Maliciously Circuit-Private FHE from Information-Theoretic Principles

Nico Döttling
Helmholtz Center for Information Security (CISPA), Saarbrücken, Germany

Jesko Dujmovic
Helmholtz Center for Information Security (CISPA), Saarbrücken, Germany
Universität des Saarlandes, Saarbrücken, Germany

Abstract

Fully homomorphic encryption (FHE) allows arbitrary computations on encrypted data. The standard security requirement, IND-CPA security, ensures that the encrypted data remain private. However, it does not guarantee privacy for the computation performed on the encrypted data. Statistical circuit privacy offers a strong privacy guarantee for the computation process, namely that a homomorphically evaluated ciphertext does not leak any information on how the result of the computation was obtained. Malicious statistical circuit privacy requires this to hold even for maliciously generated keys and ciphertexts. Ostrovsky, Paskin and Paskin (CRYPTO 2014) constructed an FHE scheme achieving malicious statistical circuit privacy.

Their construction, however, makes non-black-box use of a specific underlying FHE scheme, resulting in a circuit-private scheme with inherently high overhead.

This work presents a conceptually different construction of maliciously circuit-private FHE from simple information-theoretical principles. Furthermore, our construction only makes black-box use of the underlying FHE scheme, opening the possibility of achieving practically efficient schemes. Finally, in contrast to the OPP scheme in our scheme, pre- and post-homomorphic ciphertexts are syntactically the same, enabling new applications in multi-hop settings.

2012 ACM Subject Classification Security and privacy → Public key encryption; Security and privacy → Information-theoretic techniques

Keywords and phrases Fully Homomorphic Encryption, FHE, Homomorphic Encryption, Oblivious Transfer, Malicious Statistical Circuit Privacy, Multi-Hop, Information Theory, Cryptography

Digital Object Identifier 10.4230/LIPIcs.ITC.2022.4

Related Version *Full Version*: `https://eprint.iacr.org/2022/495`

Funding *Nico Döttling*: This work is partially funded by the Helmholtz Association within the project "Trustworthy Federated Data Analytic" (TFDA) (funding number ZT-I-OO1 4).
Jesko Dujmovic: This work is partially funded by the Helmholtz Association within the project "Trustworthy Federated Data Analytics" (TFDA) (funding number ZT-I-OO1 4).

1 Introduction

Fully Homomorphic Encryption

Fully homomorphic encryption (FHE) [18] has caused a paradigm shift in achieving round and communication efficient secure computation. FHE allows an untrusted server to publicly evaluate any function over encrypted data without the help of a secret key. FHE has become a tremendous success story in the last ten years, with constructions from increasingly weaker assumptions and achieving better efficiency [29, 11, 10, 21, 12, 2]. By now (levelled) FHE is even considered a standard cryptographic primitive, which can be based on the standard Learning with Errors (LWE) problem [27] with polynomial modulus-to-noise ratio. An

3rd Conference on Information-Theoretic Cryptography (ITC 2022).
Editor: Dana Dachman-Soled; Article No. 4; pp. 4:1–4:21
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

important feature of FHE is ciphertext compactness, which means that homomorphically evaluated ciphertexts do not grow with the size of the evaluated circuit. Furthermore, a recent line of work [16, 9, 19] has succeeded in achieving FHE with essentially optimal rate, i.e. for sufficiently long messages, the size of ciphertexts is only an additive amount larger than the encrypted plaintext. Thus, we say that these schemes achieve (or approach) plaintext-size to ciphertext-size ratio 1; we call this a rate-1 scheme for short.

Circuit-Private FHE

The standard security notion of FHE, IND-CPA security, guarantees the privacy of encrypted data. But it does not guarantee any concrete security for the evaluator beyond the guarantee that a ciphertext can convey only a limited amount of information about the computation from which it resulted due to compactness. In a *circuit-private* FHE scheme, an evaluator holding a circuit $\mathcal{C}$ has the following security guarantee. Assume that c is a ciphertext encrypting a message x, and assume the evaluator homomorphically evaluates $\mathcal{C}$ on c, resulting in a ciphertext d. The evaluator has the guarantee that d encrypts the output $\mathcal{C}(x)$ of the homomorphic computation but does not convey any further information about the circuit $\mathcal{C}$. We say that an FHE scheme satisfies semi-honest circuit privacy if this property holds for honestly generated keys and ciphertexts. Gentry [18] describes a simple *drowning-based* mechanism to achieve semi-honest circuit privacy (which typically leads to poor parameters for the underlying hardness assumption). Later works [17, 8] provided transformations adding semi-honest circuit privacy with very little overhead and without parameter deterioration.

In essence, circuit privacy can be seen as a property of a specific homomorphic evaluation algorithm. A circuit-private evaluation algorithm must be randomized, while non-circuit private evaluation algorithms can be deterministic.

Ostrovsky, Paskin and Paskin [26] provided the first *maliciously circuit-private FHE scheme.* This scheme was later generalized to the *multi-key setting* by Chongchitmate and Ostrovsky [13]. Malicious circuit privacy requires that the above property holds even for maliciously generated keys and ciphertexts. On a technical level, the notion of malicious statistical circuit privacy requires the existence of an (unbounded) ciphertext extractor, which extracts a plaintext from a given pair of public key and ciphertext, and a simulator which, given an output $\mathcal{C}(x)$ simulates a homomorphically evaluated ciphertext encrypting $\mathcal{C}(x)$. In the presence of a common reference string (CRS), the well-formedness of both keys and ciphertexts can be enforced by requiring keys and ciphertexts to include non-interactive zero-knowledge proofs of knowledge (NIZKPoK) of their well-formedness, such that plaintexts can be extracted using the knowledge extractor for the NIZKPoK.

However, [26] provide a maliciously circuit-private FHE scheme in the plain model (i.e. without CRS) and guarantee *statistical circuit privacy.* The main idea of their construction is to leverage a *conditional disclosure of secrets* protocol [1] instead of NIZK proofs. That is, an input ciphertext c contains additional *encrypted well-formedness information* γ, which they use in the maliciously circuit-private evaluation algorithm to enforce that the output ciphertext d is independent of the circuit $\mathcal{C}$ if c was not well-formed. This well-formedness information γ is *consumed* by the maliciously circuit-private evaluation algorithm, and the output ciphertext d contains no such well-formedness information. Hence, d cannot be used as input for the maliciously circuit-private evaluation algorithm but can still serve as input for standard (non-maliciously-circuit-private) homomorphic evaluation.

We outline the main ideas of [26] in the appendix of this paper's full version.

Multi-Hop FHE

We say that an FHE scheme is *single-hop* if ciphertexts resulting from a homomorphic evaluation cannot be used as input ciphertexts for further homomorphic evaluations. We refer to FHE schemes where homomorphically computed ciphertexts can again be used as input ciphertexts for further homomorphic computation as *multi-hop* (a notion introduced by [20]).

The basic scheme of [26] is only single-hop, but they show how to modify it to support multi-hop (non-maliciously-circuit-private) homomorphic evaluation. By the discussion in the last paragraph, this means that in the multi-hop setting, circuit privacy is only guaranteed if all evaluators are honest. Furthermore, it seems hard to establish that their techniques could yield a scheme that satisfies malicious circuit privacy even if some evaluators are malicious. That is, consider a scenario in the 2-hop setting, where we have a malicious key-generator and encryptor as well as a malicious first evaluator E_1 and an honest second evaluator E_2. The basic issue is that while the techniques of [26] enforce that both keys and ciphertexts produced by the encryptor are well-formed, they cannot provide a similar guarantee for ciphertexts produced by the first evaluator E_1. Consequently, E_1 may pass an arbitrarily malformed ciphertext to E_2. Then all circuit privacy guarantees for E_2 are lost.

1.1 Our Results

This work provides a conceptually simple construction of a fully homomorphic encryption scheme with malicious circuit privacy. As a bonus, ciphertexts generated by the encryption algorithm and ciphertexts produced by the homomorphic evaluation procedure are syntactically the same. This means our scheme supports malicious circuit privacy even if the input ciphertexts themselves are potentially the result of a homomorphic evaluation. Our construction significantly departs from the blueprint of [26]. On a technical level, our constructions build on and leverage rate-1 FHE schemes [19, 9], but also inherit the rate-1 property. As we will explain below, our construction equips a rate-1 FHE scheme with a novel evaluation algorithm but otherwise leave the underlying construction unmodified and is black-box in the underlying rate-1 FHE scheme. This means, in particular, that our maliciously circuit-private evaluation algorithm also supports input-ciphertexts which themselves are the result of homomorphic evaluations. We call such a scheme a *multi-hop-secure* maliciously circuit-private FHE scheme. Note that this property solely comes down to the type of input-ciphertext supported by the maliciously circuit-private homomorphic evaluation algorithm but otherwise leaves the definition of malicious statistical circuit-privacy unchanged.

Compared to the construction of [26], our construction can be considered a more direct way of achieving malicious circuit privacy.

1.2 Applications

We will briefly discuss two related applications we envision as use-cases for our multi-hop-secure MCP-FHE scheme.

- **Encrypted Databases with privacy for Write-Queries:** Consider a scenario where a cloud server holds a database encrypted under an FHE scheme. The owner of the database, who generated the FHE keys goes offline, but several mutually mistrusting workers perform homomorphic computations on the database, and these computations involve sensitive data held by the workers. While the IND-CPA security of the FHE scheme protects the privacy of the database, the privacy of the workers' operations is ensured by the circuit privacy of the FHE scheme. However, if a malicious database owner and several malicious workers collude against a worker, then single-hop circuit privacy does not offer any guarantee to this worker. Consequently, to protect the privacy of this worker's operation, we need a multi-hop-secure MCP-FHE scheme.

- **Federated Learning with Model-Privacy:** In the machine-learing subfield of federated learning [25], the training data is distributed among several (physically) separated servers. A central server, coordinating a learning process sends partially-trained models to the training servers, who compute model-updates using their local training data and send the updates back to the central server. The purpose of this separation of the training data is two-fold. First, by ware-housing the training-data locally with the servers and only communicating (relatively small) model updates, an enormous amount of bandwidth can be saved which would otherwise be needed to transfer vast quantities of training data. Second, and maybe more importantly, each server is in control of the amount of outgoing data and therefore has the guarantee that his local data cannot be retrieved entirely by the central server.

  Now consider a scenario where a model-owner, in possession of a partially trained model, wants the training servers to compute updates on his model. However, the model may contain sensitive data which should not be leaked to the training servers. Consequently, encrypting the model under an FHE scheme protects the privacy of this model. To protect the privacy of the training servers' training data, we need to require circuit privacy. However, if the model owner colludes with some of the training servers, standard malicious circuit privacy is insufficient to protect the privacy of any of the training servers training-data. By using a multi-hop-secure MCP-FHE scheme, the training servers have the guarantee that even if the model owner colludes with other users, they will not learn more about this users data than they would have in a plain federated learning protocol (i.e. without the additional layer of homomorphic encryption).

1.3 Technical Outline of our Approach

Our construction significantly departs from the OPP approach [26]. On a very high level, our approach is to augment a given FHE scheme to *natively* support malicious function privacy for a very basic class of functions, namely affine functions, without resorting to tools which enforce the well-formedness of input ciphertexts. We will then be able to amplify this to the class of all functions by relying on the machinery of affine randomized encodings [22, 5], aka information-theoretically secure garbled circuits.

Statistically Sender-Private OT from High-Rate OT

We will first describe how a *high-rate* FHE scheme can be augmented to support malicious function privacy for affine functions. As described above, such high-rate FHE schemes were recently constructed by Gentry and Halevi [19] and Brakerski et al. [9].

Our starting point is a recent work of Badrinarayanan et al [6], who observed that high rate (sender-input to sender-message ratio) can be leveraged to achieve statistical sender privacy. This is similar in spirit to the work of [14], who build an OT protocol in the bounded-quantum-storage model. In more detail, [6] observed that any string-OT with *high rate* (i.e. greater than $1/2$) yields a statistically sender private OT protocol (called weak OT in [6]) via a simple information-theoretic transformation. Specifically, the high-rate OT is used to transfer two random strings r_0 and r_1. But since the OT has high rate, the OT-sender message ot_2 is shorter than the concatenation of the two random strings. Consequently, one can argue that one of the two strings r_0 and r_1 must have high conditional min-entropy given ot_2. Thus, using a suitable randomness extractor Ext, one can derive two masks $k_0 = \mathsf{Ext}(r_0, s_0)$ and $k_1 = \mathsf{Ext}(r_1, s_1)$ (for two seeds s_0 and s_1) and argue that either k_0 or k_1 must be statistically close to uniform conditioned on ot_2. The sender then also sends $(m_0 \oplus k_0, m_1 \oplus k_1)$, i.e. the actual messages blinded with the corresponding mask. An honest receiver will then be able to recover the m_b corresponding to his choice-bit b.

Note that this argument did not assume the well-formedness of the OT-sender message ot_1[1]. So consequently, no matter how malformed ot_1 is, the message ot_2 *must lose information* about either r_0 or r_1, and consequently one of the masks k_0, k_1 is uniformly random from the view of the receiver.

While the high-level idea of the proof and the statement of the corresponding theorem in [6] is true, there is a subtle loophole in their proof, which we will briefly explain here. To establish malicious statistical sender privacy, one needs to show the existence of an (unbounded) extractor which extracts the receiver's choice bit from the ot_1 message. In [6], this is achieved via the following argument: For a fixed ot_2 it holds that $H_\infty(r_0, r_1 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) \geq n$, thus it must either hold that $H_\infty(r_0 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) > n/2$ or $H_\infty(r_1 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) > n/2$. The unbounded extractor then computes both $h_b = H_\infty(r_b | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2)$ for $b \in \{0, 1\}$, and sets the extracted bit b^* to 0 if $h_0 < h_1$, otherwise to 1.

This reasoning assumes that conditional min-entropy obeys a chain-rule, i.e. the conditional min-entropy of (r_0, r_1) must split into the conditional min-entropies of r_0 and r_1. However, in general this is not the case. There are (contrived) choices of the "leakage function" $\mathsf{OT}_2(ot_1, \cdot, \cdot)$, for which even though $H_\infty(r_0, r_1 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) > n$, it holds that

$$H_\infty(r_0 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) = H_\infty(r_1 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2) \approx 1,$$

i.e. even (r_0, r_1) have n bits of min-entropy, each of them individually only has a single bit of min-entropy[2].

Essentially, the problem is that it might depend on (r_0, r_1) which one of r_0 or r_1 is leaked by $\mathsf{OT}_2(ot_1, r_0, r_1)$, i.e. the choice of the bit b is not necessarily fixed by the function $\mathsf{OT}_2(ot_1, \cdot, \cdot)$ as implicitly assumed in the above argument. In other words, the function $\mathsf{OT}_2(ot_1, \cdot, \cdot)$ does not fix a choice bit b, but rather a *distribution of choice-bits* $b(r_0, r_1)$ which may depend on r_0, r_1 in arbitrary ways.

Consequently, a more involved extraction strategy is required to make the proof rigorous. This can indeed be achieved by resorting to the *min-entropy splitting lemma* of [14]. In essence, translated to our context, this lemma states that for every leakage function $\mathsf{OT}_2(ot_1, \cdot, \cdot)$ there does exist an explicit random variable $b = b(r_0, r_1)$ such that $H_\infty(r_b | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2, b) > n/2 - 1$[3].

Thus, we can adapt the extractor of [6] to extract based on the conditional min-entropies $H_\infty(r_0 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2, b = 0)$ and
$H_\infty(r_1 | \mathsf{OT}_2(ot_1, r_0, r_1) = ot_2, b = 1)$ and make the proof strategy of [6] work.

[1] Indeed, we haven't even mentioned it yet.

[2] Example: If first bit of r_0 is 0, leak last $n-1$ bits of r_0, otherwise leak last $n-1$ bits of r_1. See also [24, 28].

[3] The actual statement holds for smooth min-entropy, but we omit this somewhat technical detail for the sake of this outline.

FHE with Statistical Function Privacy for Affine Functions

Our core-observation is that this very same approach also works if we replace the high-rate OT by a high-rate FHE scheme. As explained above, such FHE schemes with a rate approaching 1 were recently constructed in [19] and [9].

We remark that these schemes have two different ciphertext types. Type 1 ciphertexts are *decompressed* and allow for homomorphic operations, but these ciphertexts have a poor rate, as each ciphertext encrypts (say) just a single bit[4]. Type 2 ciphertexts are in a *compressed format*, and each ciphertext encrypts say ℓ bits, and these ciphertexts have a rate approaching 1, but do not support homomorphic computations. These have a public compression procedure, which takes a vector of ℓ type 1 ciphertexts and produces a single type 2 ciphertext. Likewise, there is a public decompression procedure which takes a single type 2 ciphertext and returns a vector of ℓ type 1 ciphertexts. We remark that compressing type 1 into type 2 ciphertexts is fairly efficient, but decompressing type 2 into type 1 ciphertexts involves a rather expensive bootstrapping operation in current schemes [19, 9].

In essence, we will harness the compress operation to *lose information* about strings which should remain private. Specifically, assume we have such a compressible FHE scheme Π. Now let $c = \mathsf{Enc}(\mathsf{pk}, b)$ be a ciphertext encrypting a bit b under Π. We obtain malicious statistical function privacy for affine functions via the following evaluation procedure, which mimics an oblivious transfer in Π. The evaluator chooses two uniformly random strings $r_0, r_1 \in \{0,1\}^\ell$ and evaluates the affine function $f(x) = x \cdot r_1 + (1-x) \cdot r_0$ on c, obtaining an encryption of $c' = \mathsf{Enc}(f(b))$. The ciphertext c' is of type 1 and has thus low rate. The evaluator now compresses c' into a high-rate type 2 ciphertext and immediately decompresses it into a type 1 ciphertext d, which is an encryption of r_b. As above, the evaluator now chooses two extractor seeds s_0 and s_1 and computes $v_0 = m_0 \oplus \mathsf{Ext}(k_0, s_0)$ and $v_1 = m_1 \oplus \mathsf{Ext}(k_1, s_1)$. Finally, It homomorphically evaluates the function $g(x,y) = (\mathsf{Ext}(y, s_1) \oplus v_1) \cdot x + (\mathsf{Ext}(y, s_0) \oplus v_0) \cdot (1-x)$ on the ciphertexts c and d, obtaining an encryption e of

$$\begin{aligned} g(b, r_b) &= (\mathsf{Ext}(r_b, s_1) \oplus \mathsf{Ext}(r_1, s_1) \oplus m_1) \cdot b \\ &\quad + (\mathsf{Ext}(r_b, s_0) \oplus \mathsf{Ext}(r_0, s_0) \oplus m_0)(1-b) \\ &= m_b, \end{aligned}$$

and the ciphertext e is the output of the homomorphic evaluation.

Thus, correctness follows from the derivation above. To argue statistical function privacy, we argue analogously as in the last paragraph. Namely, even if both the public key and the ciphertext c are arbitrarily malformed, we observe that when we compress c' into a type 2 ciphertext, call it $\hat{c}$, then since $\hat{c}$ is high-rate, it cannot fully determine both r_0 and r_1. Consequently, as in the argument above, either r_0 or r_1 must have high conditional min-entropy given $\hat{c}$[5]. Since d is computed from $\hat{c}$, the same holds for d, i.e. conditioned on d either r_0 or r_1 has high min-entropy. Consequently, by the extraction property of Ext either v_0 or v_1 is statistically close to uniform conditioned on d. Thus, e does not depend on both m_0 and m_1. To make the argument formal, we can argue as above that a bit b can be extracted from the ciphertext c (via an unbounded extractor) and that the output ciphertext e can be simulated given only m_b.

[4] In both [19] and [9] the ciphertexts in this mode are essentially GSW ciphertexts [21]

[5] Where the same caveat as above applies, i.e. we need to condition on an additional *spoiling bit b*.

Note that our construction makes no additional non-black-box of underlying cryptographic primitives beyond whatever the underlying FHE scheme does. That is, given the current high-rate FHE constructions [19, 9] the only operation in the above construction which needs to do any heavy lifting is the decompression step, which in these constructions involves a bootstrapping operation.

We remark, however, that even though bootstrapping involves making non-black-box use of the decryption circuit of the underlying FHE scheme. This non-black-box use typically comes to just performing a *rounding operation* homomorphically. Furthermore, it is conceivable that there might exist construction of high-rate FHE schemes which deviate from the blueprint of [19, 9] and do not rely on bootstrapping to achieve high rate.

Malicious Statistical Circuit Privacy for NC1 Circuits

We will now outline how malicious statistical circuit privacy for affine functions can be amplified to malicious statistical circuit privacy for NC1 circuits. The go-to tool to achieve this are decomposable affine randomized encodings (DARE), also known as garbled circuits. A garbling scheme allows us to encode a computation into an affine and a non-affine part. For any input it holds that the output of the affine part together with the non-affine part does not leak more than the result of this computation on this input. Information-theoretically DAREs are known for NC1 circuits (i.e. circuits of logarithmic depth) [23, 22, 5]. Randomized encodings have, e.g. been used to bootstrap KDM security for affine functions to KDM security for bounded-size circuits [3].

We make use of DAREs/GCs as follows, starting with an FHE scheme with malicious function privacy for affine functions as described in the previous paragraph. Assume that the evaluator wants to homomorphically evaluate an NC1 circuit $\mathcal{C}$ on a potentially maliciously generated input ciphertext c. First, the evaluator computes a randomized encoding of $\mathcal{C}$ consisting of an affine part T and a non-affine part $\tilde{\mathcal{C}}$. Then, it evaluates the affine function T on the ciphertext c using the maliciously function private evaluation procedure for affine functions, resulting in a ciphertext d. Finally, it evaluates the non-affine part $\tilde{\mathcal{C}}$ on d, resulting in an output ciphertext e. Correctness follows immediately from the correctness of the FHE scheme and the DARE. To argue malicious circuit privacy, first note that by the malicious function privacy for affine functions, the ciphertext d does not leak more than $T(x)$ (where x is the value which can be extracted from c) about T. Consequently, it holds that e does not leak more than $T(x)$ and $\tilde{\mathcal{C}}$ about $\mathcal{C}$, which by the security of the DARE scheme does not leak more than $\mathcal{C}(x)$.

We remark that in our construction the output ciphertext e potentially leaks the same information about the circuit $\mathcal{C}$ that $T(x)$ and $\tilde{C}$, i.e. essentially the size of $\mathcal{C}$. This is somewhat in contrast to the construction of [26], which ensures that no information about the evaluator's circuit is leaked. Whether leaking the size of the evaluator's circuit is inherent in multi-hop-secure MCP-FHE remains an (in our opinion interesting) open problem.

Malicious Statistical Circuit Privacy for all Circuits

We will briefly outline how the above techniques can be leveraged to handle arbitrary polynomial depth circuits. To achieve this, we will resort to an idea of Kilian [23]. Specifically, given a polynomial-depth circuit $\mathcal{C}$, we will slice $\mathcal{C}$ into layers $\mathcal{C}_1, \dots, \mathcal{C}_k$ such that each $\mathcal{C}_i$ is an NC1 circuit and $\mathcal{C} = \mathcal{C}_k \circ \cdots \circ \mathcal{C}_1$ (i.e. we can evaluate $\mathcal{C}$ by sequentially evaluating the $\mathcal{C}_i$). The circuits $\mathcal{C}_i$ can now be evaluated using the techniques described in the previous section. However, this basic idea has an issue as the intermediate outputs of the $\mathcal{C}_i$ are not protected and may therefore leak information about the $\mathcal{C}_i$ and therefore $\mathcal{C}$. To deal with this issue, we will replace the circuits $\mathcal{C}_i$ by circuits D_i which *encrypt their output wires* using a one-time pad. Specifically, the circuit D_1 first computes $\mathcal{C}_1$, but xors a one-time pad K_1 on the output, i.e. $D_1(x) = \mathcal{C}_1(x) \oplus K_1$. The circuit D_2 first decrypts its input using the key K_1 and encrypts its output using a key K_2, i.e. $D_2(x) = \mathcal{C}_2(x \oplus K_1) \oplus K_2$. We continue in the same fashion, until we reach D_k which computes $D_k(x) = \mathcal{C}_k(x \oplus K_{k-1})$. By the security of the one-time pad, the outputs of the D_i leak no information about the outputs of the $\mathcal{C}_i$.

We will further show that if one is willing to settle for computational rather than statistical circuit privacy, then the transformation described in the previous paragraph can be implemented using computational garbled circuits, which means that the most expensive step, the function private evaluation of the affine function, only needs to be performed once. In this setting, some care has to be taking in the security proof as our input-extractor is unbounded but security of the garbled circuits only holds computationally. However, this issue can be dealt with using a standard trick which moves the information obtained by the unbounded extractor into non-uniform advice, which is provided to the non-uniform reduction against the garbling scheme.

This concludes the overview.

Roadmap

In Section 2 we show how to turn any high-rate FHE into one, which allows for circuit private evaluation of affine functions. We use this in Section 3 to build a circuit private scheme for NC1, which we extend to arbitrary circuits in Section 4. We cover the preliminaries in Appendix A.

For more information see the full version of the paper.

2 OT from High-Rate LHE

Here we reiterate the statistical sender private OT of [6] with slight modifications in notation and sender-privacy proof. It transforms a high-rate linearly homomorphic encryption scheme (LHE) into a statistically sender private OT.

2.1 Construction of [6]

Let ($\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}$) be a high-rate LHE scheme where the messages are vectors over $\{0,1\}$. We will use the following circuit $\mathcal{C}$ where strings r_0 and r_1 are hard-wired into the circuit, and one of them is selected according to input bit b. Notice, this circuit is a linear function over $\{0,1\}$.

Circuit $\mathcal{C}[r_0, r_1](b)$:

- output r_b

Now follows the construction. In this construction n is the size of the messages m_0, m_1 and the parameter m is dependent on λ but can be chosen arbitrarily large.

$\mathsf{OT}_1(1^\lambda, b)$:

- Generate keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
- Let $c \leftarrow \mathsf{Enc}(\mathsf{pk}, b)$
- return (pk, c)

$\mathsf{OT}_2(1^\lambda, ot_1 = (\mathsf{pk}, c), m_0, m_1)$:

```
- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire $r_0$, $r_1$ into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- return $s_0$, $s_1$, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, $\mathsf{Ext}(s_1, r_1) \oplus m_1$, $e$, and $\mathsf{Eval}(\mathcal{C}', c)$
```

In the output, c is an encryption of b and $\mathsf{Eval}(\mathcal{C}', c)$ an encryption of r_b.

$\mathsf{OT}_3(\mathsf{sk}, ot_2)$:

```
- Let $s_0$, $s_1$, $x_0$, $x_1$, $c$, and $e$ be the content of the message $ot_2$
- Let $b \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
- Let $r_b \leftarrow \mathsf{Dec}(\mathsf{sk}, e)$
- return $x_b \oplus \mathsf{Ext}(s_b, r_b)$
```

2.2 Correctness

Since ($\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}$) is correct c is a correct encryption of b in that scheme. OT_2 then outputs s_0, s_1, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, and $\mathsf{Ext}(s_1, r_1) \oplus m_1$ together with correct encryptions of b and r_b. In OT_3 we then decrypt b and r_b. Because Ext is deterministic (with a fixed seed s_b) we can reconstruct $m_b = m_b \oplus \mathsf{Ext}(s_b, r_b) \oplus \mathsf{Ext}(s_b, r_b)$.

2.3 Computational Receiver's Security

The sender only ever sees encryptions of the receivers input b and the public key of the LHE. Therefore, if the sender can learn anything about b he can also break the CPA security of the LHE.

2.4 Statistical Sender's Security

▶ **Theorem 1.** *Let* (KeyGen, Enc, Eval, Dec) *be an LHE with high rate, then* (OT_1, OT_2, OT_3) *as detailed in Subsection 2.1 is a statistically sender private OT protocol.*

Proof. In the following, we show an unbounded simulator Sim that does not know m_0 or m_1 but has one-time access to an oracle for the function $f(b) = m_b$. With this oracle access, she produces an output which is statistically close to the output of OT_2, which has full access to r_0 and r_1.

$\mathsf{Sim}^f(ot_1 = (\mathsf{pk}, c))$:

```
- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire $r_0$, $r_1$ into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- Let $e \leftarrow \mathsf{Eval}(\mathcal{C}', c)$
- Let $C$ be the value such that $H_\infty(R_{1-C}|C, E)$ is minimal with $C$ being chosen as in
  corollary 31.
- Query the oracle $f$ for $m_C$
- Choose $S_{1-C} \leftarrow_\$ \{0,1\}^n$ uniformly at random
- If $C = 0$:
    - return $s_0$, $s_1$, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, $S_{1-C}$, $c$, and $e$
- Else:
    - return $s_0$, $s_1$, $S_{1-C}$, $\mathsf{Ext}(s_1, r_1) \oplus m_1$, $c$, and $e$
```

We now use a hybrid argument to show that the above construction is statistically sender private. H_0 is the honest execution of the protocol.

$\boldsymbol{H_0(\mathsf{pk}, c, m_0, m_1)}$**:**

- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire r_0, r_1 into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- return s_0, s_1, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, $\mathsf{Ext}(s_1, r_1) \oplus m_1$, c, and $\mathsf{Eval}(\mathcal{C}', c)$

In hybrid H_1 we replace $\mathsf{Ext}(s_{1-C}, r_{1-C})$ by a uniformly random S_0 of same size.

$\boldsymbol{H_1(\mathsf{pk}, c, m_0, m_1)}$**:**

- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire r_0, r_1 into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- Let $e \leftarrow \mathsf{Eval}(\mathcal{C}', c)$
- Let C be the value such that $H_\infty(R_{1-C}|C, E)$ is minimal with C being chosen as in corollary 31.
- Choose $S_{1-C} \leftarrow_\$ \{0,1\}^n$ uniformly at random
- If $C = 0$:
  - return s_0, s_1, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, $S_{1-C} \oplus m_1$, c, and e
- Else:
  - return s_0, s_1, $S_{1-C} \oplus m_0$, $\mathsf{Ext}(s_1, r_1) \oplus m_1$, c, and e

---

In H_2 we remove the real sender inputs.

$\boldsymbol{H_2^f(\mathsf{pk}, c)}$**:**

- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire r_0, r_1 into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- Let $e \leftarrow \mathsf{Eval}(\mathcal{C}', c)$
- Let C be the value such that $H_\infty(R_{1-C}|C, E)$ is minimal with C being chosen as in corollary 31.
- Query the oracle f for m_C
- Choose $S_{1-C} \leftarrow_\$ \{0,1\}^n$ uniformly at random
- If $C = 0$:
  - return s_0, s_1, $\mathsf{Ext}(s_0, r_0) \oplus m_0$, S_{1-C}, c, and e
- Else:
  - return s_0, s_1, S_{1-C}, $\mathsf{Ext}(s_1, r_1) \oplus m_1$, c, and e

---

Now we argue why the hybrids are statistically close.

$\boldsymbol{H_0 \approx H_1}$**:**

In H_1 we replace $\mathsf{Ext}(s_{1-C}, r_{1-C})$ by a uniformly random chosen S_{1-C}. Here we argue that the statistical distance between the two hybrids is negligible using 31.

Lemma 30 gives that

$$\begin{aligned} H_\infty(R_0, R_1|E = e) &> H_\infty(R_0, R_1) - log(1/\mathsf{Pr}[E = e]) \\ &\geq 2m - |e| \end{aligned}$$

Then corollary 31 gives that

$$H_\infty^\varepsilon(R_{1-C}|C, E = e) > (2m - |e|)/2 - 1 - log(1/\varepsilon)$$

for any ε. Then the smooth min-entropy conversion lemma 32 gives that

$$H_\infty(R_{1-C}|C, E = e) \geq -log(2^{-(2m-|e|)/2-1-log(1/\varepsilon)} + \varepsilon)$$

In the following, this number will be called α. Notice that α can only be positive if $2m - |e|$ is positive and e encrypts a message of size m. Therefore, the rate ρ need to be bigger than 1/2 (i.e. $1/2 < \rho = m/|e|$).

Then we use the property of the extractor to ensure that $\mathsf{Ext}(s_{1-C}, r_{1-C})$ is statistically close to uniform (i.e. $SD(\mathsf{Ext}(s_{1-C}, r_{1-C}), S_{1-C}) \leq \varepsilon'$). Clearly, this can be reached if the rate $\rho > 1/2$. Therefore, the statistical distance between H_0 and H_1 is at most ε'.

$H_1 \approx H_2$:

In this hybrid, we altogether remove m_{1-C} which we can do because it is being XORed with a uniformly random string and therefore is perfectly hidden. Thus, H_1 and H_2 are identically distributed in this case. ◀

2.5 FHE with Circuit-Private OT Evaluation

Here, we show how to add a evaluation procedure $\mathsf{Eval}_{\mathsf{OT}}$ to a high-rate FHE, which can evaluate choice functions in a circuit private manner.

The construction is the same as for the OT above but the message reconstruction of OT_3 is done on the sender's side. Again, we use circuit $\mathcal{C}$

Circuit $\mathcal{C}[r_0, r_1](b)$:

- output r_b

But we also use circuit $\tilde{\mathcal{C}}$ which except for decrypting takes the role of OT_3

Circuit $\tilde{\mathcal{C}}[s_0, s_1, x_0, x_1](b, r_b)$:

- output $x_b \oplus \mathsf{Ext}(s_b, r_b)$

$\mathsf{Eval}_{\mathsf{OT}}(1^\lambda, \mathsf{pk}, m_0, m_1, c)$:

- Choose $s_0, s_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Choose $r_0, r_1 \leftarrow_\$ \{0,1\}^m$ uniformly at random
- Hardwire r_0, r_1 into $\mathcal{C}[r_0, r_1]$ to get circuit $\mathcal{C}'$
- Let $e \leftarrow \mathsf{Eval}(1^\lambda, \mathsf{pk}, \mathcal{C}', c)$
- Hardwire s_0, s_1, $x_0 = \mathsf{Ext}(s_0, r_0) \oplus m_0$, and $x_1 = \mathsf{Ext}(s_0, r_0) \oplus m_1$ into $\tilde{\mathcal{C}}[s_0, s_1, x_0, x_1]$ to get circuit $\tilde{\mathcal{C}}'$
- return $\mathsf{Eval}(1^\lambda, \mathsf{pk}, \tilde{\mathcal{C}}', (c, e))$

Correctness and receiver's security (in this case CPA security) stay the same as before. For circuit privacy (previously sender privacy) we now need to argue over the compression in e. The last step in $\mathsf{Eval}_{\mathsf{OT}}$ can be thought of as post-processing and does not change anything about the circuit privacy.

3 Circuit-Private NC1-HE from FHE with OT

An OT is similar to a circuit private HE for affine functions. We use Decomposeable Affine Randomized Encodings (DARE) to increase the set of function that we can evaluate with circuit privacy to all functions in NC1. We achieve this by letting the OT do the affine operations and then evaluate the DARE inside another layer of FHE.

3.1 Construction

Let (KeyGen′, Enc′, Eval′, Dec′) be an FHE with circuit private choice function evaluation procedure $\mathsf{Eval}'_{\mathsf{OT}}$ and (Garble, GarbleInput, Ev) be a ϕ-private DARE. In this construction we use a circuit $\mathcal{C}$ with hardcoded garbled function F which simply evaluates the garbled function on the input.

```
$\mathcal{C}\ [F](d = (d_i)_{i\in[n]})$:
    return $\mathsf{Ev}(F, (d_i)_{i\in[n]})$
```

The construction then is:

```
$\mathsf{KeyGen}(1^\lambda)$:
    return $\mathsf{KeyGen}'(1^\lambda)$
$\mathsf{Enc}(\mathsf{pk}, m)$:
    return $\mathsf{Enc}'(\mathsf{pk}, m)$
$\mathsf{Eval}(1^\lambda, \mathsf{pk}, f, c = (c_i)_{i\in[n]})$:
    $(F, (r_{i,j})_{i\in[n], j\in\{0,1\}}) \leftarrow \mathsf{Garble}(f, 1^\lambda)$
    For each $i \in [n]$ let $z_i \leftarrow \mathsf{Eval}'_{\mathsf{OT}}(1^\lambda, \mathsf{pk}, r_{i,0}, r_{i,1}, c_i)$
    Hardwire $F$ into $\mathcal{C}[F]$ to get the circuit $\mathcal{C}'$
    return $\mathsf{Eval}'(1^\lambda, \mathsf{pk}, \mathcal{C}', z = (z_i)_{i\in[n]})$
$\mathsf{Dec}(\mathsf{sk}, c)$:
    return $\mathsf{Dec}'(\mathsf{sk}, c)$
```

First Eval garbles f and then emulates the encoding mechanism GarbleInput inside of the FHE with the help of $\mathsf{Eval}_{\mathsf{OT}}$. This works because the GarbleInput is a choice function which is exactly what an OT calculates. With the encoded input and the garbled circuit F we run the Ev function inside the FHE and will only be able to leak as much information about the function as $(F, \mathsf{GarbleInput}(r, m))$ would have.

The correctness of (KeyGen, Enc, Dec, Eval) follows routinely from the correctness of (Garble, GarbleInput, Ev), and (KeyGen′, Enc′, Eval′, $\mathsf{Eval}'_{\mathsf{OT}}$, Dec′). Likewise, CPA security of (KeyGen, Enc, Dec, Eval) follows routinely from the CPA security of (KeyGen′, Enc′, Eval′, $\mathsf{Eval}'_{\mathsf{OT}}$, Dec′).

3.2 Malicious Statistical Circuit Privacy

▶ **Theorem 2.** *Let (*KeyGen′*,* Enc′*,* Eval′*) be an FHE with circuit private choice function evaluation procedure* $\mathsf{Eval}'_{\mathsf{OT}}$ *and (*Garble*,* GarbleInput*,* Ev*) be a ϕ-private DARE (for some function ϕ) then the NC1-HE as detailed in Subsection 3.1 is ϕ-circuit-private.*

The proof of the theorem is in the full version of the paper.

3.3 Computational Circuit Privacy

If we use a computationally ϕ-private garbled circuit in this transformation instead of its information theoretical counterpart we instantly get an FHE which is ϕ-circuit-private against computational adversaries. Nothing about the construction needs to change; we only need to adjust the proof as detailed in the full version.

3.4 Multi-Hop-Security

Since evaluating does not change the structure of the ciphertexts the $NC1$-HE inherits the multi-hop-security property from the FHE (if the FHE is multi-hop then the $NC1$-HE is as well).

4 Circuit-Private FHE from Circuit-Private NC1-HE

To build a circuit-private FHE from a Circuit-Private NC1-HE, we go back to techniques from Kilian's classic paper [23]. On a high level, we split up the circuit into NC1 circuits and encrypt the connecting wires with the one-time pad.

Assume we want to evaluate a circuit $\mathcal{C}$ of polynomial depth. We show an example of this in Figure 1.

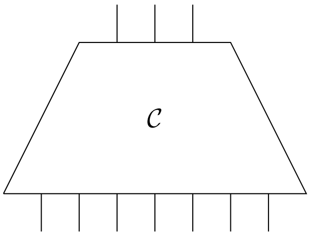

Figure 1 Circuit $\mathcal{C}$.

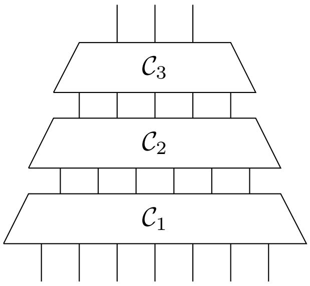

Figure 2 Circuit $\mathcal{C}$ split into subcircuits $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$. We chose three subcircuits for illustrative reasons. The amount of subcircuits depends on the depth of circuit $\mathcal{C}$.

We split up that circuit into subcircuits of depth $log(\lambda)$ such that they are $NC1$ circuit (as in Figure 2). If the circuit-private $NC1$-HE scheme is multi-hop, we can then evaluate each of these subcircuits sequentially in a circuit-private manner. This construction is an FHE scheme which leaks the depth of the circuit and the intermediate values.

We can, however, encrypt these intermediate values with a one-time pad and then decrypt it in the next subcircuit. We demonstrate this modification of the circuit in Figure 3.

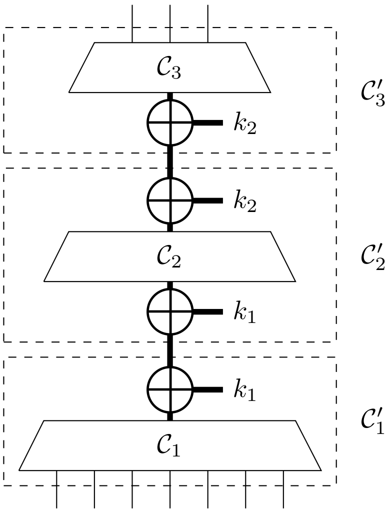

Figure 3 Subcircuits of $\mathcal{C}$ together with OTP encryption and decryption. Each thick wire represents a collection of wires. We use the circuits $\mathcal{C}'_1$, $\mathcal{C}'_2$, and $\mathcal{C}'_3$.

This is possible because encrypting and decrypting the one-time pad is incredibly (computationally) cheap. Therefore, the subcircuits combined with encryption and decryption are still in $NC1$. This way the intermediate values are statistically hidden.

The result is an FHE scheme, which is $\Phi_{depth,width}$ circuit private. $\Phi_{depth,width}$ leaks the depth of the circuit and the size of the intermediate values.

References

1 William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135. Springer, 2001. doi:10.1007/3-540-44987-6_8.

2 Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 297–314. Springer, 2014. doi:10.1007/978-3-662-44371-2_17.

3 Benny Applebaum. Key-dependent message security: Generic amplification and completeness. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 527–546. Springer, 2011. doi:10.1007/978-3-642-20465-4_29.

4 Benny Applebaum. Garbled circuits as randomized encodings of functions: a primer. *IACR Cryptol. ePrint Arch.*, page 385, 2017. URL: http://eprint.iacr.org/2017/385.

5 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc^0. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 166–175. IEEE Computer Society, 2004. doi:10.1109/FOCS.2004.20.

6 Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303. Springer, 2017. doi:10.1007/978-3-319-70700-6_10.

7 Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 784–796. ACM, 2012. doi:10.1145/2382196.2382279.

8 Florian Bourse, Rafaël Del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 62–89. Springer, 2016. doi:10.1007/978-3-662-53008-5_3.

9 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437. Springer, 2019. doi:10.1007/978-3-030-36033-7_16.

10 Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 309–325. ACM, 2012. doi:10.1145/2090236.2090262.

11 Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 97–106. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.12.

12 Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In Moni Naor, editor, *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 1–12. ACM, 2014. doi:10.1145/2554797.2554799.

13 Wutichai Chongchitmate and Rafail Ostrovsky. Circuit-private multi-key FHE. In Serge Fehr, editor, *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part II*, volume 10175 of *Lecture Notes in Computer Science*, pages 241–270. Springer, 2017. doi:10.1007/978-3-662-54388-7_9.

14 Ivan Damgård, Serge Fehr, Renato Renner, Louis Salvail, and Christian Schaffner. A tight high-order entropic quantum uncertainty relation with applications. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 360–378. Springer, 2007. doi:10.1007/978-3-540-74143-5_20.

15 Yevgeniy Dodis, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. Springer, 2004. doi:10.1007/978-3-540-24676-3_31.

16 Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2019. doi:10.1007/978-3-030-26954-8_1.

17 Léo Ducas and Damien Stehlé. Sanitization of FHE ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 294–310. Springer, 2016. doi:10.1007/978-3-662-49890-3_12.

18 Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178. ACM, 2009. doi:10.1145/1536414.1536440.

19 Craig Gentry and Shai Halevi. Compressible FHE with applications to PIR. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 438–464. Springer, 2019. doi:10.1007/978-3-030-36033-7_17.

20 Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. *i*-hop homomorphic encryption and rerandomizable yao circuits. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172. Springer, 2010. doi:10.1007/978-3-642-14623-7_9.

21 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, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013. doi:10.1007/978-3-642-40041-4_5.

22 Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 294–304. IEEE Computer Society, 2000. doi:10.1109/SFCS.2000.892118.

23 Joe Kilian. Founding cryptography on oblivious transfer. In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 20–31. ACM, 1988. doi:10.1145/62212.62215.

24 Stephan Krenn, Krzysztof Pietrzak, and Akshay Wadia. A counterexample to the chain rule for conditional HILL entropy - and what deniable encryption has to do with it. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 23–39. Springer, 2013. doi:10.1007/978-3-642-36594-2_2.

25 Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.*, 37(3):50–60, 2020. doi:10.1109/MSP.2020.2975749.

26 Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 536–553. Springer, 2014. doi:10.1007/978-3-662-44371-2_30.

27 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93. ACM, 2005. doi:10.1145/1060590.1060603.

28 Maciej Skórski. Strong chain rules for min-entropy under few bits spoiled. In *IEEE International Symposium on Information Theory, ISIT 2019, Paris, France, July 7-12, 2019*, pages 1122–1126. IEEE, 2019. doi:10.1109/ISIT.2019.8849240.

29 Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer, 2010. doi:10.1007/978-3-642-13190-5_2.

30 Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986. doi:10.1109/SFCS.1986.25.

A Appendix: Preliminaries

In this appendix, we define the concepts and notation that we use in the paper.

A.1 Notation

Assignments

Assignment of a value to a variable is denoted by $\leftarrow$ and $\leftarrow_{\$}$ is used for choosing a value from a set uniformly at random.

Negligible Functions

A function $f : \mathbb{N} \to \mathbb{R}$ is negligible in λ if there exists no positive polynomial p such that $f(\lambda) < \frac{1}{p(\lambda)}$ for all but finitely many λ.

Logarithms

The base of every logarithm in this document is 2.

Circuits

Typical implementations of FHE evaluate using circuit representation for functions. Therefore, we create circuits and then evaluate them. If $\mathcal{C}[a]$ is a circuit, a is a value which we hardwire into the circuit. The input size of a circuit $\mathcal{C}$ is called $in(\mathcal{C})$.

A.2 Public-Key Encryption Schemes

A public-key encryption scheme uses two keys, a public key pk and a secret key sk. We use the public key to encrypt messages, the result of which is called ciphertext. Without knowledge of the secret key, it is virtually impossible to recover the message from the ciphertext. The secret key, however, enables the holder to reliably retrieve the message from the ciphertext.

▶ **Definition 3** (Public-Key Encryption). *The following PPT algorithms describe a public-key encryption scheme:*

KeyGen(1^λ)*: The key-generation algorithm takes the security parameter λ as input and outputs a key pair* $(\mathsf{pk}, \mathsf{sk})$*.*

Enc(pk, m)*: The encryption algorithm takes a public key* pk *and a message m as input and outputs a ciphertext c.*

Dec(sk, c)*: The decryption algorithm takes a secret key* sk *and a ciphertext c as input and outputs a message m. It rarely requires randomness.*

In the rest of the document, every encryption scheme will be public key.

▶ **Definition 4** (Correctness). *An encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is correct if for all message m and security parameters λ and* $(\mathsf{pk}, \mathsf{sk})$ *in the range of* $\mathsf{KeyGen}(1^\lambda)$ *we have* $m = \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m))$

The most popular notion of security for encryption schemes is CPA security (also known as IND-CPA security or semantic security).

▶ **Definition 5** (CPA Security). *An encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is cpa secure if for all PPT adversary pairs* $(\mathcal{A}_1, \mathcal{A}_2)$

$$\left| Pr\left[b = b' \middle| \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{pk}) \\ b \leftarrow_{\$} \{0,1\} \\ b' \leftarrow \mathcal{A}_2(\mathsf{Enc}(\mathsf{pk}, m_b), \sigma) \end{array} \right] - \frac{1}{2} \right|$$

is negligible in λ.

A.3 Homomorphic Encryption

Certain changes on a ciphertext change the underlying plaintext in a structured way.

▶ **Definition 6** (Homomorphic Encryption). *These four PPT algorithms describe a homomorphic encryption scheme:* KeyGen,Enc, *and* Dec *as in pubilc-key encryption and*

Eval$(1^\lambda, \mathsf{pk}, f, c_1, ..., c_n)$*: The evaluation algorithm takes a security parameter λ, a public key* pk*, a string representation of a function f and n where n is the input size of f ciphertexts $c_1, \ldots, c_n$ as inputs and outputs a new ciphertext c.*

▶ **Definition 7** (Homomorphic Correctness). *Let $\mathcal{F}$ be a set of functions, f be an arbitrary element of $\mathcal{F}$, and $n = in(f)$. An $\mathcal{F}$-homomorphic encryption scheme* ($\mathsf{KeyGen}, \mathsf{Enc}$, Eval, Dec) *is correct if* ($\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$) *is a correct encryption scheme, and for all messages $m_1, \ldots, m_n$, security parameters λ, and* $(\mathsf{pk}, \mathsf{sk})$ *from the support of* $\mathsf{KeyGen}(1^\lambda)$ *we have* $f(m_1, \ldots, m_n) = \mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(1^\lambda, \mathsf{pk}, f, \mathsf{Enc}(\mathsf{pk}, m_1), \ldots, \mathsf{Enc}(\mathsf{pk}, m_n)))$

▶ **Definition 8** (Linearly-Homomorphic Encryption). *A linearly-homomorphic encryption scheme (LHE) is an $\mathcal{F}$-homomorphic encryption scheme where $\mathcal{F}$ is the set of all multivariate linear functions.*

▶ **Definition 9** (Fully-Homomorphic Encryption). *A fully-homomorphic encryption scheme (FHE) is an $\mathcal{F}$-homomorphic encryption scheme where $\mathcal{F}$ is the set of all computable functions.*

CPA security is unchanged from public key encryption.

The ability to use a homomorphic evaluation on a ciphertext which has already gone through evaluation is called multi-hop. To define the correctness of a multi-hop HE we need to define a set $\mathcal{C}_{\mathsf{pk}}$ correctly generated ciphertexts. Each ciphertext comes from encryption or homomorphic evaluation on a correct plaintext.

▶ **Definition 10** (Multi-Hop Homomorphic Encryption). *Just like a $\mathcal{F}-HE$ scheme, a multi-hop $\mathcal{F}-HE$ scheme is a quadruple of PPT algorithms (*KeyGen,Enc,Eval,Dec*). Let λ be a security parameter,* $(\mathsf{pk}, \mathsf{sk})$ *be the output of* $\mathsf{KeyGen}(1^\lambda)$ *then*

$$\mathcal{C}_{\mathsf{pk}} = \left\{ c \,\middle|\, \begin{array}{l} m \in \mathcal{M} \wedge c = \mathsf{Enc}(\mathsf{pk}, m) \vee \\ f \in \mathcal{F} \wedge n = in(f) \wedge c_1, \ldots, c_n \in \mathcal{C}_{\mathsf{pk}} \wedge c = \mathsf{Eval}(1^\lambda, \mathsf{pk}, f, c_1, \ldots, c_n) \end{array} \right\}$$

is a set of correctly generated ciphertexts under public key pk. *Such a quadruple of algorithms is a multi-hop $\mathcal{F}-HE$ scheme if it is a $\mathcal{F}-HE$ and for all security parameters λ, outputs of the* $\mathsf{KeyGen}(1^\lambda)$ $(\mathsf{pk}, \mathsf{sk})$, *functions $f \in \mathcal{F}$, $n = in(f)$, and ciphertexts $c_1, \ldots c_n \in \mathcal{C}_{\mathsf{pk}}$ we have* $f(\mathsf{Dec}(\mathsf{sk}, c_1), \ldots, \mathsf{Dec}(\mathsf{sk}, c_n)) = \mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(1^\lambda, \mathsf{pk}, f, c_1, \ldots, c_n))$

The rate captures how big a ciphertext is in comparison to its plaintext content.

▶ **Definition 11** (Rate). *An $\mathcal{F}-HE$ scheme* ($\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}$) *has rate ρ if there exists a polynomial μ such that for all security parameters λ, possible outputs of* $\mathsf{KeyGen}(1^\lambda)$ $(\mathsf{pk}, \mathsf{sk})$, *correctly generated ciphertexts $c \in \mathcal{C}_{\mathsf{pk}}$ of size $\geq \mu(\lambda)$ we have* $|\mathsf{Dec}(\mathsf{sk}, c)|/|c| \geq \rho(\lambda)$

We call an encryption scheme high rate if it has a rate greater than $1/2$.

Typically a HE is also defined with compactness. For compactness, we require the ciphertext to be independent in size from the functions evaluated to arrive at the ciphertext.

▶ **Definition 12** (Compactness). *An $\mathcal{F}-HE$ scheme* ($\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}$) *is compact if there exists a rate ρ that only depends on λ.*

There is also a notion of malicious circuit privacy that guarantees that the ciphertext does not leak information about the function which was homomorphically evaluated on it beyond the result even if the public key and the ciphertexts are maliciously generated [26].

▶ **Definition 13** ((Malicious) Circuit Privacy). *We say an $\mathcal{F}-HE$ scheme is maliciously, statistically circuit private if there exists an unbounded simulator* Sim *with one-time oracle access to f such that for all λ, and for all public keys* pk, *functions $f \in \mathcal{F}$, and ciphertexts $c = (c_1, \ldots, c_n)$ for $n = in(f)$ we have $SD(\mathsf{Sim}^f(1^\lambda, \mathsf{pk}, c), \mathsf{Eval}(1^\lambda, \mathsf{pk}, f, c))$ is negligible in λ.*

Our constructions do not quite achieve the malicious, statistically circuit privacy guarantee of [26]. However, we achieve a slightly weaker notion defined in the following.

▶ **Definition 14** (Φ-Circuit Privacy). *Let $\Phi : \mathcal{F} \to \{0,1\}^*$ be a (leakage) function. We say an $\mathcal{F}-HE$ scheme is Φ (maliciously) circuit private if there exists an unbounded simulator* Sim *with one-time oracle access to f such that for all λ, public keys* pk, *ciphertexts $c = c_1, \ldots, c_n$, functions $f \in \mathcal{F}$, and PPT adversaries $\mathcal{A}$ we have $|Pr[\mathcal{A}(\mathsf{Sim}^f(1^\lambda, \mathsf{pk}, c, \Phi(f)))] - Pr[\mathcal{A}(\mathsf{Eval}(1^\lambda, \mathsf{pk}, f, c))]|$ is negligible in λ.*

The only difference to the above notion of circuit privacy is that the simulator gets some leaked information Φ about the circuit. In most cases, Φ would leak some structural information such as the size of the circuit or its topology. This notion is adapted to expose some properties of the circuit from privacy definitions for garbled circuits.

A.4 Garbling Schemes

Garbling schemes were famously introduced by Yao in an oral presentation [30] about techniques for secure function evaluation. Our notation is adapted from [7] and also influenced the definition of Φ circuit privacy for HE. It allows to split up the evaluation of a function such that different parties can do parts of the computation. One party knows the input x to the function f and encodes it such that the other party can evaluate the function on the encoding (i.e. learn $f(x)$) without being able to compute the input.

▶ **Definition 15** (Garbling Schemes). *A garbling scheme is described by the following PPT algorithms:*

$\mathsf{Garble}(1^\lambda, f)$: *The circuit garbling algorithm takes a security parameter and the circuit representation of a function f as inputs and outputs a garbled circuit F and $2n$ bitstrings $X_1^0, X_1^1, \ldots, X_n^0, X_n^1$ where n is the input size of f.*

$\mathsf{GarbleInput}((X_1^0, X_1^1, \ldots, X_n^0, X_n^1), m)$: *The input garbling mechanism takes $2n$ bitstrings $X_1^0, X_1^1, \ldots, X_n^0, X_n^1$ and a message x as inputs and outputs the n bitstrings $X_1^{x_1}, \ldots, X_n^{x_n}$.*

$\mathsf{Ev}(F, (X_1, \ldots, X_n))$: *The evaluation algorithm takes a garbled function F and n bitstrings $X_1, \ldots X_n$ as inputs and outputs $f(x)$.*

▶ **Definition 16** (Correctness). *A garbling scheme* $(\mathsf{Garble}, \mathsf{GarbleInput}, \mathsf{Ev})$ *is correct if f is a function, x is an input to that function, λ is the security parameter, (F, e) is from the range of* $\mathsf{Garble}(1^\lambda, f)$ *then* $\mathsf{Ev}(F, \mathsf{GarbleInput}(e, x)) = f(x)$.

▶ **Definition 17** (Statistical Privacy). *A garbling scheme is Φ statistically private if there exists a unbounded algorithm* $\mathsf{Sim}(1^\lambda, y, \varPhi)$ *such that,*

$$SD(\mathsf{Sim}(1^\lambda, y, \Phi(f)))|y = f(x)], \left[\mathcal{D}(F, X) \middle| \begin{array}{l} (F, e) \leftarrow \mathsf{Garble}(1^\lambda, f) \\ X \leftarrow \mathsf{GarbleInput}(e, x) \end{array} \right])$$

is negligible in λ.

Garbled circuits with statistical privacy are usually researched under the guise of Decomposable Affine Randmized Encodings (DARE) [22, 5, 4].

An example for this is [23]'s construction for branching programs.

A.5 Oblivious Transfer

String oblivious transfer (OT) is a protocol which allows two parties (sender and receiver) to interact in the following way: The sender has two strings m_0, m_1 and the receiver has a bit b. The goal is that the receiver learns m_b but the sender does not learn anything about b.

▶ **Definition 18** (Oblivious Transfer). *A (two-message) OT is described by the following PPT algorithms:*

$\mathsf{OT}_1(1^\lambda, b)$: *With the input of a security parameter λ and a bit b, the algorithm returns ot_1 and state.*

$\mathsf{OT}_2(1^\lambda, ot_1, m_0, m_1)$: *With the input of a security parameter λ, request ot_1, and two strings of same length m_0, m_1, the algorithm returns a response ot_2*

$\mathsf{OT}_3(ot_2, state)$: *With the input of a response ot_2 and a state state, the algorithm returns a string m*

▶ **Definition 19** (Correctness). *An OT* $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ *is correct if for all security parameters λ, bits b, messages m_0, m_1, $(ot_1, state)$ from the range of* $\mathsf{OT}_1(1^\lambda, b)$ *and ot_2 from the range of* $\mathsf{OT}_2(1^\lambda, ot_1, m_0, m_1)$ *we have* $m_b = \mathsf{OT}_3(ot_2, state)$

▶ **Definition 20** (Receiver's Security). *An OT* $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ *has (computational) receiver's security if for every PPT adversary $\mathcal{A}$, and security parameters λ we have* $\left|Pr[\mathcal{A}(\mathsf{OT}_1(1^\lambda, 0)] - Pr[\mathcal{A}(\mathsf{OT}_1(1^\lambda, 1)]\right|$ *is negligible in λ.*

▶ **Definition 21** (Statistical Sender's Security). *An OT* $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ *has statistical sender's security if there exists a deterministic unbounded simulator* Sim *such that for all security parameters λ, strings ot_1, strings m_0, m_1 of length k we have* $SD(\mathsf{OT}_2(1^\lambda, ot_1, m_0, m_1), \mathsf{Sim}^{m_{(\cdot)}}(1^\lambda, ot_1, k))$ *is negligible in λ with* Sim *having one time access to a $m_{(\cdot)}$ oracle.*

▶ **Definition 22** (Rate). *An OT* $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ *has rate ρ if there exists a polynomial μ such that for all security parameters λ, possible outputs ot_1 of* $\mathsf{OT}_1(1^\lambda, b)$*, and messages m_0, m_1 with $|m_0| = |m_1| \geq \mu(\lambda)$ we have* $|m_0|/|\mathsf{OT}(1^\lambda, ot_1, m_0, m_1)| \geq \rho(\lambda)$

For the purposes of this document every OT has computational receiver's security, and statistical sender's security.

A.6 Information Theory

The statistical distance is a metric on probability distributions. It is often used in cryptography because it is at the core of the definition of statistical indistinguishability. Statistical indistinguishability is a strictly stronger notion than computational indistinguishability, which is the most popular tool to define security notions in cryptography.

▶ **Definition 23** (Statistical Distance). *Let X and Y be two distributions with support in $\{0,1\}^k$. The statistical difference between X and Y, $SD(X,Y)$ is given by,*

$$SD(X,Y) = \frac{1}{2} \sum_{x \in \{0,1\}^k} |\Pr[X = x] - \Pr[Y = x]|$$

▶ **Lemma 24.** *The statistical distance has an equivalent definition*

$$SD(X,Y) = max_{f:\{0,1\}^k \to \{0,1\}} |\Pr[f(X) = 1] - \Pr[f(Y) = 1]|$$

Entropy measures a lack of knowledge about a system. The most famous entropy is the Shannon entropy H, which measures the lack of knowledge in a system that behaves randomly. Min-entropy, on the other hand, assumes a system which behaves maliciously.

▶ **Definition 25** (Min-Entropy). *Let X be a distribution. The min-entropy of X is*

$$H_\infty(X) = -log(max_x \Pr[X = x])$$

▶ **Definition 26** (Conditional (Smooth) Min-Entropy [14]). *The conditional smooth min-entropy* $H^{\varepsilon}_{\infty}(X|Y)$ *is defined as* $H^{\varepsilon}_{\infty}(X|Y) = max_{\mathcal{E}} min_y H_{\infty}(X\mathcal{E}|Y = y)$*, where the maximum is over all events* $\mathcal{E}$ *with* $Pr(\mathcal{E}) \geq 1 - \varepsilon$*.*

▶ **Corollary 27** (Corollary of Lemma 1 from [14]). *Let* X, Y *be distributions then* $H^{\varepsilon}_{\infty}(X|Y) > H_{\infty}(X, Y) - H_0(Y) - log(1/\varepsilon)$ *for all* ε*.*

Strong extractors make it possible to use one source of uniform randomness to convert a non-uniform distribution with some min-entropy into a uniform distribution.

▶ **Definition 28** (Strong Extractor). *A function* $\mathsf{Ext} : \{0,1\}^m \times \{0,1\}^d \to \{0,1\}^n$ *is a* (k, ϵ)*-strong extractor if for every distribution* X *with support in* $\{0,1\}^m$ *and* $H_{\infty}(X) = k$*, we have* $SD((\mathsf{Ext}(X, U_d), U_d), (U_n, U_d)) \leq \epsilon$ *where* U_d *is a uniform distribution over* $\{0,1\}^d$ *and* U_n *is one over* $\{0,1\}^n$*.*

Many of the useful rules like the chain rule for conditional Shannon entropy $H(X|Y) = H(X, Y) - H(Y)$ do not hold for min-entropy. Therefore we have to do hard work to handle claims about min-entropy.

The next lemma allows to lower bound the min-entropy using the average conditional min-entropy.

▶ **Lemma 29** (Weakened Lemma 2.2 of [15]). *For all random variables* X, Y*,* $\delta > 0$ *the conditional min-entropy we have* $H_{\infty}(X|Y = y) \geq \tilde{H}_{\infty}(X|Y) - log(1/\delta)$ *with probability* $1 - \delta$ *over the choice of* y

The leakage lemma for min-entropy helps with bounding the min-entropy of distributions that are conditioned on events.

▶ **Lemma 30** (Leakage Lemma for Min-Entropy of [28]). *For all random variables* X *and events* A, B *we have* $H_{\infty}(X|B, A) > H_{\infty}(X|B) - log(1/Pr(A|B))$

▶ **Corollary 31** (Corollary 4.3 of [14]). *Let* $\varepsilon \geq 0$*, and let* X_0*,*X_1 *and* Z *be random variables such that* $H^{\varepsilon}_{\infty}(X_0, X_1|Z) \geq \alpha$*. Then, there exists a binary random variable* C *over* $\{0,1\}$ *such that* $H^{\varepsilon+\varepsilon'}_{\infty}(X_{1-C}|Z, C) \geq \alpha/2 - 1 - log(1/\varepsilon')$ *for any* $\varepsilon' > 0$*.*

▶ **Lemma 32** (Smooth Min-Entropy Conversion). *If* $H^{\varepsilon}_{\infty}(X) \geq \alpha$ *then* $H_{\infty}(X) \geq -log(2^{-\alpha} + \varepsilon)$

Proof. Since $H^{\varepsilon}_{\infty}(X) \geq \alpha$ there exists a distribution Y such that $H_{\infty}(Y) \geq \alpha$ and $SD(X, Y)$. This means, for all y', $\Pr_{y \leftarrow Y}[y' = y] \leq 2^{-\alpha}$. Therefore, the biggest probability of X can only be bigger by ε. Then, for all x', $\Pr_{x \leftarrow X}[x' = x] \leq 2^{-\alpha} + \varepsilon$. ◀