Technical Report No. 2018-636

FULLY HOMOMORPHIC ENCRYPTION: A GENERAL FRAMEWORK AND IMPLEMENTATIONS

Selim G. Akl
School of Computing
Queen's University
Kingston, Ontario K7L 3N6
Canada
akl@cs.queensu.ca

Ibrahim Assem
Département de Mathématiques
Université de Sherbrooke
Sherbrooke, Québec J1K 2R1
Canada
Ibrahim.Assem@USherbrooke.ca

November 7, 2018

Abstract

Two data items a and b are stored on an untrusted database in their encrypted forms $E(a)$ and $E(b)$, respectively. It is required that the encryption function be fully homomorphic, that is, E must satisfy the conditions $E(a)+E(b) = E(a+b)$ and $E(a)\times E(b) = E(a\times b)$. This would allow the owner of the data, as well as an untrusted party operating the database on behalf of the owner, to perform operations directly on the encrypted data without the need for decryption. We propose a general framework that allows the derivation of such fully homomorphic functions. Two implementations of the general framework are then described that, in addition to being secure, enjoy the property of being efficient to compute.

1 Introduction

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite set of nonnegative integers representing data to be stored securely on an untrusted cloud service. It is not necessary, though possible, that the set X be stored locally by its owner. The data are encrypted in order to protect the secrecy and integrity of their information. It is required that the data be manipulated remotely and directly in their encrypted form through addition and multiplication operations, such that the sum/product of two encrypted data results in the encryption of the sum/product of the original values. In other words, let E be the encryption function, defined over the set $\mathbb{Z}$ of all integers. For any two elements of X, for example, x_i and x_j used henceforth for illustration, we need to have:

$$E(x_i) + E(x_j) = E(x_i + x_j)$$

and

$$E(x_i) \times E(x_j) = E(x_i \times x_j).$$

This means that both the domain and the range of the function E are equipped each with an addition and a multiplication. Such algebraic systems are called rings in mathematics and the encryption function E is then what is called a ring homomorphism. In cryptography, an encryption function E satisfying the two equalities above is said to be *fully homomorphic*.

Ever since the idea of homomorphic encryption was articulated over forty years ago [15], several homomorphic encryption schemes have been developed, ranging from so-called 'partial' or 'somewhat' homomorphic encryption schemes to fully homomorphic encryption schemes; see, for example, [2, 6, 7, 8, 14], and for surveys [1, 5, 12]. These schemes vary widely in terms of the intricacy of their algorithms, their computational requirements, and the security they provide [3, 4, 11, 13].

In this paper we prove that, if we deal with a ring having a reasonable notion of division (like the ring of integers), then the function associating to each element of the ring its remainder upon division by a fixed element (namely, the secret encryption key) can be used as an efficient and secure encryption homomorphism. We derive a general framework for fully homomorphic encryption, and illustrate the general approach with

two specific implementations. The resulting cryptosystems, one for integers and one for polynomials possess the important property of allowing an arbitrary number of additions and multiplications to be performed, in any order, on the encrypted data.

The remainder of the paper is organized as follows. Section 2 offers some mathematical preliminaries and provides a description of our general framework for fully homomorphic encryption. Two implementations of the general framework are proposed is Section 3. Some closing thoughts are presented in Section 4.

2 The mathematical setting

For the benefit of completeness, we begin with some background from Algebra. This is followed by a presentation of our proposed general framework.

2.1 Algebraic preliminaries

In mathematics, a *ring* is a set with two algebraic operations, addition and multiplication, satisfying a set of axioms. Typical examples of rings are the set $\mathbb{Z}$ of integers or the set of all square matrices of a fixed size with integral entries. Here, we deal only with *commutative* rings, that is, rings A such that, for any $a, b \in A$, we have $a \times b = b \times a$.

In order to have a reasonable concept of division, we need a further condition: our commutative ring must be what is called an *integral domain*, namely if two elements $a, b \in A$ are nonzero, then their product $a \times b$ is nonzero. Examples of integral domains are the set $\mathbb{Z}$ of integers, or the sets $\mathbb{R}[t], \mathbb{Z}[t]$ of polynomials in one undeterminate t with real, or integral coefficients, respectively.

An integral domain A is called an *Euclidean domain* if it admits a so-called *valuation*, that is a map φ from the set of nonzero elements of A to the set of natural numbers $\mathbb{N}$ such that:

1. If a, b are two nonzero elements of A, then $\varphi(a) \leq \varphi(a \times b)$
2. If $a \in A$ is arbitrary and $b \in A$ is nonzero, then there exist $q, r \in A$ such that $a = q \times b + r$ with either $r = 0$ or $\varphi(r) < \varphi(b)$.

That is, Euclidean domains are integral domains in which division is possible. The elements q, r in condition 2. above are respectively called the *quotient* and the *remainder* of the division of a by b. It is important to observe that in a general Euclidean domain, quotient and remainder are not necessarily unique. This is the case if and only if the valuation φ satisfies the following additional condition: if a, b are two nonzero elements of A, then $\varphi(a + b) \leq \max\{\varphi(a), \varphi(b)\}$.

An example of an Euclidean domain is the set $\mathbb{Z}$ of integers, with valuation function $\varphi(a) = |a|$ for $a \neq 0$. If one assumes moreover that the remainder is nonnegative, then division has unique quotients and remainders. An example of an Euclidean domain where division has unique quotients and remainders (without additional hypothesis) is the set $\mathbb{R}[t]$ of polynomials in one undeterminate t with real coefficients, with valuation function $\varphi(p) = \deg(p)$ the degree of p. On the other hand, $\mathbb{Z}[t]$ is not an Euclidean domain: one can only divide by polynomials whose leading coefficient is 1 or -1, but then, the quotient and the remainder are unique.

We shall freely use concepts and results about integral domains and division as can be found in any algebra book, see, for example, [9].

2.2 A general framework for fully homomorphic encryption

Let A be an Euclidean domain having unique quotients and remainders, and the secret encryption key m be a fixed element of A. The latter is known only to the owner of the data X and stored safely locally. Then m generates the ideal Am consisting of all multiples $a \times m$ with $a \in A$ and therefore the quotient ring A/Am consisting of all residual classes of elements of A modulo m.

Elements of this quotient ring can be identified with elements r of A such that $r = 0$ or $\varphi(r) < \varphi(m)$. However, if one does so, addition and multiplication of such elements have to be performed modulo m, that is, if r, r' are such that $\varphi(r) < \varphi(m)$ and $\varphi(r') < \varphi(m)$, then $r + r'$ is by definition the remainder of division of the sum of r, r' in A upon division by m. Similarly, $r \times r'$ is the remainder of division of the product of r, r' in A upon division by m.

Consider the function $E : A \to A/Am$ mapping each element of A onto the residual class modulo m containing it. Using the identification above, one can alternatively map each $x \in A$ onto an element x' such

that $x = x' + q \times m$ with either $x' = 0$ or $\varphi(x') < \varphi(m)$. Because the remainder x' is unique, this is a well-defined function with values in A/Am.

This function is well-known to be a ring homomorphism, called the projection of A onto A/Am. We give the proof for the benefit of the reader. Let $x_i, x_j \in A$, then $E(x_i + x_j)$ is the class containing the remainder of the division of $x_i + x_j$ by m. However, $x_i + x_j = {x_i}' + {x_j}' + (q_i + q_j) \times m$. Therefore, this remainder is the same as that of ${x_i}' + {x_j}' = E(x_i) + E(x_j)$. Hence $E(x_i + x_j) = E(x_i) + E(x_j)$ in A/Am. One proves in exactly the same manner that $E(x_i) \times E(x_j) = E(x_i \times x_j)$. One also has $E(1) = 1$, even though this is not needed.

The function E is surjective but by no means injective, thus it is not an isomorphism of rings. Nonetheless, decryption can easily be performed. Indeed, knowledge of m, q, x' allows an easy computation of $x = x' + q \times m$. It is worth observing here that all operations mod m are to be performed by the user (owner of the data set X) *locally*, at a secure location. To emphasize, no mod m operation is performed on the cloud facility, for this would necessarily divulge m. Finally, note that, even if the key m is divulged, its knowledge does not suffice to find x from the knowledge of x': one needs absolutely to know the quotient q.

3 Fully homomorphic implementations

In what follows we present two implementations of the general framework of Section 2.2.

3.1 Integer encryption

Assume first that the data set X is a subset of the set $\mathbb{Z}$ of integers and a positive integer m be the secret key. Let $\mathbb{Z}_m$ be the ring of classes of integers modulo m, identified with its full set of representatives $\{0, 1, \ldots, m-1\}$, namely, the possible remainders of division of an integer by m. We define E to be the composition of the inclusion map of X into $\mathbb{Z}$ with the projection of $\mathbb{Z}$ onto $\mathbb{Z}_m$ thus, if $x \in X$, we let

$$E(x) = x'$$

be the remainder of the division of x by m, that is, x' is the unique integer such that $0 \leq x' \leq m-1$ and $x = x' + q \times m$, where q denotes the quotient of x by m. In particular, x' is equivalent to x modulo m.

The image $E(x)$, namely the remainder x', is then stored on the cloud. In order to be able to recover x later, the user also stores the quotient q locally. In most applications, however, q can instead be stored on the cloud. In this case, all the quotients q_i, $1 \leq i \leq n$, corresponding to all the elements x_i of X, are encrypted (each individually) using a symmetric-key encryption algorithm and a single secret key held by the user; see, for example, [10]. Note that the algorithm used for this purpose need not be homomorphic.

As proved in Section 2.2 above, E is a ring homomorphism. Therefore if x_i and x_j are to be added or multiplied, then their images ${x_i}'$ and ${x_j}'$ are operated upon. The user computes ${x_i}' + {x_j}'$ or ${x_i}' \times {x_j}'$ remotely. The same operations can be performed by anyone acting on behalf of the user. Note here that the mod m operation is not performed on the result of the sum or product. Both ${x_i}' + {x_j}'$ and ${x_i}' \times {x_j}'$ are stored on the cloud for later use.

3.1.1 Decryption of ${x_i}' + {x_j}'$ and ${x_i}' \times {x_j}'$

At a later time, the user may wish to recover $x_i + x_j$ or $x_i \times x_j$ from ${x_i}' + {x_j}'$ and ${x_i}' \times {x_j}'$, respectively. Using

$$x_i + x_j = ({x_i}' + q_i \times m) + ({x_j}' + q_j \times m) = ({x_i}' + {x_j}') + (q_i + q_j) \times m$$

and

$$x_i \times x_j = ({x_i}' + q_i \times m) \times ({x_j}' + q_j \times m) = ({x_i}' \times {x_j}') + ({x_i}' \times q_j + {x_j}' \times q_i + q_i \times q_j \times m) \times m$$

the results are obtained *locally*.

3.1.2 Space requirements

In what follows we do not concern ourselves with whether the input data, that is, the set $X = \{x_1, x_2, \ldots, x_n\}$ is or is not stored locally by the user at the user's facility. Instead, we focus on the storage needed by the encryption algorithm. If $q_1, q_2, \ldots, q_n$ are to be stored at the user's end, then the local space requirements are linear in n. Otherwise, the only space needed is to store the secret key m, and this is constant. On the cloud, space at least quadratic in n will be required if all pairs of elements in X will be involved in arithmetic operations, the results of which need to be stored. It could be as high as exponential in n if all subsets of X are operated on individually.

It is helpful to observe here that no encrypted value stored on the cloud (namely, ${x_i}'$, ${x_j}'$, ${x_i}' + {x_j}'$, ${x_i}' \times {x_j}'$, and so on) is ever larger than its unencrypted counterpart (namely, x_i, x_j, $x_i + x_j$, $x_i \times x_j$, and so on).

3.1.3 Time requirements

For a positive integer d, assuming that all integers in X are of $O(d)$ digits, each of the computations needed in this implementation, namely, computing ${x_i}'$, ${x_j}'$, ${x_i}' + {x_j}'$, ${x_i}' \times {x_j}'$, and so on, as well as decrypting each of these values, requires $O(d^2)$ time.

3.1.4 Security

The fully homomorphic cryptosystem for integers is secure as long as the numbers m and $\{q_1, q_2, \ldots, q_n\}$ are kept secret. This is true, for excluding the possibility of extraneous information being available (a threat to any and all cryptosystems), x_i cannot be derived from ${x_i}'$ without knowledge of q_i and m.

3.1.5 Example of fully homomorphic encryption for integers

Let $x_i = 17$, $x_j = 24$, and $m = 5$.

We have $x_i = 2 + (3 \times 5)$, $x_j = 4 + (4 \times 5)$, ${x_i}' = 2$, $q_i = 3$, ${x_j}' = 4$, and $q_j = 4$. Since ${x_i}' + {x_j}' = 6$ and ${x_i}' \times {x_j}' = 8$, we have $x_i + x_j = 6 + (3 + 4) \times 5 = 41$, which is indeed 17+24, and $x_i \times x_j = 8 + ((2 \times 4) + (4 \times 3) + (3 \times 4 \times 5)) \times 5 = 408$, which is indeed 17×24.

In closing, we note that while the cryptosystem of this section is intended for nonnegative integers, it can be easily adapted for negative integers (for example, if $-17 = 3 - (4 \times 5)$, we have that -17 modulo 5 is equivalent to 3, and therefore -17 is encrypted as 3), as well as for rational numbers (here a rational r is expressed as a reduced fraction $r = x/y$, and treated as two integers x and y, each of which is encrypted separately to yield x' and y', respectively).

3.2 Polynomial encryption

We now assume that the data set X is a subset of the set $\mathbb{R}[t]$ of polynomials in t with real coefficients. Fix a polynomial m considered as the secret key. Let $B = \mathbb{R}[t]/\mathbb{R}[t]m$, identified with the set of polynomials of degree less that the degree of m. We define E to be the composition of the inclusion map of X into $\mathbb{R}[t]$ with the projection of $\mathbb{R}[t]$ onto B, thus, if $x \in X$, we let

$$E(x) = x'$$

be the remainder of the division of x by m, that is, x' is the unique polynomial which is either zero or of degree less than the degree of m and $x = x' + q \times m$, where q denotes the quotient of x by m. In particular, x' is equivalent to x modulo m.

If we wish X to be a subset of the set $\mathbb{Z}[t]$ of polynomials in t with integral coefficients, we must assume m to have leading coefficient 1 (or -1), then the definition proceeds as before.

We note here that the analyses of space requirements, time requirements, and security presented in Sections 3.1.2, 3.1.3, and 3.1.4 in connection with the cryptosystem for integers, apply to the current cryptosystem as well.

3.2.1 Example of fully homomorphic encryption for polynomials

Let $A = \mathbb{Z}[t]$ be the ring of polynomials in t with integral coefficients. Given the polynomial $m = t-1$, then, for any polynomial x, there exists a unique polynomial x' such that $x = x' + q \times m$ with x' either zero or of degree less than the degree of x. Since the degree of $m = t - 1$ is one then x' is a constant, easily proven to equal the evaluation $x(1)$ of the polynomial x when $t = 1$.

4 Conclusion

Homomorphic cryptography is expected to make significant gains in importance as the volume of data stored by individuals, businesses, and organizations on untrusted databases increases dramatically over the next few years. Simultaneously, the search for a fully homomorphic encryption scheme that is conceptually simple, computationally efficient and cryptographically secure will continue.

In this paper we have presented a general approach for fully homomorphic encryption, as well as two specific implementations of homomorphic cryptosystems, one for integers and one for polynomials. We believe that our proposed systems enjoy the desirable properties of simplicity, efficiency and security, and represent a modest contribution to this growing and exciting field.

Acknowledgments

The first author wishes to acknowledge generous support from the Faculty of Arts and Science at Queen's University. The second author gratefully acknowledges partial support from the NSERC of Canada.

References

[1] Acar, A., Aksu, H., Uluagac, A.S., Conti, M., A survey on homomorphic encryption schemes: Theory and implementation, arXiv:1704.03578 [cs.CR]

[2] Brakerski, Z. Gentry, C., and Vaikuntanathan, V., (Leveled) fully homomorphic encryption without bootstrapping, *Innovations in Theoretical Computer Science Conference*, Cambridge, Massachusetts, 2012, pp. 309–325.

[3] Cao, Z. and Liu, L., On the weakness of fully homomorphic encryption, arXiv:1511.05341 [cs.CR]

[4] El Makkaoui, K., Ezzati, A., and Beni Hassane, A., Challenges of using homomorphic encryption to secure cloud computing, *Proceedings of the International Conference on Cloud Technologies and Applications*, 2015, pp. 1–7.

[5] Frederick, R., Core concept: Homomorphic encryption, *Proceedings of the National Academy of Science*, Vol. 112, No. 28, 2015, pp. 8515–8516.

[6] Gentry, C., A fully homomorphic encryption scheme, Ph.D. Thesis, Stanford University, September 2009.

[7] Gentry, C., Computing arbitrary functions of encrypted data, *Communications of the ACM*, Vol. 53, No. 3, 2010, pp. 97–105.

[8] Gentry, C.B., Fully homomorphic encryption, U.S. Patent No. 9,083,526 B2, July 14, 2015.

[9] Hartley, B. and Hawkes, T.O., *Rings, Modules and Linear Algebra*, Chapman and Hall, Cambridge (1970).

[10] Katz, J. and Lindell, Y., *Introduction to Modern Cryptography* (2nd edition), CRC Press, Boca Raton, 2015.

[11] Lauter, K. and Naehrig, M., Can homomorphic encryption be practical? *Proceedings of the Third ACM Cloud Computing Security Workshop*, New York, 2011, pp. 113–124.

[12] Martins, P., Sousa, L., and Mariano, A., A survey on fully homomorphic encryption: An engineering perspective, *ACM Computing Surveys*, Vol. 50, No. 6, January 2018, Article No. 83.

[13] Nguyen, P.Q., Breaking fully-homomorphic-encryption challenges, in: *CANS 2011*, Lin, D. et al. (Eds.), LNCS 7092, Springer, 2011, pp. 13–14.

[14] Van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V., Fully homomorphic encryption over the integers, in: *EUROCRYPT 2010*, Gilbert, H. (Ed.), LNCS 6110, Springer, 2010, pp. 24–43.

[15] Rivest, R.L., Adleman, L., and Dertouzos, M.L., On data banks and privacy homomorphisms, in: *Foundations of Secure Computation*, De Millo, R.A., et al. (Eds.), Academic Press, 1978, pp. 171–181.