Electronic Colloquium on Computational Complexity, Report No. 125 (2018)

Fundamentals of Fully Homomorphic Encryption – A Survey

Zvika Brakerski[*]

Abstract

A homomorphic encryption scheme is one that allows computing on encrypted data without decrypting it first. In fully homomorphic encryption it is possible to apply *any* efficiently computable function to encrypted data. We provide a survey on the origins, definitions, properties, constructions and uses of fully homomorphic encryption.

1 Homomorphic Encryption: Good, Bad or Ugly?

In the seminal RSA cryptosystem [RSA78], the public key consists of a product of two primes $N = p \cdot q$ as well as an integer e, and the message space is the set of elements in $\mathbb{Z}_N^*$. Encrypting a message m involved simply raising it to the power e and taking the result modulo N, i.e. $c = m^e \pmod N$. For the purpose of the current discussion we ignore the decryption process. It is not hard to see that the product of two ciphertexts c_1 and c_2 encrypting messages m_1 and m_2 allows to compute the value $c_1 \cdot c_2 \pmod N = (m_1 m_2)^e \pmod N$, i.e. to compute an encryption of $m_1 \cdot m_2$ without knowledge of the secret private key. Rabin's cryptosystem [Rab79] exhibited similar behavior, where a product of ciphertexts corresponded to an encryption of their respective plaintexts. This behavior can be expressed in formal terms by saying that the ciphertext space and the plaintext space are *homomorphic (multiplicative) groups*. The decryption process defines the homomorphism by mapping a ciphertext to its image plaintext.

Rivest, Adleman and Dertouzos [RAD78] realized the potential advantage of this property. In a time where complex computations required "buying computing cycles" from a mainframe computer maintained by an external company, one would be exposed to the danger of their private information being revealed to the vendor of computing power. However, if the computation only involves group operations on the input data, then homomorphism will allow the vendor to perform the computation on the ciphertext, rather than the plaintext, so that sensitive data is not revealed on one hand, and the heavy computational load is outsourced to the vendor on the other. Remarkably, 40 years down the line, outsourcing computation gained popularity once again with the introduction of cloud computing. Indeed, privacy in the era of the cloud is one of the most fascinating topics in modern cryptographic research. Naturally, one would like to extend homomorphism beyond group operations, and indeed [RAD78] put forth the question whether there exist encryption schemes that are homomorphic also with respect to *ring* (or field) operations, which would allow to perform arbitrary computation on the input data.

[*]Weizmann Institute of Science, `zvika.brakerski@weizmann.ac.il`. Supported by the Israel Science Foundation (Grant No. 468/14), Binational Science Foundation (Grants No. 2016726, 2014276), and by the European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482) and via Project PROMETHEUS (Grant 780701).

ISSN 1433-8092

Alas, plain RSA and Rabin's scheme provide a very weak level of security (and indeed today they are referred to as "trapdoor functions" and not as encryption schemes, see e.g. [Gol01, Section 2.4.4.2]). The revolutionary work of Goldwasser and Micali [GM82] on randomized encryption defined a new notion, *semantic security*, as a standard for encryption security. Since previous schemes, such as the aforementioned plain RSA and Rabin schemes, were not semantically secure, it was up to Goldwasser and Micali to present a different candidate. Indeed, they presented one based on the hardness of the quadratic residuosity problem (QR). The Goldwasser-Micali encryption scheme was again based on $N = pq$ as public key, but now each element in $\mathbb{Z}_N^*$ was only used to encrypt a single bit. Squares (a.k.a quadratic residues) encrypt 0, and quadratic non-residues (non-squares with Jacobi symbol 1) encrypt 1. Note that in such a scheme, as in *any* semantically secure encryption scheme, each message is associated with a super-polynomial number of possible ciphertexts, all decrypting to the same value. Despite this significant conceptual difference, the Goldwasser-Micali encryption scheme still exhibits group homomorphism, since a product of ciphertexts will decrypt to the XOR of the plaintexts. The ElGamal scheme [Gam84] that followed soon after exhibited similar behavior, even though it was based on the hardness of a different type of problem (related to the discrete logarithm problem). A decade down the line, as *lattice-based* encryption emerged [AD97, GGH97, HPS98], they also exhibited homomorphic properties, despite being based on a very different mathematical structure.

It turned out that homomorphic encryption (at least for groups) is abundant and one could have speculated that it is even unavoidable. As [RAD78] showed, this can have positive implications, since it could lead to private outsourcing of computation. On the other hand, one could speculate that this property only indicates that public key encryption schemes have too much structure. Perhaps this is a symptom of insecurity?

Consider the following scenario, Alice and Bob are bidding for some goods in an auction. Each one submits their bid in a sealed envelope, implemented using an encryption scheme. Bob is willing to pay y and he knows that Alice's bid x is much lower than y, but he does not know what it is. Bob can see Alice's sealed envelope, in the form of a ciphertext $\mathsf{Enc}(x)$. If the encryption scheme is homomorphic, then Bob can generate an encryption $\mathsf{Enc}(x + 1)$ thus creating the smallest bid to win the auction, even without learning anything about Alice's input. This demonstrates that in some situation we would like a guarantee that it is impossible to perform any alteration of the ciphertext, in particular homomorphism. This property is called *non-malleability* [DDN91]. One conclusion from this example is that one should not think of homomorphism as intrinsically useful or intrinsically harmful, but rather consider the specific situation.

In this context, we mention that the aforementioned notion of semantic security is equivalent (in the public key setting) to security under *chosen plaintext attacks* (CPA) where an attacker gets access to the encryption function but no access at all to the decryption function. In many situations one would consider stricter notions where (limited) access to the decryption function is allowed, e.g. to model settings where an adversary can send "made up" ciphertexts to the decryptor and observe the decryptor's behavior upon receiving the message. This is formalized via the notion of security under chosen ciphertext attacks (CCA), and comes in two main flavors. CCA1 is a notion that models a setting where an adversary has access to the decryption function (as oracle) before the it gets hold of the challenge ciphertext it wants to attack. CCA2 allows the adversary to access the decryption oracle even after seeing the target ciphertext (with a non degeneracy condition that the adversary cannot use this access to decrypt the challenge itself). It is not hard to see that homomorphism (or malleability) contradict CCA2 security. However, homomorphic encryption schemes can be CCA1 secure [CS98].

As explained above, group homomorphic encryption schemes emerged naturally from attempts for constructing public key encryption scheme. However, ring homomorphism seems much harder to construct. Indeed, over 35 years passed until the vision of [RAD78] was materialized by Gentry [Gen09b] in one of the most inspiring works in cryptography in recent years.

2 Definition and Basic Properties

Motivated by the application of outsourcing computation, we might not want to restrict ourselves to algebraic terminology. Instead, we can define $\mathcal{F}$-homomorphism with respect to the class of operations $\mathcal{F}$ that can be applied to encrypted data. Notation-wise, a public key encryption scheme consists of a (randomized) key generation process, that produces a secret key sk and a public key pk, a (randomized) encryption function Enc and a (deterministic w.l.o.g) decryption function Dec. Throughout this manuscript we will consider a plaintext space of binary strings $\{0,1\}^*$, and an encryption procedure that encrypts the message bits one at a time. Syntactically, encrypting a message x using a public key pk is denoted $\mathsf{Enc}_{\mathsf{pk}}(x)$. Decrypting a ciphertext c is denoted $\mathsf{Dec}_{\mathsf{sk}}(c)$. We can now define $\mathcal{F}$-homomorphism.

Definition 2.1 *Let $\mathcal{F}$ be a set of functions in $\{0,1\}^* \to \{0,1\}$. A public key scheme is $\mathcal{F}$-homomorphic if there exists an* evaluation algorithm Eval *s.t.* $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}(f, \mathsf{Enc}_{\mathsf{pk}}(x))) = f(x)$ *for all $f \in \mathcal{F}$ and $x \in \{0,1\}^*$ of appropriate length.*

A fully *homomorphic encryption (FHE) is a homomorphic encryption scheme where $\mathcal{F}$ is the set of all functions (or at least the set of all efficiently computable functions).*

That is, encrypting a value x, followed by applying homomorphic evaluation with f, and decrypting the output, should result in the value $f(x)$. This is the minimal requirement for the purpose of private outsourcing. There are a few points that are worth noting about this definition.

- **The syntax of the homomorphic evaluation procedure.** It is simplest to define the homomorphic evaluation procedure as only taking the respective ciphertexts as input. While this is true without loss of generality (as we explain momentarily), in many cases the evaluation procedure also uses the public key of the encryption scheme. Syntactically this can be avoided by redefining the ciphertexts as containing the public key, and thus allowing evaluation using only the ciphertexts, w.l.o.g. Still, often for reasons of efficiency and syntactic elegance the Eval procedure takes the public key as an additional parameter.

  Furthermore, in many candidates, it is easy to identify a part of the public key that is used for homomorphic evaluation, and a separate part that is used for encryption. It is sometimes convenient to refer to the former as the "evaluation key" of the scheme, thus characterizing a homomorphic encryption scheme as having a secret key sk and *two* public keys $\mathsf{pk}, \mathsf{evk}$, one used for encryption and one for homomorphic evaluation. This is particularly convenient in cases where it is possible to amplify the homomorphic capabilities of the scheme by modifying evk while keeping sk, pk unchanged (e.g. via *bootstrapping*, see Section 3).

- **Representation of functions.** The evaluation procedure takes a function $f \in \mathcal{F}$ as input. This means that it is not enough to think about $\mathcal{F}$ as a class of functions, but rather we must consider the representation of these functions. In particular, since Eval needs to be polynomial time computable, the representation of f effects the permitted running time of $\mathsf{Eval}(f, \cdot)$. It is most common to consider the *boolean circuit model* to represent f.

- **Homomorphic evaluation needs not preserve form.** We only required above that the evaluated ciphertext (i.e. the ciphertext output by Eval) is decryptable to the correct value. There is no requirement that $c_f = \mathsf{Eval}(f, \mathsf{Enc}_{\mathsf{pk}}(x))$ looks similar to a fresh ciphertext $\mathsf{Enc}_{\mathsf{pk}}(f(x))$. This choice is made in order to capture the minimal meaningful definition for private outsourcing of computation. However, this minimal definition opens the door to a degenerate FHE construction as follows. Consider any secure public-key encryption scheme, and append it with the function $\mathsf{Eval}(f, c)$ that simply outputs the tuple (f, c). Furthermore extend the the decryption algorithm to decrypt pairs (f, c) by first decrypting the c component and then applying the f component on the output. This scheme is homomorphic with respect to the above definition, but fails to capture a notion of non-trivial outsourcing.

  To avoid this degeneracy, we present two properties that are natural requirements in the context of outsourcing. Neither one of these is captured by the aforementioned degenerate example.

  - **Compactness.** If our intent in homomorphic encryption is to delegate the computational complexity of the computing f to a remote server, then it is natural to require that the decryption complexity does not depend on the complexity of the function being evaluated. Formally, adopting the convention that the decryption procedure runs in fixed polynomial time in its input length, it is sufficient to require that the bit-length of the evaluated ciphertext c_f does not depend on the complexity of f (beyond the obvious dependence on the output length).
  - **Function Privacy.** In certain situations, it may be important that c_f does not reveal any information about f itself (e.g. when the evaluator uses a proprietary algorithm). Function privacy should hold even with respect to an adversary that has the secret key, i.e. the requirement is that even the decryptor cannot learn anything about f from c_f, except for the value $f(x)$. One could consider even stronger notions of function privacy, for example one that considers public-keys and ciphertexts that are maliciously generated in attempt to extract more information about f than permitted [OPP14].

  Compactness and function privacy are both sought after properties in certain situations, and in others it could make sense to require only one but not necessarily both. More often than not, the term FHE refers to compact FHE, and it is explicitly mentioned where a non-compact scheme is sufficient (e.g. if only function privacy is needed).

  It can be shown that a compact FHE scheme implies a (different) FHE scheme which is both compact and function private via a non-trivial transformation (this is implicit in [GHV10]).

- **No additional security requirements.** Our definition of homomorphism above did not make any requirements about security, except that the underlying scheme (without homomorphic evaluation) is secure. Standard notions of security (e.g. semantic security) are only concerned with information leaked by freshly encrypted ciphertexts, and not about ones that are a result of some manipulation such as homomorphic evaluation. Therefore, one might be worried that post-evaluation ciphertexts might be more vulnerable. However, since the evaluation procedure only uses public information, semantic security guarantees that homomorphic evaluation cannot assist in breaching security of the original ciphertexts. This, in turn, also implies that post-evaluation ciphertexts are protected, at least to the extent that it should not be possible to reveal information about the output of the evaluation process that can assist in learning something about the inputs.

We note that while we are guaranteed that c_f cannot reveal any information about x, it is allowed to reveal information about $f(x)$, to the extent that the information revealed is independent of x. For example, if f is the all zero function, then c_f might expose that $f(x) = 0$ (unless we impose stronger guarantees such as function privacy).

- **Single-hop vs. multi-hop homomorphism.** In the aforementioned definition it is only required that the post-evaluation ciphertext decrypts properly. As we explained above, this does not necessitate that the output ciphertext is structurally similar to a freshly encrypted ciphertext. In particular, it might be the case that it is not possible to re-apply the homomorphic evaluation function to post-evaluated ciphertexts. Schemes that adhere to the basic definition are sometimes referred to as *single-hop homomorphic* as opposed to *multi-hop homomorphism* which allows multiple successive applications of homomorphic evaluation. These notions have been studied in [GHV10]. Gentry's bootstrapping theorem [Gen09b, Gen09a] allows to convert any compact single-hop fully homomorphic encryption into a multi-hop scheme (see more details in Section 3).

- **Leveled fully homomorphic encryption.** As explained above, a fully homomorphic encryption scheme is one that can evaluate any input circuit. Unfortunately, in some cases, this goal is not directly achievable, or requires security and functionality overhead. In those cases it is sometimes useful to define the notion of *leveled* FHE, which refers to a *family* of FHE schemes that allow, for any depth bound d, to generate an instance of the FHE scheme that supports the evaluation of depth-d circuits. The parameters of the scheme are allowed to grow polynomially with d, and some definitions are even stricter and require that evk is the only parameter that depends on d and that this dependence is linear. Leveled FHE schemes are by themselves sufficient for some applications, and in most cases can be upgraded to (non-leveled) FHE using Gentry's bootstrapping theorem [Gen09b, Gen09a], albeit with efficiency loss and an additional security assumption.[1]

3 Bootstrapping and Circular Security

We will now describe one of the most fundamental and useful tools in the construction of fully homomorphic encryption, *the bootstrapping theorem*, introduced in Gentry's seminal work [Gen09b, Gen09a]. The bootstrapping theorem is, to date, a necessary component in all FHE candidates. Using the bootstrapping theorem in its strongest form requires introducing an additional hardness assumption concerning the *circular security* of encryption schemes (we will explain this in detail below). It is currently unknown how to relate this additional assumption to standard cryptographic

[1]In early works on FHE, the term "somewhat homomorphic encryption" (SHE) was used to indicate a scheme with homomorphic capabilities against a restricted class of functions (depth bounded). The two terms are sometimes used interchangeably, however in the original SHE scheme [Gen09b, Gen09a] the parameters of the scheme grew *exponentially* with d.

assumptions, thus the use of bootstrapping subjects all known FHE candidates to the additional circularity requirement.

Key-Switching. We start by introducing the key-switching technique which is useful for bootstrapping but can also be used in other settings. Perhaps the simplest motivation for key-switching is to show that given a (possibly non-homomorphic) scheme with very efficient encryption, and a different homomorphic scheme (possibly with very inefficient, but still polynomial time, encryption), it is possible to create a scheme that inherits the encryption complexity of the former and homomorphic abilities of the latter.

We denote the keys of the non-homomorphic scheme by $(\mathsf{nhsk}, \mathsf{nhpk})$, and its encryption and decryption functions by $\mathsf{NHEnc}, \mathsf{NHDec}$. Let $(\mathsf{hsk}, \mathsf{hpk})$ denote the secret key and public key of the homomorphic scheme (with encryption and decryption functions $\mathsf{Enc}, \mathsf{Dec}$). Consider a ciphertext c that encrypts a plaintext x under the non-homomorphic scheme, i.e. such that $\mathsf{NHDec}_{\mathsf{nhsk}}(c) = x$. Our goal is to apply homomorphic evaluation of some function f, namely to generate a ciphertext that encrypts the value $f(x) = f(\mathsf{NHDec}_{\mathsf{nhsk}}(c))$. Note that f and c are publicly known and the only unknown in the expression $f(\mathsf{NHDec}_{\mathsf{nhsk}}(c))$ is nhsk. We can thus define an efficiently computable function $\tilde{f}_c(\alpha) = f(\mathsf{NHDec}_{\alpha}(c))$ (we omit the subscript c and write $\tilde{f}$ when it is clear from the context). Thinking of the value $f(x)$ as a function of $\alpha = \mathsf{nhsk}$ instead of as a function of x itself, we can think about *homomorphic evaluation* of the function $\tilde{f}_c$. This means that we no longer care that c is encrypted under a non-homomorphic scheme, all we care about now is that α, the input to $\tilde{f}_c$, is encrypted under the homomorphic key hpk. That is, if we had a ciphertext $c^* = \mathsf{Enc}_{\mathsf{hpk}}(\alpha)$, i.e. a homomorphic encryption of a value α, then we can compute $c_{\tilde{f}} = \mathsf{Eval}(\tilde{f}_c, c^*)$ (note that the syntax here is correct since we are applying Eval on a ciphertext encrypted under the homomorphic key hpk). What can we say about $c_{\tilde{f}}$? As the output of a homomorphic evaluation of a function $\tilde{f}_c$ on a properly encrypted ciphertext c^*, we can say that $c_{\tilde{f}}$ should decrypt under hsk to the value $\tilde{f}_c(\alpha) = f(\mathsf{NHDec}_{\alpha}(c))$. Since c is encrypted (under the non-homomorphic scheme), this value will be meaningless for almost all values of α, but it will be meaningful for $\alpha = \mathsf{nhsk}$, for which $\tilde{f}_c(\mathsf{nhsk}) = f(x)$.

The conclusion is that if we can provide the auxiliary information $c^* = \mathsf{Enc}_{\mathsf{hpk}}(\mathsf{nhsk})$, i.e. an encryption of the non-homomorphic secret key under the homomorphic public key, then it would be possible, given f and c, to compute an encryption of the value $f(x)$, thus performing homomorphic evaluation over a ciphertext encrypted using the non-homomorphic scheme. Specifically, to generate a value $c_{\tilde{f}}$ s.t.

$$\mathsf{Dec}_{\mathsf{hsk}}(c_{\tilde{f}}) = \mathsf{Dec}_{\mathsf{hsk}}(\mathsf{Eval}(\tilde{f}_c, c^*)) = \tilde{f}_c(\mathsf{Dec}_{\mathsf{hsk}}(c^*)) = \tilde{f}_c(\mathsf{nhsk}) = f(\mathsf{NHDec}_{\mathsf{nhsk}}(c)) = f(x) \ .$$

The value c^* should be posted publicly alongside the public keys hpk and nhpk of the homomorphic and non-homomorphic scheme.

It is important to notice that the output ciphertext $c_{\tilde{f}}$ indeed constitutes an encryption of $f(x)$, but under the *homomorphic* key hsk. In fact, what we showed was a *key switching* technique that allows to take a ciphertext encrypted under a certain encryption scheme, and convert it into a ciphertext encrypted under a different scheme (using the homomorphic properties of the latter). This explains why the secret key nhsk is required for the generation of $c^* = \mathsf{Enc}_{\mathsf{hpk}}(\mathsf{nhsk})$, since otherwise the ability to decrypt $c_{\tilde{f}}$ using hsk would contradict the semantic security of the non-homomorphic scheme.

We will see next how to extend key-switching into bootstrapping, but let us mention that the switching technique by itself is quite useful. For example, the encryption complexity of an FHE schemes might be quite high, or the ciphertexts are long (which is indeed the case in many of the current candidates). With key switching, it is possible to use a quick and cheap encryption procedure (in fact, even symmetric key encryption will do), and defer all FHE related operations to the evaluation phase.

From Key-Switching to Bootstrapping. Let us assume that the homomorphic encryption scheme from above was only single-hop homomorphic. This still allows us to define c^* and compute $c_{\tilde{f}}$. However, this would still leave us stuck at single-hop homomorphism, since $c_{\tilde{f}}$ cannot undergo additional homomorphic evaluation. However, equipped with our knowledge of key switching, we do not give up so easily. We showed that using the appropriate auxiliary input, we can perform homomorphic evaluation even on ciphertexts that on the face of it cannot be evaluated. We know that $\mathsf{Dec}_{\mathsf{hsk}}(c_{\tilde{f}}) = f(x)$, and let us assume we want to apply a function g on top of this value. Then again we can define $\tilde{g}(\alpha) = g(\mathsf{Dec}_{\alpha}(c_{\tilde{f}}))$ and define an appropriate c^{**} such that $c_{\tilde{g}} = \mathsf{Eval}(\tilde{g}, c^{**})$ decrypts to the right value $g(f(x))$.

What should the new auxiliary information c^{**} be? It needs to be an encryption of the homomorphic secret key hsk, otherwise the evaluation procedure produces a meaningless value. So what we want is $c^{**} = \mathsf{Enc}_{\mathsf{hpk}}(\mathsf{hsk})$, namely an encryption of the homomorphic secret key under its own public key.[2] Given this value, we can compute $c_{\tilde{g}} = \mathsf{Eval}(\tilde{g}, c^{**})$ as desired, and obtain $c_{\tilde{g}}$ such that

$$\mathsf{Dec}_{\mathsf{hsk}}(c_{\tilde{g}}) = \tilde{g}(\mathsf{hsk}) = g(\mathsf{Dec}_{\mathsf{hsk}}(c_{\tilde{f}})) = g(f(\mathsf{Dec}_{\mathsf{nhsk}}(c))) = g(f(x)) \ .$$

We see that indeed $c_{\tilde{g}}$ decrypts to the desired value, so given c^{**} we can increase the evaluation capacity of our scheme.

The critical observation is that c^{**} is in fact much more useful than our previous c^*. While the latter allows to switch a ciphertext from the non-homomorphic scheme to the homomorphic scheme, and was completely useless afterwards, the former allows us to take homomorphic ciphertexts and produce homomorphic ciphertexts. This in particular means that the same c^{**} can be used more than once. Assume that we want to homomorphically evaluate an additional function h on top of $c_{\tilde{g}}$, we observe that this can be done with the same c^{**}, i.e. without requiring a new auxiliary information. Specifically, just define $\tilde{h}(\alpha) = h(\mathsf{Dec}_{\alpha}(c_{\tilde{g}}))$, and set $c_{\tilde{h}} = \mathsf{Eval}(\tilde{h}, c^{**})$. One can verify that $c_{\tilde{h}}$ indeed decrypts to $h(g(f(x)))$. Note that in order for this to apply, we only require that our encryption scheme is single-hop homomorphic. This is since the Eval function is only executed on the input ciphertext c^{**} which is a freshly encrypted ciphertext and not the result of a previous homomorphic operation. In a sense, we "tricked" the single-hop scheme to perform multi-hop operations by embedding the "real" input inside the function description. At this point we can forget about the initial non-homomorphic scheme (although, as we explained, this application is also sometimes useful) and just consider the task of amplifying single-hop to multi-hop homomorphism. We see that this is possible given the auxiliary information c^{**}, which should be placed as a part of the public key of the new multi-hop scheme (or more accurately as a part of the evaluation key).

To extract even more out of this technique, we notice that in a multi-hop homomorphic scheme, it is sufficient to only be able to evaluate the NAND gate (or any other universal family of boolean gates). This is since each boolean circuit can be written a sequence of such gates, and homomorphic

[2]A knowledgeable reader may have noticed a *circularity* issue, we will discuss this aspect shortly.

evaluation of the circuit can proceed by evaluating the gates one at a time (in topological order) on the output of their predecessors. Plugging this observation into our construction of a multi-hop scheme, we see that in order to allow the amplification from single-hop to multi-hop, all that is required is that the single hop scheme supports the homomorphic evaluation of functions of the form $\tilde{f}(\alpha) = \tilde{f}_{c_1,c_2}(\alpha) = \text{NAND}\left(\mathsf{Dec}_\alpha(c_1), \mathsf{Dec}_\alpha(c_2)\right)$, where c_1 and c_2 are bit strings interpreted as ciphertexts for the single-hop scheme. Thus, if we can devise a homomorphic encryption scheme (even single-hop) that supports this family of functions (NAND-augmented decryption functions), then this scheme can be amplified into full-fledged (even multi-hop) FHE for all functions, at the cost of adding c^{**} to the evaluation key (evk) of the scheme (recall that evk is the part of the public key that is used for homomorphic evaluation).

Gentry's Bootstrapping Theorem states exactly this fact: that once we are able to achieve a certain level of homomorphism, then FHE readily follows. However, our discussion so far neglected an important aspect of the above transformation: Whether the addition of c^{**} to the public evaluation key evk of our resulting scheme (and thus revealing it to a potential attacker) preserves the security of the original scheme. At first glance, this seems to be a non-issue, by definition c^{**} is a properly encrypted ciphertext, so the security of the single-hop scheme should guarantee that revealing it to an attacker should do no harm. However, it turns out that standard notions of encryption security are only concerned with hiding messages that can be generated by an adversary (that has the public key). Encrypting a scheme's secret key using its own public key does not fall under this definition. Indeed, almost all proofs showing that encryption schemes are secure under certain assumptions (e.g., factoring) do not extend to showing security for encrypting the secret key, with the exception of schemes designed especially to have this property such as [BHHO08, ACPS09, BG10]. Therefore, the bootstrapping theorem requires that the homomorphic scheme to be amplified is *circular secure*, namely that it is secure even against adversaries that see an encryption of the scheme's secret key under its public key. To be precise, circular security, or more generally the notion of security against key dependent messages (KDM-security) [BRS02] is a stronger notion where the adversary can adaptively ask for encryptions of messages with some dependence of the secret key. Thus the notion required from bootstrapping is named "weak" circular security.

Theorem 3.1 (Gentry's Bootstrapping Theorem) *If there exists an encryption scheme that is single-hop homomorphic with respect to NAND-augmented decryption circuits, and is weakly circular secure, then there exists a multi-hop FHE scheme.*

A scheme that is single-hop homomorphic with respect to NAND-augmented decryption circuits is called *bootstrappable.*

In particular, the bootstrapping theorem states that if we have a scheme that supports depth bounded homomorphism, and its depth bound is strictly larger than its decryption complexity, then this scheme can be amplified to an FHE (assuming that it is also weakly circular secure). Schemes with such homomorphic capacity can be constructed from standard cryptographic assumptions, such as the learning with errors (LWE) assumption (see Section 4). However, it is not known how to prove weak circular security under a standard assumption for any bootstrappable scheme. Furthermore, bootstrapping underlies all known (non-leveled) FHE constructions, so the current state of affairs is that while leveled FHE can be constructed from standard assumptions, non-leveled FHE requires an explicit weak circular security assumption. *This is the only remaining theoretical barrier towards constructing FHE from standard assumptions.*

The Necessity and Plausibility of the Circular Security Requirement. As explained above, it is not known how to prove circular security based on standard assumptions. However, in the proposed constructions, it is not known how to improve the best known attacks using an encryption of the secret key. Thus, as a heuristic, it appears plausible to assume the circular security holds for known FHE candidates. Having said that, recent works [GKW17] show that weak circular security does not necessarily hold for *every* encryption scheme that is secure under standard assumptions. This is done by introducing contrived schemes where the secret key is design so that its encryption provides additional power to the adversary.

Gentry [Gen09a] proposed a heuristic argument showing that *any* homomorphic encryption scheme supporting high enough evaluation depth should be circular secure. Assume there exists a hash function H s.t. providing the adversary with $(\tilde{c} = \mathsf{Enc}_{\mathsf{hpk}}(\rho), \sigma = H(\rho) \oplus \mathsf{hsk})$ for a random ρ does not make the scheme insecure. This assumption indeed holds in the random oracle model as shown in [BRS02]. If we had such a function H in the standard model, then it would have been possible to compute $\mathsf{Eval}(\tilde{H}_\sigma, \tilde{c})$ where $\tilde{H}_\sigma(\alpha) = \sigma \oplus H(\alpha)$. Note that the output of this homomorphic evaluation procedure is an encryption of hsk as needed. While we do know that no explicit hash function can perfectly implement the random oracle heuristic in all applications, in some applications it is possible. Gentry's argument suggests that refuting the circular security of FHE might require showing that for this application it is impossible to replace random oracle with *any* hash function.

One seemingly simple way to get around the circular security problem can be devised by considering our original example of converting a non-homomorphic scheme into a homomorphic one. In that example, there was no circularity problem since hpk is used to encrypt nhsk, i.e. we encrypted a secret key of one scheme under the public key of another scheme. This allows the security proof to go through, since we can argue that even if an adversary knows nhsk, it should still not be able to breach the security of hpk, and thus it cannot distinguish whether c^* contains an encryption of nhsk or an encryption of an unrelated message. We could therefore hope that the following trick could work for bootstrapping homomorphic encryption schemes: Rather than having a single $c^{**} = \mathsf{Enc}_{\mathsf{hpk}}(\mathsf{hsk})$, we will generate two homomorphic key pairs $(\mathsf{hsk}_1, \mathsf{hpk}_1)$, $(\mathsf{hsk}_2, \mathsf{hpk}_2)$, and generate two auxiliary ciphertexts $c_1^{**} = \mathsf{Enc}_{\mathsf{hpk}_1}(\mathsf{hsk}_2)$ and $c_2^{**} = \mathsf{Enc}_{\mathsf{hpk}_2}(\mathsf{hsk}_1)$. Then, during homomorphic evaluation we will alternate between using c_1^{**} and c_2^{**} for each hop of the computation. This indeed provides the intended functionality, however in terms of security we can see that the prior proof outline no longer works. Even if we only reveal hsk_2 to the adversary, it is straightforward to extract hsk_1 by decrypting c_2^{**}, so we cannot rely on the hardness of hpk_1. Indeed, such a 2-cycle is a type of circular security and the same problems arise.

Before giving up completely on the key cycle concept, we notice that the problem only arises because the chain of keys we generate is a closed loop, so that any of the secret keys can be used to recover all other secret keys. We can consider generating d key pairs $(\mathsf{hsk}_i, \mathsf{hpk}_i)$, and auxiliary information $c_i^{**} = \mathsf{Enc}_{\mathsf{hpk}_{i+1}}(\mathsf{hsk}_i)$ for $i = 1, \ldots, d-1$ (note that we *do not* close the loop since we do not provide $\mathsf{Enc}_{\mathsf{hpk}_1}(\mathsf{hsk}_d)$, and in fact we do not provide any information at all on hsk_d beyond its respective public key). This chain allows us to perform $d-1$ homomorphic hops, and the resulting scheme can be proven secure based only on the security of the original scheme. Instantiating our hops with NAND-augmented decryption circuits, we can get a leveled FHE for any polynomial depth bound d, where the only parameter of the scheme that depends on d is the scheme's evaluation key evk (which now contains evk_i for all i, as well as all auxiliaries c_i^{**}), and this evaluation key only grows linearly with d. There is no need for circular security to prove security for this leveled scheme.

Theorem 3.2 (Gentry's Bootstrapping Theorem for Leveled FHE) *If there exists an encryption scheme which is single-hop homomorphic with respect to NAND-augmented decryption circuits, then there exists a leveled FHE scheme.*

4 Constructing FHE

We will now explain how to construct homomorphic encryption schemes from the learning with errors (LWE) assumption. The scheme we construct will be bootstrappable so it is possible to apply Gentry's bootstrapping theorem to achieve full FHE.

4.1 Learning with Errors – A Primer

The learning with errors (LWE) problem was introduced by Regev [Reg05] and has had a profound effect on cryptographic literature, often allowing to realize cryptographic primitives that are not known under any other assumption. LWE considers a set of many random linear equations over a set of n variables that will be assigned random values, modulo a global modulus $q \gg n$ (where the meaningful range of parameters ranges from q being polynomial to subexponential in n). The vector of variables is denoted by $\mathbf{t} \in \mathbb{Z}_q^n$ (we set it as a row vector), and the (random) coefficients of the linear equations are represented by a uniform matrix $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$, where $m = \mathrm{poly}(n)$ is the number of equations.[3] A set of linear equations is solvable even modulo q, so given $(\mathbf{B}, \mathbf{tB})$ it is possible to efficiently find a solution $\mathbf{t}$ to the set of equations. The LWE problem considers slightly perturbed equations, by adding a small noise to each one. Specifically, let χ be a distribution supported only over integers smaller than some bound B.[4] Consider sampling a noise vector $\mathbf{e}$ from χ^m (a noise term for each equation), and setting $\mathbf{b} = \mathbf{tB} + \mathbf{e} \pmod q$. The (decisional) LWE assumption with parameters (n, q, χ) states that for a uniformly sampled $\mathbf{t}$, the pair $(\mathbf{B}, \mathbf{b})$ is indistinguishable from uniform, even when m is allowed to be an arbitrarily large polynomial. Note that information theoretically this distribution is very far from uniform. The distribution χ is often taken to be a discrete Gaussian, but this is immaterial for the purpose of this survey. For our discussion we can consider setting $q = n^{10}$ and a distribution χ with a bound $B = n$. To further simplify our notation, we will not explicitly write the noise vector $\mathbf{e}$ and instead write $\mathbf{b} \approx \mathbf{tB}$.

We now present a tool that proved extremely useful in LWE-based cryptography. Let $x \in \{0, \dots, q-1\}$, then x can be represented as a sequence of $\lceil \log q \rceil$ bits as $x = \sum 2^i \cdot x_i$, which can also be written as an inner product $(1, 2, 2^2, \dots) \cdot (x_0, x_1, \dots) = \mathbf{g} \cdot \mathbf{x}$. More generally, considering a vector $\mathbf{v} \in \mathbb{Z}_q^n$, one can consider the vector $\mathbf{v}'$ containing a concatenation of the binary representations of all elements of $\mathbf{v}$. The vector $\mathbf{g}$ can thus be generalized to a matrix $\mathbf{G} \in \mathbb{Z}_q^{n\times n\lceil \log q\rceil}$ s.t. $\mathbf{v} = \mathbf{G}\mathbf{v}'$ (the matrix $\mathbf{G}$ is a block diagonal matrix with each block equaling to $\mathbf{g}$). We note that this matrix found additional uses in contexts beyond what is covered in this survey (see, e.g., [MP12]). It is customary to denote the binary representation of $\mathbf{v}$ by $\mathbf{G}^{-1}(\mathbf{v})$, so that it will hold that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{v}) = \mathbf{v}$. We note that $\mathbf{G}^{-1}$ is not a matrix, but rather a function. This notation can be even further extended to apply to matrices so that $\mathbf{G}^{-1}(\mathbf{V})$ for a matrix $\mathbf{V} \in \mathbb{Z}_q^{n\times m}$ is a matrix in $\{0,1\}^{n\lceil \log q\rceil\times m}$, whose every column is the binary decomposition of the respective column of $\mathbf{V}$, so again $\mathbf{G}\mathbf{G}^{-1}(\mathbf{V}) = \mathbf{V}$.

[3]We note that the standard notation for the LWE problem is using $\mathbf{s}, \mathbf{A}$ instead of $\mathbf{t}, \mathbf{B}$. However, this notation will be more convenient for us as we will use $\mathbf{s}, \mathbf{A}$ to denote different quantities in Section 4.2 below.

[4]It is sufficient that the distribution is bounded with overwhelming probability.

4.2 A Homomorphic Encryption Scheme Based on LWE

LWE-based homomorphic encryption was constructed in [BV11]. We present a later construction due to [GSW13] (using notation from the even later [AP14]). The public key is a matrix $\mathbf{A} = \left[\begin{smallmatrix}\mathbf{B}\\ \mathbf{b}\end{smallmatrix}\right]$, where $\mathbf{B}$ is uniform and $\mathbf{b} \approx \mathbf{tB}$ for a random $\mathbf{t}$. The secret key is a vector $\mathbf{s} = (-\mathbf{t}, 1)$. Note that it holds that $\mathbf{sA} \approx 0$, but that $\mathbf{A}$ is indistinguishable from uniform assuming LWE. This public key is identical to that of Regev's original LWE-based public key encryption scheme. However, the ciphertext itself is quite different. The encryption of a message is done in a bit-by-bit manner, where the encryption of each message bit is a *large matrix*.[5] The encryption of a bit $x \in \{0, 1\}$ is the matrix $\mathbf{C} = \mathbf{AR} + x\mathbf{G}$, where $\mathbf{R}$ is a random *binary* matrix (whose dimensions are chosen based on those of $\mathbf{A}$ and $\mathbf{G}$ to ensure syntactic compatibility).

Since $\mathbf{A}$ is indistinguishable from a uniform matrix, the leftover hash lemma guarantees (for properly chosen parameters) that $\mathbf{C}$ is indistinguishable from a completely random matrix, and in particular hides the value of the message x. On the other hand, it holds that

$$\mathbf{sA} = \underbrace{\mathbf{sA}}_{\approx \mathbf{0}}\mathbf{R} + x\mathbf{sG} \approx x\mathbf{sG} \ ,$$

and one can verify that x can indeed be recovered out of this value (knowledge of $\mathbf{s}$ is naturally required). It is important to note that it was important to sample $\mathbf{R}$ from a distribution over small values in order to argue that if $\mathbf{sA} \approx \mathbf{0}$ then $\mathbf{sAR} \approx \mathbf{0}$. Multiplying by $\mathbf{R}$ will most likely somewhat increase the amplitude of the output vector, and in the formal analysis we must keep guard that the amplitude of the resulting vector indeed remains small (i.e. $\ll q$).

To show that the scheme is homomorphic, we will show that starting with two ciphertexts $\mathbf{C}_1, \mathbf{C}_2$ s.t. $\mathbf{sC}_i \approx x_i\mathbf{sG}$, where $x_1, x_2 \in \{0, 1\}$, we can construct a ciphertext $\mathbf{C}'$ s.t. $\mathbf{sC}' = (1 - x_1x_2)\mathbf{sG}$, i.e. $\mathbf{C}'$ is an encryption of $1 - x_1x_2 =$ NAND (x_1, x_2). After doing that, we will explain how this translates to full homomorphism. We will be guided by the following intuitive observation: since $\mathbf{sC}_i \approx x_i\mathbf{sG}$, then we can think of $\mathbf{C}_i$ as "equivalent" to $x_i\mathbf{G}$ (where the equivalence is expressed by the two being approximately equal under multiplication by $\mathbf{s}$).

To test the validity of this intuition, let us start by trying to implement the negation functionality $x \to (1-x)$. We can verify that indeed setting $\mathbf{C}' = \mathbf{G} - \mathbf{C}_1$, leads to $\mathbf{sC}' \approx (1-x_1)\mathbf{sG}$. Now, let us try to implement conjunction $x_1, x_2 \to x_1x_2$. We notice that $(x_1\mathbf{G}) \cdot \mathbf{G}^{-1}(x_2\mathbf{G}) = x_1x_2\mathbf{G}$, and indeed letting $\mathbf{C}' = \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$, we get:

$$\mathbf{sC}_1\mathbf{G}^{-1}(\mathbf{C}_2) \approx x_1\mathbf{sGG}^{-1}(\mathbf{C}_2) \approx x_1x_2\mathbf{sG} \ ,$$

where as before it is important that $\mathbf{G}^{-1}(\mathbf{C}_2)$ is low norm in order to propagate the validity of the $\approx$ symbol. Putting our two observations together, we have that $\mathbf{C}' = \mathbf{G} - \mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2)$ is indeed an encryption of NAND $(x_1, x_2) = 1 - x_1x_2$. We note that this expression is asymmetric (since $\mathbf{C}_1\mathbf{G}^{-1}(\mathbf{C}_2) \neq \mathbf{C}_2\mathbf{G}^{-1}(\mathbf{C}_1)$) and this asymmetry gives rise to useful properties in terms of efficiency and security [BV14].

Being able to evaluate the NAND function can be extended to evaluating arbitrary boolean circuit using the universality of NAND, as explained above. However, as we noted, the approximation $\mathbf{sC} \approx x\mathbf{sG}$ becomes worse with every gate being evaluated. This puts a bound on the *maximal*

[5] In other words, the *information rate* of this scheme is very low and approaches 0 asymptotically. However, since the ciphertext is still polynomial in the key and message sizes, this is an acceptable solution in a purely theoretical world. Discussion of more efficient solutions will follow.

depth supported by the scheme. The depth bound roughly corresponds to $\log(q/B)$, where B is the bound on the LWE noise distribution. Since the depth of the scheme's decryption circuit grows polynomially with $\log n + \text{loglog} q$ (since it essentially computes an inner product of vectors in $\mathbb{Z}_q^n$), one can choose parameters to allow the evaluation of the decryption circuit, and thus make the scheme bootstrappable (subject to a circular security assumption, if a leveled scheme is not sufficient).

4.3 Efficiency and Implementations

The GSW scheme presented above imposes a high communication and computation overhead compared to performing the evaluation on unencrypted data, which is naturally an undesirable property. Nevertheless, various optimization methods were introduced that reduce the computational overhead to a level that is useful for some applications [DM15, CGGI16]. A significant reduction of the communication overhead for GSW-style schemes remains an open problem.

The information rate overhead problem can be solved in an *amortized* manner using schemes that follow the prior [BV11] paradigm. Such schemes allow to *batch* multiple messages into a single ciphertext in a way that allows to perform homomorphic operations in parallel on all encrypted messages. The ciphertext size grows only mildly with the total amount of information. This idea goes back to the prior work of Smart and Vercauteren [SV10], and was applied to the [BV11] paradigm starting in [BGV12]. The most liberal parameter settings allow to reduce the information rate to a constant, but it is currently unclear, even in this setting, whether it is possible to achieve information rate approaching 1 while preserving full homomorphism.

Using either paradigm, the best efficiency is achieved when using variants of the scheme over polynomial rings. That is, with symbolic polynomials replacing integer vectors, and polynomial multiplication (modulo some ambient polynomial) replacing inner product. This allows for both improved computational complexity and improved information rate. Specifically, current implementations are based either on variants of the NTRU encryption scheme [HPS98] or on the Ring-LWE assumption [LPR10, LPR13].

5 Beyond Vanilla FHE

To conclude this survey, we mention a few uses and extensions of FHE that go beyond the basic functionality.

Multi-Key FHE. The standard notion of FHE only considers a single user who owns data and wishes this data is processed remotely. A natural extension is the case of multiple users, each with their own individually generated secret key and public key, and with their own data, and they wish to outsource a computation on the aggregation of data from all users. To maintain security, it must be the case that decryption of the evaluated ciphertext requires using *all* user secret keys. This notion is called Multi-Key FHE and was first introduced by Lopez-Alt, Tromer and Vaikuntanathan [LTV12]. Their original scheme was based on a variant of the NTRU assumption [HPS98]. A scheme with improved properties and relying on the LWE assumption was later introduced by Clear and McGoldrick [CM15].

Evaluating Quantum Circuits. Considering that a major use of FHE is private delegation of computation suggests considering models where the computational power of the evaluator is qualitatively superior to that of the client. One such case is where the evaluator is in possession of a *quantum computer*. In such case, a classical client may wish to delegate a quantum computation to the evaluator. It was recently shown that this can be achieved under similar assumptions to those required from classical FHE [Mah17].

Acknowledgments

We thank Oded Goldreich for his extensive editorial help.

References

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 284–293. ACM, 1997.

[AP14] 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.

[BG10] Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2010.

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

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In David A. Wagner, editor, *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St.*

John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75. Springer, 2002.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *FOCS*, pages 97–106. IEEE, 2011. Full version in https://eprint.iacr.org/2011/344.pdf.

[BV14] 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.

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 3–33, 2016.

[CM15] Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 630–656. Springer, 2015.

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 542–552. ACM, 1991.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 617–640. Springer, 2015.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.

[Gen09a] Craig Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford University, 2009.

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178, 2009.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 112–131. Springer, 1997.

[GHV10] 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.

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. Separating semantic and circular security for symmetric-key bit encryption from the learning with errors assumption. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 528–557, 2017.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377. ACM, 1982.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1, Basic Techniques.* Cambridge University Press, 2001.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *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.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS98*, pages 267–288, 1998.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 35–54. Springer, 2013.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1219–1234. ACM, 2012.

[Mah17] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. *CoRR*, abs/1708.02130, 2017.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012.

[OPP14] 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.

[Rab79] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA, 1979.

[RAD78] Ron Rivest, Leonard Adleman, and Michael L. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–180, 1978.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *STOC*, pages 84–93. ACM, 2005. Full version in [Reg09].

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[SV10] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography - PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer, 2010.