of $\mathcal{M}$ conditioned that its public-key is well-formed, and the encrypted input bits are valid encryptions under that key. The validation circuit also gets a purported additional randomness $r_{M,k}, r_{M,e}$ as used by the key generation and encryption algorithms. This CDS is implemented by homomorphically evaluating the validation circuit (where the purported public key and encryptions are hardwired) under an auxiliary scheme $\mathcal{A}$ which is (maliciously) circuit-private, and supports the evaluation of such a circuit. The inputs of the circuit of the "secret" random strings $r_{M,k}, r_{M,e}$, provided as bit-by-bit encryptions under $\mathcal{A}$ provided as part of the public key (along with pk_A). One way to perform the disclosure it to evaluate an augmented validation circuit (with several output bits) that output the string out_M (known upon circuit construction, and can be hardwired into it) or a string of zeros according to whether the validity condition is satisfied. The main observation showing why this works is that (malicious) circuit privacy of $\mathcal{A}$ ensures that even if pk_A and the encryptions of $r_{M,k}, r_{M,e}$ under $\mathcal{A}$ are malformed, they induce *some* values $\boldsymbol{x}^*_{\boldsymbol{A}}$, such that all the output of the validation circuit reveals is its output corresponding to these values (and the hardwired $pk_M, \boldsymbol{c_M}$). Now, if everything was well formed, the validation circuit outputs out_M. By semi-honest privacy of $\mathcal{M}$, this reveals only $C(x)$, where x is the bit vector encrypted via c. Otherwise, the validation circuit's output on $\boldsymbol{x}^*_{\boldsymbol{A}}$ is $\mathbf{0}$, and thus reveals nothing about out_M. The resulting scheme is compact if $\mathcal{A}$ is, and is "somewhat compact" if $\mathcal{A}$ is not compact, with encrypted outputs of size $poly(\lambda, n)$.

Note that the scheme $\mathcal{A}$ does not need to be the same one in both steps, as long as each scheme supports the corresponding functionality (decryption and validation respectively).

What do we have so far? Running steps 1,2 above on $\mathcal{M}, \mathcal{A}$ results in (possibly non-compact) maliciously circuit-private FHE. Fortunately, this can be addressed by applying Lemma 3 to compact the scheme using (the compact) $\mathcal{M}$ itself.

More problematically, this clean two-step transformation runs into technical difficulties, since it is unclear how to implement the first step in a way that results in a semi-honestly secure $\mathcal{M}'$ which has validation circuits, supported by any maliciously circuit-private scheme $\mathcal{A}$ we know how to implement. Concretely, we do not know of maliciously circuit-private OT with decryption circuits in NC^1. We note that some schemes obtained by direct ad-hoc re-randomization (see e.g. [vDGHV09][Section 6]) do have efficient enough circuits, and can be plugged into the second step of the transformation.

However, it turns out we can combine the decryption and validation circuits into one, which either outputs the decrypted bit if validation passed, and 0 otherwise. We can then evaluate this circuit under a single maliciously private scheme $\mathcal{A}$, obtaining a single, direct transformation from (compact) FHE with no circuit private into a maliciously circuit-private FHE. This is the version we present in the technical part of the paper.

We instantiate $\mathcal{A}$ with a variant of Yao's garbled circuits combined with malicious-receiver OT, obtaining a scheme for evaluating NC^1. Then $\mathcal{M}$ can be instantiated with several FHE schemes from the literature, such as [vDGHV09], or [BV11]. Note that (e.g.) [BV11] has negligible (non-zero) decryption error, when the added noise happens to be too high. As circuit privacy of our construction relies on perfect correctness, this is not acceptable. However, properly truncating the noise in [BV11] yields a (still secure) scheme which is perfectly correct. Thus, after running step 3, we obtain the following theorem.

Theorem 4. *(informal) Assume there exists an OT-homomorphic maliciously circuit-private scheme. Assume further that there exists a compact FHE scheme with decryption and validation circuits in* NC^1*. Then there exists a compact circuit-private FHE scheme* $\mathcal{F}$*.*

Before describing step 3, let us see look more closely at the compactness of the scheme we already achieve.

The Yao-based instantiation of $\mathcal{A}$ described above is not compact. That is, the encrypted output size depends on λ, m, where m is the size of the input to the evaluated circuit C, and $|\mathcal{C}|$ as well. The first observation is that we only evaluate a family of functions in NC^1, so the depth of formulas for these functions are bounded by $O(\log n)$.

Thus, circuit size is at most $2^{O(\log m)} = m^c$. Since $m = poly(\lambda, n)$ for the validation circuit (and $poly(\lambda)$ for the decryption circuit), overall encrypted output size is $poly(\lambda, n)$.

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. More importantly, it is currently unknown how to achieve (compact) FHE without assuming so called (weak) *circular security* for a certain HE scheme, referred as "somewhat homomorphic" (see [Gen09]). This assumption is generally considered somewhat risky and undesirable. A common solution found in the FHE literature to avoid a circular security assumption is using a different key per circuit layer; the price is that the scheme becomes *leveled*. That is, KeyGen takes a bound d on the circuit depth, and the size of public key passed to Eval grows with d. [2] The size of all other outputs of the scheme (pk_{Enc}, sk, encrypted inputs and outputs) remains independent of d.

As to our application of computing on encrypted data, using leveled FHE instead of FHE makes receiver's complexity worse, by making his work depend on d. Although this dependence is sublinear, this is quite undesirable but inherent when using leveled FHE (even without circuit

[2]This also (mildly) constrains circuit privacy — a bound on the circuit size must always be leaked.

privacy). To minimize the ill effect of this dependence, we would like to limit it to the key generation complexity. This way, a single expensive key generation can be followed by multiple cheap evaluation queries, amortizing the dependence on circuit depth.

Going back to our framework, if $\mathcal{M}$ is leveled and $\mathcal{A}$ is not compact, then validation should take pk_{Eval} as input as well, making the input to the validation circuit of size $poly(\lambda, n, d)$. Thus, by similar computation to the above, encrypted output size is now $poly(\lambda, d, n)$ — a dependence which is better to avoid.

Step 3. The idea is to combine circuit-private (non-compact) HE $\mathcal{P}$ with (non-circuit-private) FHE $\mathcal{S}$ "in the other direction". That is, use $\mathcal{S}$ to homomorphically decrypt the encrypted output of the HE scheme to "deflate" it. Intuitively, even though the FHE $\mathcal{S}$ used for decrypting is not circuit-private, the resulting scheme is, because homomorphic decryption merely acts upon a string that we originally sent "in the plain", so there is no need to protect it.

To avoid circular security for the FHE scheme in this construction, we use leveled FHE as well, resulting in leveled circuit-private HE for NC^1. We then plug this compact HE scheme as $\mathcal{A}$ into our main construction. Resolving some technical issue related to bounding the depth of circuits to be evaluated by $\mathcal{S}$ is now leveled, we obtain a compact leveled circuit-private FHE.

2 Preliminaries

Notation We use ***vector*** to denote over D^t, for some $t > 0$, and some finite domain D. For a pair of vectirs v_1, v_2 (v_1, v_2) denotes the vector resuting from concatenating v_1, v_2. For vectors v_1, v_2 over some D_1^t, D_2^t $(v_1; v_2)$ denotes $((v_{1,1}, v_{2,1}), \ldots, (v_{1,t}, v_{2,t}))$. For a function $f(a, b, c, \ldots)$, $\boldsymbol{f}(\boldsymbol{a}, b, \boldsymbol{c}, \ldots)$ is a shorthand for $f(a_1, b, c_1, \ldots), f(a_2, b, c_2, \ldots), \ldots$. When considering function vectors, all inputs which are the same in all executions do not appear in bold (even if they are vectors by themselves). 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 sometines write $out \in A(x)$ as a shorthand for $out \in support(A(x, r))$. By $neg(k)$ we refer to a function that for all polynomials $p(k)$, $neg(k) < p(k)$ for all $k > K$, where K is a constant determined by p.

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

2.1 Representation Models

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 effecient 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).

Two models we will be interested in are boolean circuits and variants thereof such as formulas. More concretely, by circuits we mean boolean circuits over AND (denote $\wedge$), XOR (denote $\oplus$) gates of fan-in 2, and NOT gates (no constants).

We assume wlog. that the circuit graph (a DAG) is connected, in the sense that all input nodes are connected to the output wire; we also assume wlog. that all not gates are applied to input nodes. For completeness, for circuits and other models we let $U(C, x) = 0$ whenever the input (C, x) is syntactically malformed, such as, for isntance, when C is not a valid encoding of a circuit at all, or the underlying DAG is not connected, x contains less variables then expected by C etc.

The *size* of a circuit is the number of wires in the circuit's underlying graph. The *depth* of the circuit is the number of gates on the longest path between an input wire and the output wire of the circuit. As circuits are the strongest model one can hope for in the sense of program compactness (up to polynomial factors), and as HE candidates for general circuits (FHE) are known since the seminal work of [Gen09], this is the model typically considered in modern literature on homomorphic encryption. This is also the default representation model used throughout the paper.

By *formulas* we mean circuits as above for which the underlying graph is a tree. Furthermore, valid formulas have depth at most $c \log \text{size}(C)$, where c is a global constant. c is picked large enough, so that the expressive power of formulas remains the same (that is, it is known every formula can be transformed into this form with polynomial blowup in size).

Complexity Classes. When we consider a family of circuits, this family is (polynomial time) uniform unless stated otherwise. The class poly denotes the set of all (boolean) families of functions with circuits of size bounded by some polynomial $p(n)$. NC^i denotes the class of all (boolean) functions with $c \log \text{size}(C)$-depth bounded formulas C for some constant c, and size $poly(n)$.

2.2 Homomorphic encryption

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. Throughtout the paper λ denotes the security parameter taken by a scheme.

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

$\mathsf{Eval}(pk, C, \boldsymbol{c} = (c_1, \ldots, c_n))$: Takes a public key pk, a bit-by-bit encryption $\boldsymbol{c}$ of a bit vector $\boldsymbol{x} \in \{0,1\}^n$, a function represented by a program C (encoded in a pre-fixed representation model U) and outputs an encryption out of bit $U(C, \boldsymbol{x})$. We assume wlog. that pk includes 1^λ (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.

Definition 3 ($(U,\mathcal{C})$-homomorphic encryption)**.** Let $\mathcal{C} = \bigcup C_\lambda$. We say a scheme $\mathcal{E}$ is $(U,\mathcal{C})$-homomorphic if for every family of programs $C_1(x_1,\ldots,x_{n_1}), C_2, \ldots$, so that $C_i \in \mathcal{C}_\rangle$, and for all λ

$$\mathsf{Dec}(sk, \mathsf{Eval}(1^\lambda, \mathsf{KeyGen}(1^l), C_\lambda, \mathbf{Enc}(pk, \boldsymbol{x}))) = U(C_\lambda, x)$$

for all λ, and all random choices of the algorithms involved. We say the scheme is λ-independent if $\mathcal{C}_\lambda = \mathcal{C}_\lambda$ for all λ. By default our scheme are λ-independently homomorphic (in particular the $\mathcal{C}_\lambda$'s are not explicitely defined).

Definition 4. We say a $(U,\mathcal{C})$-homomorphic scheme $\mathcal{E}$ is *compact* if there exists an output bound $B(\lambda, n, |C|) = \mathsf{poly}(\lambda)$ on all outputs of Eval on 1^λ and programs $C \in \mathcal{C}_\lambda$. We say the scheme is *input-compact* if $B(\cdot) = \mathsf{poly}(\lambda, n)$ (but does not depend on C). If the scheme is compact (input-compact) λ-independetly homomorphic for all circuits, we (and elsewhere in the literature) call it a comact (input-compact) FHE (fully homomorphic scheme).

Definition 5 (Bootstrappable homomorphic encryption)**.** For a $\mathcal{C}$-homomorphic scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$, let $\mathsf{Dec}^{\oplus}_\lambda(c_1, c_2, sk), \mathsf{Dec}^{\wedge}_\lambda(c_1, c_2, sk), \mathsf{Dec}^{\neg}_\lambda(c, sk)$ 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 λ, $\mathsf{Dec}^{\oplus}_\lambda\big|_{c_1,c_2}$, $\mathsf{Dec}^{\wedge}_\lambda\big|_{c_1,c_2}$, $\mathsf{Dec}^{\neg}_\lambda\big|_c$ are in $\mathcal{C}_\lambda$.

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 inludes a fixed part pk_{Enc}, depends only on λ; likewise, sk depends only on λ. 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 algorithms for all d. The notions of compact and input-compact HE are as for non-leveled schemes (the bound $B(\cdot)$ on output size is independent of d). For compact schemes, the algorithm Dec is also independent of d. We say such a leveled scheme is an FHE if the (standard) scheme $\mathcal{E}^{(\mathcal{D})}$ induced by fixing $d = D$ when calling $\mathsf{KeyGen}(1^d, \cdot)$ induces a λ-independently $\mathcal{C}$-homomorphic scheme $\mathcal{E}^{(\mathcal{D})}$, where C is the set of all depth-D circuits. In such FHE, the encrypted outputs' size is still $poly(\lambda)$ (for a global polynomial independent of d).[3]

Standard FHE schemes can be thought of as a special case of leveled FHE schemes where KeyGen simply ignores d. Thus, all schemes $\mathcal{E}^{(\lceil)}$ are the same (standard) FHE scheme. We refer to this special case as *unleveled* FHE.

The reason this relaxation is considered, is that so far, all FHE schemes were obtained using Gentry's bootstrapping theorem, which assumes circular security of an underlying bootstrappable HE scheme. Constructions of bootstrappable HE schemes from (more) standard assumptions (rather than just assuming circular security) are currently unknown. On a high level, leveled FHE schemes allow to circumvent this difficulty by encrypting each key under a different key, according to an acyclic sequence of independently generated keys, and publishing them all as pk. We need the following properties of the bootstrapping theorem.

Theorem 5 ([Gen09])**.** *Let $\mathcal{E} = (\mathsf{KeyGen}_E, \mathsf{Enc}_E, \mathsf{Eval}_E, \mathsf{Dec}_E)$ denote a bootstrappable, $\mathcal{C}$-homomorphic HE scheme. Then there exists a compact leveled FHE scheme $\mathcal{BS}$ with the following properties:*

[3]All but output compactness are not a strict requirement, but it is typically achieved in leveled schemes form the literature. Also, for our application of computing on encrypted data, we want to minimize the dependence of receiver's work on circuit parameters.

- $\mathsf{KeyGen}_{BS}(1^d, 1^\lambda)$ *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* $\mathsf{KeyGen}_E(1^\lambda)$*; 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* $\mathsf{KeyGen}(1^\lambda)$ *(which results in a standard FHE scheme).*

Definition 6. 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 $\mathsf{Sim}(1^\lambda, pk^*, \boldsymbol{c^*}, b)$, deterministic $\mathsf{Ext}(1^\lambda, pk^*, c^*)$, such that for all λ, and all $pk^*, \boldsymbol{c^*} = \boldsymbol{c^*}_1, \ldots, \boldsymbol{c^*}_n$ and all programs $C : \{0,1\}^n \to \{0,1\} \in (U, \mathcal{C})$ in $\mathcal{C}_\lambda$ the following holds:

- $\boldsymbol{x^*} = \mathbf{Ext}(1^\lambda, pk^*, \boldsymbol{c^*})$.

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

In particular, for circuits $C(x_1, \ldots, x_n) \in \mathcal{C}_\lambda$ the output distribution of Eval (including length) depends only on n, λ. 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.

3 Framework

We introduce the following representation model $(U_{SI}, \mathcal{C})$ (from "split-input") we will need. Programs in the model are represented by a pair (C_p, C_s), where C_p is a circuit on some m variables, and $C_s \in \{0,1\}^t$ form some $t \leq m$. A program (C_p, C_s) is intepreted as a function f over $n = m - |C_s|$ variables via $U_{SI}((C_p, \boldsymbol{C_s}), \boldsymbol{x}) = C_p(\boldsymbol{C_s}, \boldsymbol{x})$. 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 satisties Definition 6, with the only modification that Sim receives C_p as an input.

Observe that picking $n = m^{o(1)}$ results in a program of size superpolynomial in n. This setting of parameters will in fact be useful in our constructions (resulting in efficient constructions).

Remark. 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. The scenario there is that a function f known to the adversary is to be homomorphically evaluated on a (partially) secret input z, together with the adversary's input x, so hiding f would be an overkill. More specifically, we will need to devise scheme for formulas with depth bounded by $c \log n$ for a specific constant c. It is possible to implement standard circuit privacy for this family using an information-theoretic version of Yao for universal formulas (note that on input $x \in \{0,1\}^n$, the relevant formulas are of size $poly(n)$ for a prefixed polynomial, and that formula evaluation is indeed in NC^1.) However, using ad-hoc Yao garbled circuits for each formula C_p to evaluate is a simpler approach.

3.1 From compact FHE to circuit-private FHE

In this section we formalize the decryption and validation transformations described in the introduction. In this transformation, we combine a main scheme $\mathcal{M}$ which is a (compact) FHE scheme with no circuit privacy, and combine it with an circuit-private auxiliary sheme $\mathcal{A}$ which is circuit-private. More precisely, we evaluate C via $\mathcal{M}$. Then, $\mathcal{A}$ is used to (homomorphically) decrypt out_M (under $\mathcal{A}$) and obtain a purported encryption out_A. If out_M was generated semi-honestly, this "re-randomzises" out_M to depend only on the encrypted output bit. Then we disclose out_A conditioned on the public-secret key and encryptions (of $\mathcal{M}$) are well-formed. The condition is homomorphically verified under $\mathcal{A}$.

For simplicity of implementation we combine the decryption and validation step into a single circuit, to be homomorphically evaluated by $\mathcal{A}$.

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

1. Let $Validate_{\lambda,d,n}(pk_M, \boldsymbol{c_M}, sk'_M, \boldsymbol{r_M})$, where:
   - (pk_M, sk_M) is a purported public-key private-key pair output by $\mathsf{KeyGen}_M(1^\lambda, 1^d)$. $sk'_M = (sk_M, r_k)$ where r_k is the purported random string used in the generation of (pk_M, sk_M).
   - $\boldsymbol{c_M} = (c_{M,1}, \ldots, c_{M,n})$ is a purported encryption under pk_M of a bit vector and $\boldsymbol{r_M}$ is a purported vector of randomness used when generating the c_i's.

$$Validate_{\lambda,d,n}(\ldots) = \begin{cases} 1 & \text{if } (pk_M, sk_M) \in \mathsf{KeyGen}_M(1^\lambda, 1^d), \text{and} \\ & \forall i\ (c_{M,i} \in \mathsf{Enc}_M(b_i, r_{M,i}) \text{ for some bit } b_i \in \{0,1\}) \\ 0 & \text{otherwise.} \end{cases}$$

2. Let $Dec_\lambda(sk_M, out)$ denote the decryption circuit of $\mathcal{M}$ instantiated with security parameter λ (recall $\mathsf{Dec}, \mathsf{Enc}$ are independent of d).
3. Define $Out_{\lambda,d,n}(pk_M, c_M, sk'_M, r_M, out_M) = Validate_{\lambda,d,n}(pk_M, c_M, sk'_M, r_M) \wedge Dec_\lambda(sk_M, out_M)$.
4. Let $\mathcal{C}_{\mathcal{M}}$ include all pairs of the form $C = (Out_{\lambda,d,n}(\ldots), (pk_M, c_M, out_M))$.

Theorem 6 (Main). *Let $\mathcal{M} = (\mathsf{KeyGen}_M, \mathsf{Enc}_M, \mathsf{Eval}_M, \mathsf{Dec}_M)$ denote a compact leveled FHE scheme, and $\mathcal{A}$ denote a $(U_{SI}, \mathcal{C}_{\mathcal{M}})$-homomorphic, input-private scheme. Consider the resulting scheme $\mathcal{P}$ as specified in Construction 1 when instantiating with $\mathcal{M}, \mathcal{A}$. $\mathcal{P}$ is a circuit-private FHE. It is unleveled iff $\mathcal{M}$ is unleveled, and is compact iff $\mathcal{A}$ is compact. If $\mathcal{A}$ is not compact, $\mathcal{P}$'s output complexity is $poly(\lambda, d, n)$ ($poly(\lambda, n)$ if $\mathcal{M}$ is unleveled).*

Construction 1. Let $\mathcal{M}$,$\mathcal{A}$ be schemes as above. We construct the following scheme $\mathcal{P}$.

$\mathsf{KeyGen}_P(1^\lambda)$**:** let $(pk_A, sk_A) \xleftarrow{\$} \mathsf{KeyGen}_A(1^\lambda)$, $(pk_M, sk_M) \xleftarrow{\$} \mathsf{KeyGen}_M(1^\lambda, 1^d)$, and let $sk'_M = (sk_M, r_k)$, where r_k is the randomness used by KeyGen_M; $\boldsymbol{a_{sk'_M}} = \boldsymbol{\mathsf{Enc}_A}(pk_A, \boldsymbol{sk'_M})$. Return $(pk_P, sk_P) = ((pk_A, pk_M, \boldsymbol{a_{sk'_M}}), (sk_A, sk_M))$. Here $pk_{P,\mathsf{Enc}} = (pk_A, pk_{M,\mathsf{Enc}}, a_{sk_M})$ (we naturally denote $a_{sk'_M} = (a_{sk_M}, a_{r_k})$).

$\mathsf{Enc}_P(pk_P = (pk_A, pk_{M,\mathsf{Enc}}, a_{sk_M}), b \in \{0,1\})$**:** Return $(c, \boldsymbol{a_{r_M}}) = (\mathsf{Enc}_M(pk_{M,\mathsf{Enc}}, b), \boldsymbol{\mathsf{Enc}_A}(pk_A, \boldsymbol{r_M}))$, where r_M is the randomness used by Enc_M.

$\mathsf{Eval}_P(1^\lambda, pk_P = (pk_A, pk_M, a_{sk'_M}), C, \boldsymbol{c} = (\boldsymbol{c_M}; \boldsymbol{a_{r_M}}))$**:**

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 $out_M = \mathsf{Eval}_M(pk_M, C, c))$.
3. Let $(C_p, C_s) = (Out_{\lambda,d,n}, (out_M, pk_M, \boldsymbol{c_M}))$
4. Compute and output $out_A = \mathsf{Eval}_A(pk_A, (C_p, C_s), \boldsymbol{a_{sk'_M}}, \boldsymbol{a_{r_M}})$.

$\mathsf{Dec}_P(sk_P = sk_A, out_A)$**:** Output $out = \mathsf{Dec}_A(sk_A, out_A)$.

Proof of Theorem 6. It is easy to see that the proposed scheme remains correct if all keys and ciphertexts are well-formed. Then out_A encrypts $C(x) \wedge Validate_{\lambda,d,n}(\ldots)$, where the latter equals 1, and thus equals $C(x)$.

As to efficiency: We run Eval_A on some $Out_{\lambda,d,n}$ on input $(pk, sk', \boldsymbol{c_M}, \boldsymbol{r_M}, out_M)$. By definition of leveled schemes, this input to $Out_{\lambda,d,n}$ is of size $m = poly(\lambda, d, n)$. The dependence on d is only because $|pk|$ may depend on d in leveled schemes. Thus, if $\mathcal{A}$ is compact, the encrypted output size $|out_A|$ is $poly(\lambda)$ (a compact shceme). Otherwise, if $\mathcal{M}$ is standard (not leveled), $m = poly(\lambda, n)$, thus encrypted output size is also $poly(\lambda, n)$ (input-compact).

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 $\mathsf{Ext}_P, \mathsf{Sim}_P$, as required.

$\mathsf{Ext}_P(1^\lambda, 1^d, pk^* = (pk^*_A, pk^*_M, a^*_{sk'_M}), c^* = (c^*_M, a^*_{r_M}))$**:** 1. Let $\mathsf{Sim}_A, \mathsf{Ext}_A$ as guaranteed by Definition 6, and let $\boldsymbol{x^*_A} = (r^*_M, sk'^*{}_M) = \boldsymbol{\mathsf{Ext}_A}(1^\lambda, pk^*_A, \boldsymbol{a^*_{r_M}}, \boldsymbol{a^*_{sk'_M}})$.

2. If $(pk^*_M, sk^*_M) \neq (\mathsf{KeyGen}_M(1^\lambda, 1^d, r^*_k))$, return 0. Here, the "secret" parts r^*_k, r^*_M, sk^*_M are taken from Ext_A's output, and the rest are taken from the input.
3. Otherwise, If $c^*_M = \mathsf{Enc}_M(pk^*_M, r^*_M, b)$ for some bit b, return b.
4. Otherwise, return 0.

$\mathsf{Sim}_P(1^\lambda, pk^* = (pk^*_A, pk^*_M, a^*_{sk'_M}), \boldsymbol{c^*} = ((c^*_{M,1}, a^*_{r_{M,1}}), \ldots, (c^*_{M,n}, a^*_{r_{M,n}})), b)$**:** 1. Let $\mathsf{Sim}_A, \mathsf{Ext}_A$ as guaranteed by Definition 6, and let $\boldsymbol{x^*_A} = \boldsymbol{\mathsf{Ext}_A}(1^\lambda, pk^*_A, (\boldsymbol{a^*_{r_M}}, \boldsymbol{a^*_{sk'_M}})))$.

2. If the check in $\mathsf{Ext}(1^\lambda, pk^*, c^*_i)$ fails for (pk^*_M, sk'^*_M) or for some $i \in [n]$:
   Output $\mathsf{Sim}_A(1^\lambda, pk^*_A, Out_{\lambda,d,n}, \boldsymbol{a^*_{sk'_M}}, \boldsymbol{a^*_{r_M}}, 0)$.
3. Otherwise: output $\mathsf{Sim}_A(1^\lambda, pk^*_A, Out_{\lambda,d,n}, \boldsymbol{a^*_{sk'_M}}, \boldsymbol{a^*_{r_M}}, b)$.

We have specified Ext_P of the proper form, so it remains to analyze Sim_P. Let $x^*_A = (sk^*_M, r^*_M, r^*_k) = \boldsymbol{\mathsf{Ext}_A}(1^\lambda, pk^*_A, \mathsf{Dec}_{M,\lambda}, (\boldsymbol{a^*_{sk_M}}, \boldsymbol{a^*_{r_M}})))$. There are several cases.

- Assume $Validate_{\lambda,d,n}|_{c=c^*_M, pk=pk^*_M}(sk'^*_M, r^*_M) = 0$. Then by definition $Out_{\lambda,d,n|c^*_M, pk^*_M, out_M}(sk'^*_M, r^*_M) = 0$ *for all* out_M. By construction of $\mathsf{Sim}_P, \mathsf{Ext}_P$, this condition is always correctly deteced (line 2). By circuit privacy of $\mathcal{A}$, and as the bit passed to Sim_A is $Out_{\lambda,d,n}(C_s, sk'^*_M, r^*_M)$ for

$C_s = (c^*_M, pk^*_M, out_M)$, which exactly equals $Out_{\lambda,d,n|c^*_M,pk^*_M,out_M}(sk'^*_M, r^*_M) = 0$ (as computed in Eval_P). Thus, Sim_A (in line 2) receives $b = Out_{\lambda,d,n|c^*_M,pk^*_M,out_M}(sk'^*_M, r^*_M) = 0$ and statistically simulates Eval_P's output in this case.

- Otherwise, $(pk^*_M, sk^*_M) = \mathsf{KeyGen}(1^\lambda, 1^d, r^*_k)$, and $\boldsymbol{c^*_M} = \mathbf{Enc}_{\boldsymbol{M}}(pk^*_M, \boldsymbol{x_M}, \boldsymbol{r^*_M})$ for some $\boldsymbol{x_M} \in \{0,1\}^n$. T In other words, pk^*_M, c^*_M are a valid public-key, encryption-vector pair. Thus the call to $Out_{\lambda,d,n}$ in Eval_P always returns $C(\boldsymbol{C_M})$ by (perfect) correctness of $\mathcal{M}$.

  If we show that $\boldsymbol{x_M}$ is exactly the vector returned by Ext_P, then by $Out_{\lambda,d,n|c^*_M,pk^*_M,out_M}(sk'^*_M, r^*_M) = Out_{\lambda,d,n}(C_s, sk'^*_M, r^*_M)$ for $C_s = (c^*_M, pk^*_M, out_M)$ and correctness of Sim_A, validity of Sim_P follows. So, let $\boldsymbol{x^*_M}$ denote a vector as extracted in the series of executions of Ext_P on $\boldsymbol{c^*}$. This is so, since Ext_P exits in line 3 on all c^*_i, and it can not return a different bit $x^*_i \neq x_{M,i}$, or we obtain two valid decryptions of some encryption c^*_i, contradicting the correctness $\mathcal{M}$ (Plug the identity function $f(x_1) = x_1$ into Definiton 3). Thus, Sim_P receives $b = C(b_M)$ as its last output.

3.2 Compactization of circuit-private FHE

Recall that for the application of computing on encrypted data, it is ideal that the underlying FHE has encrypted inputs of size $poly(\lambda, n)$ (independent of $|C|$), or better still, just "constant size" $poly(\lambda)$. If $\mathcal{A}$ in Theorem 6 is compact, then we indeed get encrypted inputs of size $poly(\lambda)$. There exist instantiations of $\mathcal{M}$ from the literature with validation and decryption procedures in NC^1. Since (e.g) validation typically needs to read all the input bits, this is the best one can hope for. Unfortunately, we do not know of compact circuit-private circuit-private HE for this class, but do know of non-compact ones. Thus, a plausible solution is obtained for unleveled compact unleveled FHE as $\mathcal{M}$ and a suitable (non-compact) $\mathcal{A}$.

In this section, we devise a simple transformation (corresponding to Lemma 3 in the introduction) for making a (possibly leveled) scheme's output compact (only $poly(\lambda)$), while preserving circuit privacy.

The idea is to use bootstrapping similar to the one described in Lemma 2 the introduction for transforming FHE into FHE with semi-honest circuit privacy. That construction applies a gadget that homomorphically evaluates a circuit using a (compact) FHE $\mathcal{M}$, and then (homomorphically) decrypts the result using a (not necessarily compact) maliciously circuit-private HE $\mathcal{A}$ (which is only input-compact, homomorphic for a weaker class of functions) resulting in a (compact) FHE which is *semi-honsetly* circuit-private. It turns out that reversing the roles of $\mathcal{M}$ and $\mathcal{A}$ in the gadget, results in a scheme which is both compact and preserves malicious circuit privacy (for the class $\mathcal{A}$ it is homomorphic for).

Theorem 7 (Compaction theorem)**.** *Let $\mathcal{P}$ be a leveled $\mathcal{C}$-homomorphic circuit-private scheme. Let $\mathcal{S}$ denote a compact FHE scheme.*[4] *Then the scheme $\mathcal{PS}$ in the following construction is a compact $\mathcal{C}$-homomorphic circuit-private scheme.*

Construction 2. Let $\mathcal{P}, \mathcal{S}$ be HE schemes as in Theorem 7.

[4]In fact, $\mathcal{S}$ should only be compact and homomorphic for the circuit family it is used for. It doesn't need to be an FHE.

$\mathsf{KeyGen}_{PS}(1^\lambda, 1^d)$**:** Sample $(pk_P, sk_P) \xleftarrow{\$} \mathsf{KeyGen}_P(1^\lambda, 1^d, r_k)$; let $sk'_p = (sk_P, r_k)$; $(pk_S, sk_S) \xleftarrow{\$} \mathsf{KeyGen}_S(1^\lambda)$; $\boldsymbol{s_{sk'_P}} \xleftarrow{\$} \boldsymbol{\mathsf{Enc}_S}(pk_S, \boldsymbol{(sk_P, r_k)})$. Output $(pk, sk) = ((pk_P, pk_S, s_{sk_P}), sk_S)$ Here $pk_{\mathsf{Enc}} = (pk_{P,\mathsf{Enc}}, pk_S, s_{sk_P})$.

$\mathsf{Enc}_{PS}(pk, b)$**:** Output $\mathsf{Enc}_P(pk_{P,\mathsf{Enc}}, b)$. [5]

$\mathsf{Eval}_{PS}(1^\lambda, pk, C, c)$**:**

- $out_P \xleftarrow{\$} \mathsf{Eval}_P(1^\lambda, pk_P, C, c)$.
- Let $Dec_{P,\lambda,d}$ denote the decryption circuit of $\mathcal{P}$ with parameter λ.
  Then $Dec_{P,\lambda,d}|_{out=out_P}(sk_P)$ is a circuit for decrypting (hard-wired) out_P under secret keys generated by KeyGen_P.
- Compute and output $out_S = \mathsf{Eval}_S(1^\lambda, pk_S, Dec, s_{sk_P})$.

$\mathsf{Dec}_{PS}(sk = sk_S, out)$**:** Output $\mathsf{Dec}_S(sk_S, out)$.

Proof sketch. The main observation is that Eval computes the output of Eval_P on the input $pk^*, \boldsymbol{c}^*$, which reveals no "redundent" information, and perform some randomized computation on it (homomorphically decrypt via $\mathcal{S}$), using randomness which is independent of out_P. More precisely, we can define:

$\mathsf{Ext}_{PS}(1^\lambda, 1^d, pk^*, c^*)$**:** return $x^* = \mathsf{Ext}_P(1^\lambda, pk^*, c^*)$.

$\mathsf{Sim}_{PS}(1^\lambda, 1^d, pk^* = (pk^*_P, pk^*_S, s^*_{sk_P}), c^*, b = C(\mathsf{Ext}_P(1^\lambda, pk^*, c^*)))$**:**

- Let $out'_P = \mathsf{Sim}_P(1^\lambda, pk^*, b)$.
- Output $out'_S = \mathsf{Eval}_S(1^\lambda, pk^*_A, Dec_{S,\lambda|out=out_P}, \boldsymbol{s_{sk_P}})$.

Let $C \in \mathcal{C}$, and set some pk^*, c^*. Conditioned on $out'_P = out_P$, out'_S is distributed *exactly* like out_S, as both Sim_{PS} and Eval_{PS} run the same process on the same input distribution. Now, out'_P is statistically close to out_P by circuit-privacy of $\mathcal{P}$, so out_S, out'_S are also statisically close, and validity of Sim_{PS} follows.

Putting things together. Let us set $\mathcal{P}$ to be a scheme obtained by instantiating Theorem 6 via an unleveled $\mathcal{M}$ and a suitable, non-compact $\mathcal{A}$. Then, letting $\mathcal{S} = \mathcal{M}$ in Theorem 7, we get a compact scheme "for free".[6]

Theorem 8. *Let* $\mathcal{M} = (\mathsf{KeyGen}_M, \mathsf{Enc}_M, \mathsf{Eval}_M, \mathsf{Dec}_M)$ *denote an unleveled FHE scheme, and* $\mathcal{A}$ *denote a* $(U_{SI}, \mathcal{C}_{\mathcal{M}})$*-homomorphic input-private scheme. Then there exists a circuit-private compact unleveled scheme* $\mathcal{PS}$*.*

[5]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.

[6]The construction would have worked for leveled $\mathcal{M}$ as well, but since we need a compact FHE in Construction 2, it does not buy us relaxated assumptions.

3.2.1 Getting rid of circular security

In the previous section we showed how to transform (unleveled) circuit-private FHE into compact (unleveled) FHE while preserving circuit privacy (Theorem 8). Fortunately, we are able to instantiate Theorem 8 with known constructions from the literature. Unfortunately, we need to assume compact FHE, which we currently do not know how to implement without assuming circular security of a related bootstrappable somewhat homomorphic scheme, which is considered a somewhat risky assumption. As explained before, a common solution to this issue in the FHE literature is constructing compact leveled FHE. We would like to follow this direction for constructing circuit-private compact FHE, relaxing circuit privacy to reveal d (if such a relaxation is not desired, one could settle for assuming circular security and using Theorem 8).

We follow a similar path of combining Theorem 6 with Theorem 7. Naturally, we pick a leveled scheme $\mathcal{M}$ in Theorem 6 (and a suitable non-compact $\mathcal{A}$), resulting in a circuit-private scheme $\mathcal{P}$. To eliminate the need of assuming circular security, we modify Theorem 7 to use a leveled compact $\mathcal{S}$ 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 $m = \mathsf{poly}(\lambda, d, n)$, where d, n is the bound on circuits C the scheme ($\mathcal{PS}$) is to evaluate, and n is the number of variables of the concrete circuit to evaluate. The depth of the decryption circuit $\mathcal{S}$ needs to evaluate is then generaly 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 leaved as with circuits of size (and possibly depth) $poly(m) = poly(\lambda, 2^d)$. Thus, KeyGen_S may run in exponential time in λ, d, n, and thus explode, making KeyGen_{PS} inefficient as well. Assume $\mathcal{P}$ had decryption circuits $\mathsf{Dec}_{\lambda,d}$ of depth $\mathsf{poly}(logm, d, \lambda)$. Then m would be polynomial in λ, d, solving both the potential inefficiency problem, and the need to know n. Luckily, such schemes $\mathcal{P}$ can be obtained, namely, there exist such schemes with decryption circuits of depth $O(\log m, poly(\lambda))$.

A concrete implementation. With a particular instantiation in mind, we impose some additional efficiency requirements on both $\mathcal{A}, \mathcal{M}$ in Theorem 6 allowing to use a combination of a leveled $\mathcal{M}$ and non-compact $\mathcal{A}$ resulting in compact leveled $\mathcal{P}$.

Theorem 9. *Assume there exists a compact leveled FHE $\mathcal{M}$ and $\mathcal{A}$ which is $(U_{SI}, C_{\mathcal{M}})$-homomorphic input-private. Additionally, assume $\mathsf{Dec}_{A,\lambda}(out, sk)$ has encrypted output complexity $poly(\text{depth}(C_p), \lambda)$, where C_p is the circuit used to generate out in Eval_A, and $\{Out_{\lambda,d,n}\}$ induced by $\mathcal{M}$ is in NC^1.*[7] *Then there exists a compact leveled circuit-private scheme $\mathcal{PS}$.*

Proof Sketch. We create $\mathcal{P}$ by using the output of Theorem 6 on $\mathcal{M}, \mathcal{A}$ as $\mathcal{P}$ in Theorem 7, augmented to accept a (compact) leveled $\mathcal{S}$ (and $\mathcal{M}$ as $\mathcal{S}$ as well). As explained before, the main technical difficulty arises in KeyGen_{PS}, calculating d' passed to KeyGen_S at KeyGen by some $poly(\lambda, d)$. By construction, the input to $Out_{\lambda,d,n}$ is of size $m = poly(\lambda, n, d)$ (for some global polynomial independent of d). As $\{Out_{\lambda,d,n}\}$ is in NC^1, the depth of $Out_{\lambda,d,n}$ is $c \log m$ for some (global) constant c. By assumption on $\mathcal{A}$, the size of its encrypted output is $poly(c \log m, \lambda) \leq poly(d, \lambda)$, using $n \leq 2^d$.

[7] We could do with weaker efficiency for the sake of estimating d', but as the only suitable implementations of $\mathcal{A}$ we know of support only formulas, we do not state the more realxed version.

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

4 Instantiations of the framework

We devise instantiations of schemes $\mathcal{M}, \mathcal{A}$ as required in Theorem 9. As these requirements are strictly stronger then the requirements in Theorem 8, they immediately yield an instantiation of Theorem 8 as well.

Lemma 10. *Consider the compact leveled FHE $\mathcal{M}$ obtained by applying the bootstrapping theorem 5 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 $\{Out_{\lambda,d,n}\}$ induced by $\mathcal{M}$ is in NC^1.*

The proof of the above lemma is quite straightforward, see next section for details.

Lemma 11. *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, GM84, Kal05]. Then there exists a circuit-private $(U_{SI}, \mathcal{C})$-homomorphic scheme $\mathcal{A}$ where $\mathcal{C}$ consists of all formulas (recall that we consider formulas as valid if they are sufficiently "balanced", satisfying $\mathrm{depth}(C) \leq D \log \mathrm{size}(C)$ for a global constant D). Furthermore, $\mathcal{A}$ has decryption circuits $\mathsf{Dec}_{A,\lambda}(sk, out)$ of depth $\mathrm{depth}(C_p)poly(\lambda)$, where C_p is the circuit used to generate out (for some C_s).*

The lemma above can be obtained by combining an information theoretic variant of Yao's garbled circuits [IK02] with maliciously circuit-private OT-homomorphic schemes.

We obtain the following instantiation of Theorem 9.

Corollary 12.

Assume the existence of circuit-private schemes which are homomorphic for (bit) OT. Assume further that the sparse subset sum and the approximate GCD assumptions hold (implying the somewhat homomorphic bootstrappable scheme described in [vDGHV09] is secure).Then there exists a leveled compact circuit-private FHE $\mathcal{PS}$. Additionally, if the bootstrappable scheme from [vDGHV09] is weakly circular secure, then $\mathcal{PS}$ is also unleveled.

Proof Sketch. The only non-trivial point to observe here is that Eval_{PS} is efficient. $\mathcal{A}$ is $\mathcal{C}$-homomorphic for $\mathcal{C}$ consisting of all (balanced) formulas. Since $Out_{\lambda,d,n} \in \mathsf{NC}^1$, it has formulas of size polynomial in their input, and and are balanced (c is as required by our representation of formulas). As the input to $Out_{\lambda,d,n}$ is in turn polynomial in Eval_{PS}'s input (in fact, it depends on $pk, \boldsymbol{c}$), the observation follows.

In fact, most instantiations of FHE nowadays have decryption and validation circuits in NC^1. For instance, LWE-based constructions, such as [BV11] have validation (and 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 $\mathcal{M}$.[8] Furthermore, alternative approaches to Gentry's bootstrapping theorem have been recently suggested, so one does not need to go through the sparse subset sum assumption to bootstrap (although weak circular security is still required). Thus, the following generic theorem is arguably the most useful instantiation of our result.

Theorem 13. *Assume leveled compact FHE with decryption and validation circuits ($\mathcal{C}_{\mathcal{M}}$) in NC^1 exists. Assume further that there exist (bit) OT-homomorphic circuit-private HE. Then there exists a leveled circuit-private compact FHE.*

4.1 Proof of Lemma 10.

Notation 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 3.

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

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 $\boldsymbol{s} \in \{0,1\}^\Theta$ 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 $\boldsymbol{w} = (u_1/2^\kappa, \ldots, u_\Theta/2^\kappa)$, where the w_i's are computed with precision κ bits.
3. Output $(pk, sk) = ((\boldsymbol{v}, \boldsymbol{w}), (p, \boldsymbol{s}))$.

$\mathsf{Enc}_{DGHV}(pk, b)$**:** $r \xleftarrow{\$} \mathbb{Z} \cap (-2^{\rho'}, 2^\rho)$; $s \xleftarrow{\$} \{0,1\}^\tau$; 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 $c = (c', \boldsymbol{z})$.

$\mathsf{Eval}_{DGHV}(pk, C, (c_1, \ldots, c_n))$**:** 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 $\boldsymbol{z}$ from $out', \boldsymbol{w}$ as in Enc, and output $out = (out', \boldsymbol{z})$.

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

[8] In fact [BV11] and some other schemes from the literature offer only statistical correctness, while we need perfect correctness for our transformations to work. In BV11, for instance, this issue can be easily addressed by properly trancating the noise used in Enc. As the resulting distribution is statistically close to the original one, the truncation does not affect the scheme's security.

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

- $\rho = \omega(\log \lambda)$; $\rho' = \rho + \omega(\log \lambda)$.
- $\eta \geq \rho \cdot \Theta(\lambda \log^2 \lambda)$.
- $\gamma = \omega(\eta^2 \log \lambda)$.
- $\tau = \gamma + \omega(\lambda)$.
- $\kappa = \gamma\eta/\rho', \theta = \lambda, \Theta = \omega(\kappa \log \lambda)$: 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 5 to the above scheme. We prove that the relevant $\{Validate_{\lambda,d,n}(pk_M, \boldsymbol{c_M}, sk'_M, \boldsymbol{r_M})\}_{\lambda,d,n}$, and $\{\mathsf{Dec}_\lambda(out, sk_M)\}_\lambda$ function families are in NC^1.

Construction validity proof.

- Let us first understand the complexity of relevant operations in $DGHV$ (Construction 3). Given a purported key pair $(pk = (\boldsymbol{v}, \boldsymbol{w}), sk = (p, s))$, 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 $(out, \boldsymbol{w})$, given randomness $\boldsymbol{w}, r$ verifying $out = \sum_i v_i w_i + 2r + b$, and that $\boldsymbol{z} = out' \cdot \boldsymbol{u}$ 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 $\boldsymbol{r_k}$ (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 $\mathsf{\mathbf{Enc}}(pk_i, \boldsymbol{sk_{i-1}})$. 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 $poly(\lambda) \cdot (d + n)$ 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 $O(\lambda + \log n + \log d)$ which is $O(\log \cdot)$ depth in its input size.

4.2 Proof of Lemma 11

Let $\mathcal{OT} = (\mathsf{KeyGen}_{OT}, \mathsf{Enc}_{OT}, \mathsf{Eval}_{OT}, Dec_{OT})$ 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, Kal05, IP07]. Fix such a scheme.

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

Theorem 14 (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 constact D). For each $C(x_1,\ldots,x_n) \in \mathcal{C}$, there exists a randomized, efficiently computable, functionality* $p_C(\boldsymbol{x},\boldsymbol{r}) : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^t$ *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 $\boldsymbol{x}$, $\mathsf{Sim}_C(C(x))$ 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 circuirt 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 varialbe 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 14).

Construction 4.

$\mathsf{KeyGen}_P(1^\lambda)$**:** Let $(pk_{OT}, sk_{OT}) \xleftarrow{\$} \mathsf{KeyGen}_{OT}(1^\lambda)$.

$\mathsf{Enc}_P(pk,b)$**:** output $\mathsf{Enc}_{OT}(pk_{OT},b)$.

$\mathsf{Eval}_P(pk,(C_p(C_s,x),\boldsymbol{C_s}),\boldsymbol{c})$**:** Denote $\ell = |C_s| + |x|$.

Let $p_{C_p}(x,r) = p_1,\ldots,p_{\ell+1}$ be as in Theorem 14.

- 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 $\boldsymbol{out_i} = \mathsf{Eval}_{\boldsymbol{OT}}(pk_{OT}, (\boldsymbol{v_0}; \boldsymbol{v_1}), c_i)$.
- Let $v_r = p_{\ell+1}(r)$.
- Output $(C_p, \boldsymbol{out}, v_r))$.

$\mathsf{Dec}_P(sk_{OT}, (C_p, \boldsymbol{out}, v_r))$**:** 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 $\mathsf{Dec}_{C_p,\lambda}(\cdot)$ be a circuit that recovers the output bit from $p_{C_p}(x, r)$. Output $\mathsf{Dec}_{C_p,\lambda}(k_1, k_2, \ldots, k_\ell, v_r)$.

Theorem 15. *The scheme in Construction 4 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 $poly(\lambda, \log(\text{size}(C)))$.*

As to decryption circuit complexity of the functions Dec is indeed bounded by $poly(\lambda, \log \text{size}(C))$, 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 $|C|$.[9]

Circuit privacy of Construction 4. The extractor $\mathsf{Ext}_P(1^\lambda, pk^*, \boldsymbol{c}^*)$ outputs $\boldsymbol{x} = \mathbf{Ext}_{\boldsymbol{OT}}(1^\lambda, \boldsymbol{c}^*)$. A simulator $\mathsf{Sim}_P(1^\lambda, C_p(z, x), pk^*, \boldsymbol{c}, out = C_p(C_s, x^*))$, where x^* is the output of $\mathsf{Ext}(1^\lambda, pk^*, \boldsymbol{c}^*)$ 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 $out_i = \mathbf{Sim}_{\boldsymbol{OT}}(1^\lambda, \boldsymbol{p_i}, c^*{}_i))$.
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 $i \leq |z|$ correspnds 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):

[9]That is, the decryption algorithm computes the decryption circuit that extracts the encrypted output bit somewhat inderectly. It doesn't 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.

- Let $x^* = \mathsf{Ext}_{\boldsymbol{OT}}(pk^*_{OT}, \boldsymbol{c^*})$.
- Sample $p_{C_p}(\boldsymbol{C_s}, \boldsymbol{x}, r)$ to obtain $p_1, \dots, p_\ell, p_{\ell+1}$.
- The output of Eval is statistically close to

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

By the privacy property in Theorem 14, 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).

5 Open questions

- What other properties of FHE schemes can be combined in a "getting the best of both worlds" manner by using similar bootstrapping gadgets?
- Extend to settings where the input is distributed among several parties (encrypters and a "key generator"). Provide definitions and explore what happens for various corruption thresholds. In particular, maybe it's possible for the adversary to not even learn n. Even in the current setting, can we make the distribution of encrypted outputs independent of the purported encrypted inputs vector (or even their number n)? This property seems to be useful when using the scheme as part of larger scheme, as for instance "strong OT" used in [IP07] to construct circuit-prive HE for branching programs.

References

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

[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.

[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`.

[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.

[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.

[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.

[Kal05] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 78–95. Springer, 2005.

[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.

[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.