Exploring Fully Homomorphic Encryption

Alex Grabanski
ajg137@case.edu

5/1/2017

1 Abstract

In 2009, Craig Gentry solved a long-standing problem in Cryptology [11] by presenting the first *Fully-Homomorphic Encryption* (FHE) scheme [5]. FHE allows a certain class of computations represented by polynomial "circuits" to be performed on ciphertexts in such a way that the decrypted ciphertexts inherit the transformation, yet the security of the plaintext is not compromised by the process. While homomorphic encryption schemes existed prior to Gentry's 2009 thesis, all such attempts only were able to represent homomorphisms of groups and other, similarly limited algebraic structures. While only a single operation is needed for e.g. tallying electronic votes, the prior state of the art was a far cry from securely performing arbitrary computations. Gentry's innovation opened wide a realm of new theoretical possibilities for cryptographers, allowing near-arbitrary data processing without compromising data security. While Gentry's scheme (and ensuing variants) is not yet efficient enough to do so, FHE could revolutionize cloud computing, allowing users to securely outsource their data for processing on a remote server. Given the shift toward monolithic web applications and services provided by private companies, FHE may become an excellent way to protect the privacy rights of consumers in practice.

2 Context: Computational Classes

Before launching into an overview of the scheme, it will be useful to understand why Fully-Homomorphic Encryption is the closest we could get to arbitrary computations without violating data security. The history of computer science as a formal discipline arguably began with Gödel, Church and Turing's attempts to define computation [14] in the 1930's. To do so, they worked from the notion of *effective procedures* or algorithms, which are step-by-step mechanical procedures to accomplish a given goal, and gave the notion a formal grounding in several different systems. Gradually, they established that all the

systems they considered (Lambda Calculus, Turing Machines, General Recursive Functions for contributions by Church, Turing, and Gödel, respectively) were equivalent by constructing interpreters for each inside the other. The (at the time) surprising equivalence of the computational power of these different formal systems lead to the formulation of the *Church-Turing Thesis*, which states, roughly, that everything that may be effectively calculated is expressible by a Turing Machine.

Unfortunately, and as Gentry points out, we cannot hope to securely simulate the execution of a Turing Machine on encrypted data. The fatal flaw comes from the ability of Turing Machines to represent arbitrary looping programs, which is precisely what gives them their computational power. Given some algorithm A which computes a function $f(n)$ on some data n and a scheme for securely executing this on a remote machine, an attacker could measure the time $t(n)$ it takes for the remote machine to respond to the request to compute $f(n)$ as part of a side-channel attack. From $t(n)$ and A, we may indirectly gain information about n. While we could adapt such a cryptosystem to e.g. add a random factor to $t(n)$, over time, information would still leak. The only way we could hope to not leak information from such a system would be if we could provide an upper bound on $t(n)$ for all expected inputs, but this is reducible to the Halting Problem, which is undecidable.

Consequently, we must restrict our attention to computations with $t(n) \leq T$ for some T independent of n which may be easily determined *a priori*. Furthermore, we must restrict our computations to be on inputs and outputs of a bounded bit-width for the same concern of side-channel attacks. It turns out that computations of this form may, in general, be expressed by binary combinational circuits. A *combinational circuit* is a composite structure built from boolean logic gates where the output from the circuit depends only on its present inputs. We are unable to express *sequential circuits*, which may have time-and-state-varying outputs, which are necessary components in the construction of real-world computers. However, restricting our attention to this class makes our task much more focused; all we need to do is to allow one *universal* logic gate, like NAND or NOR, to be expressible in a reliable and repeatable manner.

The need to express such boolean logic gates relates to rings and ring homomorphisms in a straightforward manner. Consider the ring $\mathbb{Z}_2$, which has two elements: "0" (the additive identity, which we suggestively name "FALSE"), and "1" (the multiplicative identity, sometimes called "TRUE".) Then, for any $a, b \in \mathbb{Z}_2$, note that the table of outputs for $a + b$ matches the truth table for a XOR b, and the table of outputs for $a \times b$ matches the truth table for a AND b, under our suggestive naming scheme. Since "TRUE XOR a" is equivalent to "NOT a", we can form NAND gates. Consequently, boolean logic circuits with a single-bit output may be represented by multivariate polynomials in $\mathbb{Z}_2$ by assigning a variable to each input bit. The goal of FHE, then, is to be able to securely apply the polynomial evaluation homomorphisms

$$E_{a_1,a_2,...a_n} : R[x_1, ...x_n] \to R \quad \text{by} \quad E(f(x_1, ...x_n)) = f(a_1, ...a_n)$$

$$\text{for} \quad a_1, ...a_n \in R$$

for a ring R with a homomorphism ϕ to $\mathbb{Z}_2$, while making it difficult to explicitly compute ϕ without the private key.

In theory, we could reduce all such circuits to their standard sum-of-products representation, but in practice, this may cause the size of the polynomial representation of the circuit to expand exponentially compared to a more compact, factored representation. Define the *depth* of a circuit to be the maximum number of nested operations in the circuit. A *somewhat homomorphic* encryption scheme is then defined to be a restricted version of a fully-homomorphic scheme which only works on circuits up to a given depth limit. The origin of such a notion will be explained later, when we describe the gory details of their implementation based on ideal lattices. Despite their limited capabilities, somewhat homomorphic schemes may find uses on their own, and will prove to be integral to the construction of Gentry's FHE.

3 History of the Problem

The original challenge to construct a fully-homomorphic encryption scheme was posed by Rivest et. al in 1978, the same year as Rivest et. al's introduction of the RSA public-key cryptosystem [12]. In "On Data Banks and Privacy Homomorphisms", Rivest et. al. demonstrate that the RSA cryptosystem has the property that the encryption of the product of plaintexts is the product of the encryptions of plaintexts, making the encryption function into a group homomorphism. In the same paper, they demonstrate other basic examples of encryption schemes able to perform limited sets of operations homomorphically on encrypted data. However, they note that "comparison operations are not possible" under any of the schemes they proposed, and so they posed the question of "whether it is possible to have a privacy homomorphism with a large set of operations which is highly secure." As an example of the hypothetical usefulness of such a "privacy homomorphism," they consider the example of a loan servicer running computerized operations securely on a (then-common) time-sharing system.

Over time, the motivation among researchers to find more such "privacy homomorphisms" only increased, and cryptosystems began to be constructed explicitly for the purpose of the performance of homomorphic operations. One such example (which Gentry points to) is the Boneh-Goh-Nassim (BGN) cryptosystem, which is capable of evaluating arbitrary quadratic forms homomorphically [2]. A major motivating factor for constructing such cryptosystems was provided by the tantalizing possibility of cryptographic voting schemes, which was responsible in part for the creation of the Paillier and Benaloh cryptosystems [10]. Over time, as homomorphic encryption schemes grew more elaborate, researchers slowly expanded the set of functions that could be securely evaluated, but there seemed to be little hope for an asymptotically efficient scheme which could evaluate arbitrary polynomials.

In 2009, Craig Gentry, advised by Boneh for his doctoral dissertation, found the holy grail of research into the theory of homomorphic cryptosystems by providing an explicit construction of a fully-homomorphic encryption scheme. Here, the story begins.

4 The Cryptosystem, Abstractly

At a very high level, Gentry defines the following operations that any fully or somewhat-homomorphic encryption scheme ϵ must implement (p23):

$$\text{KeyGen}_\epsilon : \{0,1\}^* \times \mathbb{N} \to \mathcal{K} \times \mathcal{K}$$
$$\text{Encrypt}_\epsilon : \mathcal{K} \times \mathcal{P} \to \mathcal{C}$$
$$\text{Decrypt}_\epsilon : \mathcal{K} \times \mathcal{C} \to \mathcal{P}$$
$$\text{Evaluate}_\epsilon : \mathcal{K} \times \mathfrak{C}_\epsilon \times \mathcal{C}^n \to \mathcal{C}$$

where $\mathcal{K}$ is the key space, $\mathcal{P}$ is the message space, $\mathcal{C}$ is the cipher space, and $\mathfrak{C}_\epsilon$ is the space of all circuits supported by the scheme ϵ.

The first three operations are nothing new for anyone familiar with public-key cryptography. KeyGen_ϵ takes a random string of bits and the value of the *security parameter* λ and returns a (secret key, public key) pair, (sk, pk) for short. Encrypt_ϵ encrypts messages with the public key to be decrypted by Decrypt_ϵ using the secret key.

The new element is the Evaluate_ϵ operation, which must satisfy the following correctness property:

$$\forall R \in \{0,1\}^*, \lambda \in \mathbb{N}, \quad \text{if} \quad (pk, sk) = \text{KeyGen}_\epsilon(R, \lambda),$$
$$\text{then} \quad \forall C \in \mathfrak{C}_\epsilon, \quad \pi_1, ...\pi_n \in \mathcal{P} \quad \text{with} \quad \psi_i = \text{Encrypt}_\epsilon(pk, \pi_i),$$
$$\text{Decrypt}_\epsilon(sk, \text{Evaluate}_\epsilon(pk, C, (\psi_1, ...\psi_n))) = C(\pi_1, ...\pi_n)$$

However, as Gentry points out (p5), we need to rule out trivial definitions of Evaluate_ϵ which do nothing but pass C along, relying on an expressive cipher space and on Decrypt_ϵ to perform the computation of $C(\pi_1, ...\pi_n)$. To do so, he makes the insight that if Decrypt_ϵ *itself* is representable by a circuit $\mathcal{D}_\epsilon \in \mathfrak{C}_\epsilon$ of size polynomial in λ, it is unable to evaluate the larger circuits in $\mathfrak{C}_\epsilon$. Gentry calls this last property *compactness*, and then defines a fully-homomorphic encryption scheme as something following the above format which is compact and correctly evaluates all possible circuits.

The insight to encode the decryption algorithm as a circuit $\mathcal{D}_\epsilon$ is what provided Gentry with a natural path to construct a FHE scheme. Suppose that we have a somewhat-homomorphic scheme $\epsilon^{(d)}$ which is able to evaluate all circuits with depth at most d, starting from freshly-encrypted ciphertexts. Intuitively, the somewhat-homomorphic scheme causes ciphertexts to "decay" over a sequence of operations, until decryption would eventually violate correctness. To get around this, if we could find a way to securely *re-encrypt* the

result of the computation so as to appear like a freshly-encrypted ciphertext, FHE would be achievable by breaking up any circuit into components of depth at most d, re-encrypting intermediate values as needed.

Given $\mathcal{D}_\epsilon$, Gentry does exactly that (p44) with the new operation $\text{Recrypt}_\epsilon : \mathcal{K} \times \mathfrak{C}_\epsilon \times \mathcal{C} \times \mathcal{C} \to \mathcal{C}$, defined as:

$$\text{Recrypt}_\epsilon(pk, \mathcal{D}_\epsilon, esk, \psi) = \text{Evaluate}_\epsilon(pk, \mathcal{D}_\epsilon, (esk, \text{Encrypt}_\epsilon(pk, \psi)))$$

where pk and $\mathcal{D}_\epsilon$ are as above, ψ is the ciphertext to re-encrypt, and esk is the encryption of the secret key sk under the public key pk. Note that to re-encrypt a ciphertext, we don't need sk, but we need esk, which could present a vulnerability. In particular, the encryption of the secret key could reveal information about the secret key, making the system easier to attack. Gentry manages to show that under certain conditions, the scheme is secure nevertheless (p53.)

Recrypt_ϵ works exactly because with esk, we can decrypt a ciphertext without ever leaving cipher space using a single evaluation of the decryption circuit. Initially, ψ is encrypted, resulting in a message that has been encrypted twice. When we evaluate the decryption circuit, the inner encryption (the one which may have accumulated errors) is lifted. However, note that in the process, we have introduced a source of error into the new ciphertext by evaluating the decryption circuit. For Recrypt_ϵ to be practical, we need to ensure that the depth of the decryption circuit is small enough that we can meaningfully reduce the error in the input ciphertext, otherwise we may need to apply Recrypt_ϵ on each ciphertext more than once to be able to perform more operations.

5 Applications

Given the above blueprint for a fully-homomorphic cryptosystem, Gentry points out a number of hypothetical use-cases for the system. Aside from the more obvious applications to privacy in cloud computing, Gentry points out (p25) that the definition of Recrypt_ϵ allows for straightforward *proxy re-encryption*. Proxy re-encryption allows a ciphertext to be re-encrypted under a new public key by a third party without needing to know the original private key. By modifying the definition of Recrypt_ϵ to read $\text{Evaluate}_\epsilon(pk2, \mathcal{D}_\epsilon, (esk1, \text{Encrypt}_\epsilon(pk1, \psi)))$, messages encrypted under $pk1$ may be re-encrypted under $pk2$ using esk as a tag. Proxy re-encryption lets one party in a computation delegate decryption of a message to a third party without decrypting the message or sharing their secret key, which can be useful in multi-party environments.

Gentry also provides examples of how the cryptosystem could be useful in constructing non-interactive zero-knowledge proofs, software protection schemes, and secure multi-party computation schemes. The potential utility of a FHE scheme in these endeavors was realized long before Gentry's result, but now researchers are able to evaluate the practical aspects of using such a scheme as a building-block in other cryptosystems. By far, the most-hyped application of

FHE is the new ability for remote servers to process sensitive user data without ever knowing anything about the users. Gentry provided an example (p21) of a search engine whose queries are never revealed to the remote server, but many other researchers have provided a great variety of other examples. For example, FHE has been applied to biological research by enabling string searches through gene sequences without compromising the privacy of individuals in a database [7]. More generally, FHE can support statistical analysis of a large body of participant-based data without revealing the identities of the participants [8]. Outside of medicine and data analysis, FHE has also been proposed as a way to allow true, secure blind auctions over the internet [6].

Given certain political developments in the United States over the past ten years, such as the revelations about the NSA's warrantless metadata collection, or more recently, such as the repeal of FCC internet privacy rules under the Trump administration, the data privacy rights of average Americans are being threatened. For both of those examples, the threat to consumer privacy originates from the reliance on a trusted third party, namely, telecommunications companies. Unfortunately, fully-homomorphic encryption does not protect against this threat any more than traditional cryptosystems. However, given the reliance of consumers on private companies such as Google, Yahoo, or Facebook to provide web-based applications, FHE schemes could provide an additional layer of security against data breaches of private servers.

6 Ideal Lattices

The abstract scheme given previously, while a good overview of the approach, does not describe how we can securely perform primitive operations in the first place. Moreover, the scheme above fails to describe why we might encounter "errors" in the ciphertexts after performing a sequence of operations, creating a need for re-encryption. It turns out that these "errors" have a very nice geometric interpretation in the original, lattice-based version of Gentry's scheme.

First, a *lattice* L is a subgroup of $\mathbb{R}^n$ isomorphic to $\mathbb{Z}^n$ such that $span(L) = \mathbb{R}^n$ [13]. Visually, the lattice L is, as a set, the collection of corners of a tiling of $\mathbb{R}^n$ by parallelpipeds. While this definition may give the impression that lattices are too simple a primitive to build cryptosystems on, hard problems on lattices are not hard to find. Define a *basis* B for the lattice L as an ordered collection of n linearly-independent vectors which span L for linear combinations with coefficients in $\mathbb{Z}$. Given a lattice basis B, denote the lattice generated by B as $\mathcal{L}(B)$. Then, given a norm $||.||$ on $\mathbb{R}^n$ and a lattice basis B, the *Shortest Vector Problem* is to find a vector v in $\mathcal{L}(B)$ which minimizes $||v||$. A more general version of the Shortest Vector Problem is the *Closest Vector Problem*, which modifies the Shortest Vector Problem to measure distances from some arbitrary, fixed basepoint $b \in \mathbb{R}^n$ instead of the origin.

While simple-seeming, the Closest Vector Problem with the Euclidean norm is NP-Complete. A straightforward proof of this fact as described in [9] is by an

easy reduction of the subset sum problem, a specialized version of the knapsack problem, to CVP. The security of Gentry's cryptosystem's ultimately relies on the hardness of a variant of SVP called the Shortest Independent Vector Problem, which is to find a linearly-independent set of n "short" vectors in L. Intuitively, solving the SIVP yields a near-standardized basis for the lattice L which illuminates the geometry of L by allowing us to recover the fundamental parallelpiped in the tiling we described earlier.

However, lattices have no notion of a ring structure. For that, Gentry turns to *ideal lattices*, which may be described as follows: Consider the ring $\mathcal{R} = \mathbb{Z}[x]/f(x)$ where $deg(f) = n$, and suppose that we have an ideal I in $\mathcal{R}$. Then $\mathcal{R}/I$ is a ring with a particularly nice representation: As a group, $(\mathcal{R}, +) \cong \mathbb{Z}^n$, by considering the coordinates as coefficients in (congruence classes of) polynomials of degree strictly less than n. Then, since $(I, +)$ is a subgroup of $(R, +)$, under the isomorphism, I provides a sub-lattice of $\mathbb{Z}^n$, thought of as sitting inside $\mathbb{R}^n$.

Since any ideal I may be identified with a lattice $\mathcal{L}(I)$, we can consider lattice bases B_I corresponding to ideals in $\mathcal{R}$, whence we can perform computations on representatives of $\mathcal{R}/I$ in a very straightforward manner. For a vector $v \in \mathbb{Z}^n$ corresponding to an element $r \in \mathcal{R}$, let $v \bmod B_I$ be defined as a vector in $\{v + \sum_i a_i \vec{b_i} \,|\, \forall i \quad a_i \in \mathbb{Z} \wedge \vec{b_i} \in B_I\} \cap B$, where B is a copy of the parallelpiped generated by the vectors in B_I translated to have its center at the origin. Then, ignoring the boundary of B, $v \bmod B_I$ uniquely represents the congruence class of r in $\mathcal{R}/I$, and it may be easily computed as $v - \boldsymbol{B}_I * [\boldsymbol{B}_I^{-1} v]$, where $\boldsymbol{B}_I$ denotes the matrix whose columns are vectors in B_I, and $[.]$ denotes "round the vector's components to the nearest integer."

Then, note that addition in $\mathcal{R}/I$ corresponds to vector addition in $\mathbb{Z}^n$ modulo B_I, and multiplication in $\mathcal{R}/I$ corresponds to a rather strange $\mathcal{R}$-bilinear operator $\times : \mathbb{Z}^n \times \mathbb{Z}^n \to \mathbb{Z}^n$ in the $\mathcal{R}$-module $\mathbb{Z}^n$, only taken modulo B_I. As a consequence of the metric structure of $\mathbb{R}^n$, it's natural to consider the effect of these operators on (Euclidean) vector norms before taking their results modulo B_I. By the triangle inequality, for any vectors $a, b \in \mathbb{Z}^n$, $||a+b|| \leq ||a||+||b||$, and as a consequence of the bilinearity of $\times$, $||a \times b|| \leq \gamma ||a|| ||b||$ for some constant γ dependent on $\mathcal{R}$. These bounds will be closely related to the accumulated evaluation "error" in Gentry's somewhat-homomorphic scheme, and with this rich geometric structure on the ring operations in $\mathcal{R}/I$, we are now prepared to describe the scheme.

7 A Somewhat-Homomorphic Cryptosystem

Suppose we have $\mathcal{R} = \mathbb{Z}[x]/f(x)$ with $deg(f) = n$ as before, and let I and J be two relatively-prime ideals of $\mathcal{R}$ ($I + J = R$). Then, the public key pk is (B_I, B_J^{pk}, D) for lattice bases of $\mathcal{L}(I)$, $\mathcal{L}(J)$, respectively and D a probability distribution over I. The secret key sk is another basis B_J^{sk} for $\mathcal{L}(J)$. Intuitively, KeyGen_ϵ will try to pick a "good" basis for J for the secret key to make decryption easy, but a "bad" basis for J as part of the public key.

To encrypt $\pi \in \mathcal{P}$, we compute $\psi = \text{Encrypt}_\epsilon(pk, \pi) = (\pi + i) \bmod B_J^{pk}$ where $i \in I$ is randomly sampled from D. Then, note that $\psi = \pi + i + j$ for some $j \in J$.

To decrypt $\psi \in \mathcal{C}$, we compute $\pi = \text{Decrypt}_\epsilon(sk, \psi) = (\psi \bmod B_J^{sk}) \bmod B_I$. For this to work, we need $\psi \bmod B_J^{sk} = (\pi + i)$, which imposes a constraint on B_J^{sk}. In particular, we need $\mathcal{P} + \mathcal{D} \subseteq B$ for B the origin-centered parallelpiped generated by B_J^{sk}.

To evaluate a circuit $C \in \mathfrak{C}_\epsilon$ on ciphertexts $\psi_k = \pi_k + i_k + j_k$, simply compute $C(\psi_1, ...\psi_m) \in C(\pi_1 + i_1, ...\pi_m + i_m) + J$. In particular, $C(\psi_1, ...\psi_m) \in (\pi + i) + J$ for some π and some i.

Let r_{DEC} be the inball of the parallelpiped B. Viewing $(\pi + i)$ as an offset from the lattice generated by J, note that so long as $||H(\pi_1 + i_1, \pi_2 + i_2)|| \leq r_{DEC}$, we may perform the operator H on ψ_1 and ψ_2 without worrying about winding up in the wrong congruence class modulo I due to our " mod B_J^{sk} " operation in decryption.

Given our earlier bounds on vector norms after performing $+$ and $\times$, we can see that if both operands have offsets of magnitude $\leq r$, then the magnitude of the result of addition will be $\leq 2r$, and the magnitude of the result of multiplication will be $\leq \gamma r^2$. Consequently, the growth in offsets under repeated squaring dominates the growth in offsets under repeated doubling, with the magnitude of the offset of a value squared k times being bounded above by $\gamma^{(2^{k-1})} * r^{(2^k)}$. As a result, we can only safely evaluate circuits whose depth k is such that $\gamma^{(2^{k-1})} * r^{(2^k)} \leq r_{DEC}$ on inputs with offsets of magnitude smaller than r. Since the offsets exhibit doubly-exponential growth, k may be bounded by a factor growing as $log(log(r_{DEC}))$, or more precisely as $log(log(r_{DEC})) - log(log(\gamma * r))$ (Gentry 1.4.1).

The doubly-logarithmic bound on the depth of circuits makes the task of constructing a decryption circuit difficult. Gentry notes (p97) that the most straightforward construction of the decryption circuit requires a slightly higher asymptotic circuit depth bound to succeed. To get around this, Gentry uses several tricks to make the job of the decryption circuit easier. For now, note that while the scheme above is not fully-homomorphic, it is just as useful as FHE for special situations where the circuits we want to evaluate fall within the depth bound. As a toy example, using the above bound and ignoring the "$-log(log(\gamma * r))$" term for now, we can evaluate a single multiplication if $4 \leq r_{DEC}$. We can evaluate arbitrary quadratic forms on n variables if we can evaluate sums of products with $\frac{n^2}{2}$ summands, and since the error induced by addition is additive in the scheme, we can do so if $2n^2 \leq r_{DEC}$. Due to the term we ignored, r_{DEC} will need to be slightly higher and will depend on the size bounds on the inputs.

8 Security

The security of the partially-homomorphic scheme reduces to the question of finding bases B_J from B_J^{pk} for the lattice $\mathcal{L}(J)$ such that the parallelpiped spanned by B_J contains $\mathcal{P} + \mathcal{D}$. We could do this easily if we had access to a Shortest Independent Vector Problem oracle, because we could use its output on B_J^{pk} to determine the fundamental parallelpiped for the lattice J and then scale it appropriately to contain $\mathcal{P} + \mathcal{D}$. Using a chosen-ciphertext attack, we could then send ciphertexts to the system for processing and eventually tweak B_J using our shortest independent vectors to match the statistical profile of operations performed with respect to the secret key B_J^{sk}.

While the above observation shows that access to a SIVP oracle is sufficient, in principle, to launch an attack on the cryptosystem, it takes a lot more work to provide evidence that it is necessary. Chapter 19 of Gentry's thesis shows exactly that, given his definition of the KeyGen$_\epsilon$ algorithm.

However, even if the KeyGen$_\epsilon$ algorithm ensures security against bad choices of J, Gentry points out that it is possible that the choice of the ring $R = \mathbb{Z}[x]/f(x)$ could expose a weakness if $f(x)$ isn't chosen judiciously. For example, $f(x) = x^n - 1$, which yields the class of *circulant ideal lattices*, greatly simplifies many operations and the resulting decryption circuit. In addition, circulant lattices maximize the depth of evaluable circuits for a given lattice dimension n by setting $\gamma = \sqrt{n}$. However, Gentry and Szydlo demonstrated an attack on the system under this choice which is effective if n is composite or if the ideal I has an orthonormal basis. Gentry notes (p68) that while circulant lattices can be weaker than other choices, the same weaknesses also plague the widely-available cryptosystem NTRU, and that the performance gains from using such lattices may outweigh the potential cost to security. In light of the unresolved questions about the hardness of the SIVP of some particular ideal lattices, Gentry devotes the penultimate chapter of his thesis (19) to a modified version of the scheme which reduces to the worst-case hardness of SIVP for ideal lattices. With this modification, effective attacks on the system would need to leverage properties common to all ideal lattices. The question of the existence/non-existence of effective attacks on this modified scheme is open.

9 Toward Fully-Homomorphic Encryption

In order to make the partially-homomorphic scheme into a fully-homomorphic scheme, Gentry came up with several tricks and modifications to construct a partially-homomorphic scheme capable of evaluating its own decryption circuit. First, recall that $\text{Decrypt}_\epsilon(sk, \psi) = (\psi \bmod B_J^{sk}) \bmod B_I$. Together with the definition of mod, the right-hand side may be written as $(\psi - B_J^{sk1}[B_J^{sk2}\psi]) \bmod B_I$ where $B_J^{sk1} = B_J^{sk}$ and $B_J^{sk2} = (B_J^{sk})^{-1}$, strictly following the definitions above. Note that in the above expression, the decryption circuit does not need to compute the inverse of any matrix, so long as the encryption of the secret key included in the ciphertext includes both B_J^{sk1} and B_J^{sk2}. Consequently,

the dominant contributions to the depth of the decryption circuit come from the implementations of the inner matrix-vector multiplication and the vector rounding operator $[.]$. One particularly hairy detail is that the result of $B_J^{sk2}\psi$ need not live in $\mathbb{Z}^n$, so the decryption circuit will need to internally emulate fixed-point binary arithmetic.

Gentry tackles some of this complexity (8.4) by imposing what seems, at first, to be an odd restriction. The depth limit for circuits was $log(log(r_{DEC})) - log(log(\gamma * r))$, but Gentry restricts the set of allowable circuits to those of depth within $log(log(r_{DEC}/2)) - log(log(\gamma * r))$. Seemingly, the problem has gotten worse, since the goal is to be able to evaluate deep circuits. However, note that the depth lost by this change decreases as r_{DEC} increases and is less than 1 if $r_{DEC} > 4$, which was the conservative lower bound to be able to perform multiplications. If we somehow use the restriction on allowed circuits to decrease the depth of the decryption circuit by at least two levels, this change will yield a net improvement. Gentry proves in a lemma (8.4.2) that making the change results in $B_J^{sk2}\psi$ always having coordinate values within $\frac{1}{4}$ of an integer. With this property, Gentry condenses the first matrix-vector multiplication and the rounding step into a single step, which allows room to ignore small contributions to the matrix-vector product under certain conditions. By truncating intermediate results in the calculation, operations such as addition and multiplication require less depth, since fewer carries need to be performed in sums.

Unfortunately, even after restricting the allowable circuits and taking advantage of rounding, the decryption circuit is still too deep to be evaluated. Without doing something crazy, like offloading work on decrypting ciphertexts to the encrypter, it would seem that the story ends here, and that fully-homomorphic encryption is not possible under this approach. On the other hand, imagine that we are crazy. If we could cut out the inner matrix multiplication step in binary fixed-point arithmetic by using hints provided by the encrypter, the complexity of the decryption circuit would decrease radically. Gentry does exactly this (10.2), and calls the technique "squashing the decryption circuit." Instead of carrying around B_J^{sk2} in the encrypted private key, the entity performing the initial encryption of the plaintext finds a set of matrices $\mathcal{A} = \{A_1, ...A_n\}$ and a subset S of them which sums to B_J^{sk2}. The set $\{A_1, ...A_n\}$ is then included in the public key, and a binary string representing membership of elements of S in $\mathcal{A}$ replaces B_J^{sk2} in the secret key and its encryption. Then, whenever the decryption circuit needs to be evaluated on the ciphertext ψ, the evaluator computes every $A_1\psi, ...A_n\psi$ in public and feeds the results to the decryption circuit. Since ψ contains a representation of S, all the decryption circuit needs to do to compute $B_J^{sk2}\psi$ is to evaluate the sum $\sum_{A_i \in S} A_i\psi$.

With this modification and the simplifications due to rounding, the resulting decryption circuit fits within the depth limit. Consequently, this modified scheme is fully-homomorphic, but a more careful analysis is required to show that it is still an effective encryption scheme. In particular, we need to ensure that knowledge of $A_1\psi, ...A_n\psi$ for every ciphertext ψ we homomorphically decrypt does not give information about S without significant effort. Since the

linearity of matrix-vector multiplication can be exploited, this reduces to determining S given $\{A_1, ...A_n\}$ and the encryption of the secret key. Gentry defines (p19) a computational problem called the Sparse Vector Subset Sum Problem (SVSSP) and reduces it to the previous problem. SVSSP is just like the subset sum problem used in knapsack cryptosystems over vectors, but with the knowledge that the size of S is small compared to n. The best-known attack on the scalar version, the sparse subset sum problem, is exponential in the size of the subset. As a result, the work performed in public does not compromise the security of the cryptosystem assuming that security parameters are chosen with care.

10 Modifications

While the above scheme is a monumental theoretical achievement, the computation time required to perform operations in the scheme is fairly high. In particular, if λ is the security parameter of the scheme, where the best-known attacks on the embedded SIVP and SVSSP problems asymptotically dominate 2^λ in runtime, each computation will require a secret key of size $O(\lambda^7)$ and computation quasi-linear in λ^9 (p117.) Theoretically, this runtime is nice, since it is polynomial in the security parameter, but the degree of the polynomial is too high for practical usage.

Since Gentry's PhD thesis, a number of iterative improvements have been made upon Gentry's fully-homomorphic system, and fully-homomorphic systems have been constructed outside the context of lattice cryptography which utilize the same basic principles. To deliver a more succinct exposition than his nearly 200-page thesis, Gentry also presented a version of his fully-homomorphic scheme over the integers [15], which uses ideals in $\mathbb{Z}$ and replaces the n-dimensional geometric realization of lattices with simple integer addition and multiplication, where bounds are derived using the absolute value function instead of the Euclidean norm. Unfortunately, the security of this conceptually-simpler system reduces to the Approximate-GCD problem, which may be easier to solve than SVSSP and SIVP. In fact, Chen et. al demonstrated attacks on this version of the cryptosystem which can break the encryption under an 89MB public key in 190 days on a single-core desktop computer with 72GB of RAM [4]. For this reason, most fully-homomorphic cryptosystems utilize lattice-based cryptography, and generally follow Gentry's blueprint, but with optimizations to support massively-parallel processing or to cut down on the asymptotic runtime of computations in the security parameter [1].

Since few researchers deviate from Gentry's blueprint, it is tempting to say that the abstract format of the scheme, involving bootstrapping, is unlikely to change. However, Gentry et. al. demonstrated in 2014 that with a technique called "modulus switching," a FHE scheme could be constructed without needing to evaluate the decryption circuit [3]. Unfortunately, the per-operation runtime of this modified scheme is cubic in the depth of evaluated circuits, and so it is impractical for homomorphically evaluating high-degree

polynomials. Luckily, bootstrapping may be introduced into the scheme as "an optimization" to deal with deep circuits to achieve an overall per-operation runtime linear in the number of gates and in the security parameter.

11 Conclusion

Given the developments above, the future of fully-homomorphic encryption looks bright. While existing implementations are impractical, there is hope that some day, we may be able to securely operate on encrypted data as if it were plain-old-data with a minimal performance penalty. If that ever happens, we could expect to see a renaissance in the enforcement of privacy rights of consumers in the face of monolithic and opaque third parties. More than an important theoretical development, Gentry's revolution may eventually cascade into a radical re-thinking of the security of cloud computing and web applications and allow commerce without possibility of clandestine surveillance.

References

[1] Nathanael Black. Homomorphic encryption and the approximate gcd problem. 2014.

[2] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography Conference*, pages 325–341. Springer, 2005.

[3] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13, 2014.

[4] Yuanmi Chen and Phong Q Nguyen. Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 502–519. Springer, 2012.

[5] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[6] Jong-Hyuk Im, Taek-Young Youn, and Mun-Kyu Lee. Privacy-preserving blind auction protocol using fully homomorphic encryption. *Advanced Science Letters*, 22(9):2598–2600, 2016.

[7] Yu Ishimaki, Kana Shimizu, Koji Nuida, and Hayato Yamana. Poster: Privacy-preserving string search for genome sequences using fully homomorphic encryption. *Bioinformatics*, 2016.

[8] Wen-jie Lu, Shohei Kawasaki, and Jun Sakuma. Using fully homomorphic encryption for statistical analysis of categorical, ordinal and numerical data. 2017.

[9] Daniele Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Transactions on Information Theory*, 47(3):1212–1215, 2001.

[10] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

[11] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. 1978.

[12] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. 1978.

[13] Joseph H Silverman. An introduction to the theory of lattices and applications to cryptography. 2006.

[14] Robert I Soare. Turing oracle machines, online computing, and three displacements in computability theory. *Annals of Pure and Applied Logic*, 160(3):368–399, 2009.

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