Privacy-Preserving Data Analytics

David Pointcheval

DIENS, CNRS, Inria, PSL University, Paris, France

Abstract

Big data and data lakes are gold mines for data scientists with tons of applications to finance, medicine, economics, etc. But most of these data are quite sensitive and cannot be widely distributed or even just used without strong protection. One is thus facing the huge dilemma of having to make the choice between highly valuable analytics and privacy guarantees. More generally, individuals and companies are all outsourcing their data. They are confronted to the same issue.

This survey will present some recent tools cryptography has developed to address this dilemma: Fully Homomorphic Encryption, Functional Encryption, and Secure Multi-Party Computation allow to perform computations on data, without revealing them. Applications then cover outsourced computations, federated learning, private analytics, etc.

1 Introduction

1.1 The Context

For ease of use, many individuals migrated to the cloud to store and share their photos, or any type of personal documents. For a rationalization of the costs, companies also migrate their systems, with personal, financial and technical data, towards the cloud. These data, but also the identities of those who access them, as well as the queries asked, are sensitive information, which it would be good to protect, because no one had access to them when they were stored privately, on a local disk or a personal server. It is therefore of primary importance to protect not only the data, but also the accesses and the queries to the outsourced storage.

In the solutions proposed by the cloud providers, an access control makes it possible to identify the users, and a protected communication with TLS (Transport Layer Security) guarantees the confidentiality of the communications between the customer and the server. The provider ensures encrypted storage of the database, as well as strict application of access rights, once user authentication is done. Nevertheless, the provider sees everything in the clear, the data to be stored in the database, as well as the identity of the clients, their queries, and the answers. If there is no reason to doubt the good faith of the provider, no one is safe from errors or malicious behavior from internal or external adversaries. Users, both private individuals and professionals, cannot take such a risk on privacy for medical, economic, financial or technological data. This is even more critical as an information leak can occur and only be detected much later, once the damage is irreparable.

1.2 Overview

To ensure the same security properties as on local storage, the provider should not have any information about the stored data, the users who connect, the queries that are asked, and the computations that are performed. Of course, the confidentiality of the data is essential, and it is in fact the first property to be satisfied. It also seems to be the simplest one, since any encryption scheme can make it. On the other hand, we must not loose all the useful functionalities one can get on clear data, and in particular the queries, which can be simple keyword searches or more complex computations, and the sharing:

- We will first deal with the encryption schemes that allow computations without having to decrypt, with the famous Fully Homomorphic Encryption [Gen09], or FHE. This allows the outsourcing of any computations, but does not allow sharing, since only the owner of the initial data can recover the final result in clear;
- Broadcast encryption [FN94] allows confidential data to be broadcast to several people by specifying the target set during encryption. The recipients can be listed as either the authorized members or the excluded members. Unfortunately, this does not allow any computations;
- Functional encryption has recently been proposed [BSW11]. With a functional decryption key, a user can decrypt the result of the function evaluation, associated with the key, on clear messages, and only this result, without having any information about the individual messages.

For the anonymity of the users, the best solution is not to require any authentication. With a privacy-by-design, only the decryption keys (functional or classical) allow a user to access the clear data. Eventually, the confidentiality of queries or computations can be ensured by encrypting them (the queries or the programs to be performed), if the encryption mechanism allows computations on encrypted data. We are thus brought back to the above problem with FHE.

2 Homomorphic Encryption

Encryption schemes, whether symmetric (with a secret key to be able to encrypt) or asymmetric (with a public key), allow data to be stored, while keeping it hidden from the eyes. Unfortunately, classical encryption techniques exclude any manipulation of this data by the cloud provider. Indeed, an encryption scheme is said to be "secure" if it guarantees semantic security (difficulty in distinguishing between an encryption of m_0 from an encryption of m_1) for attackers who would have access to the decryption oracle [BDPR98] (as well as to the encryption oracle in the case of symmetric encryption [BDJR97]), with of course the constraint of not asking for the decryption of the challenge ciphertext. Intuitively, this means that it is not possible to extract any exploitable information about the plaintext when seeing the ciphertext. This also prevents any adversary to derive from a ciphertext another ciphertext whose plaintexts are related in a known way. These security notions are called indistinguishability and non-malleability, under chosen-ciphertext attacks (and possibly chosen-plaintext attacks). They are very strong and require complex mechanisms or paddings, excluding any operation on the ciphertexts.

However, algebraic properties exist for some encryption schemes taken in their basic form, which no longer satisfy these stronger security notions of semantic security and non-malleability. Nevertheless, they allow some operations on the messages, by manipulating only the ciphertexts. This is the case of the RSA [RSA78] or the ElGamal [ElG85] encryption schemes, where the product of two ciphertexts corresponds to the encryption of the product of the plaintexts:

$$\begin{aligned}
\text{RSA Encryption:} \qquad & & c = \mathcal{E}(m) &= m^e \bmod n \\
& & c_0 \times c_1 &= m_0^e \times m_1^e = (m_0 m_1)^e \bmod n \\
& & &= \mathcal{E}(m_0 \times m_1) \\
\text{ElGamal Encryption:} \qquad & & (c, c') = \mathcal{E}(M, r) &= (g^r, y^r \cdot M) \\
& & (c_0, c_0') \times (c_1, c_1') &= (g^{r_0}, y^{r_0} M_0) \times (g^{r_1}, y^{r_1} M_1) \\
& & &= (g^{r_0+r_1}, y^{r_0+r_1}(M_0 \times M_1)) \\
& & &= \mathcal{E}(M_0 \times M_1, r_0 + r_1).
\end{aligned}$$

The latter ElGamal encryption is, however, semantically secure with only the knowledge of the public key y (without access to any additional oracle), under the Decisional Diffie-Hellman [DH76] assumption. It can also be made additively homomorphic, by encrypting m as $M = g^m$, which allows the sum of the plaintexts to be obtained by a simple operation on the ciphertexts. This additive homomorphism property has been exploited in electronic voting systems, but can only be used with small scalars, as extracting m from $M = g^m$ requires a discrete logarithm computation, which is only possible from small messages. There are other additively homomorphic encryption scheme: Paillier [Pai99], that extends the Goldwasser-Micali encryption scheme [GM82] initially based on the quadratic residuosity, with modulo 2 operations, to the high residuosity, with modulo N operations, with an RSA modulus N hard to factor; Castagnos-Laguillaumie [CL15], based on a DDH-like assumption on class groups in imaginary quadratic fields.

2.1 More Operations on Ciphertexts

As just seen, there have been several additively homomorphic and multiplicatively homomorphic encryption schemes, for a long time. But none of them allows to apply the two internal laws of a ring, while remaining stable. Only the BGN (Boneh-Goh-Nissim) [BGN05] encryption scheme allowed additions, multiplication, and then additions again, but without allowing further multiplications, because the ciphertext is in another algebraic structure after multiplication. Similarly, the paper by Tomas Sander, Adam Young, and Moti Yung [SYY99] proposed a variant of the above Goldwasser-Micali, to allow multiplication, but doubling the size of the cipher at each multiplication. Thus, the number of multiplications was at most logarithmic to keep polynomial-size ciphertexts.

Nevertheless, such an additively homomorphic encryption scheme in $\mathbb{Z}_2$ (allowing XOR – Exclusive OR – between two bits) and multiplicatively homomorphic in $\mathbb{Z}_2$ (allowing AND between two bits) can evaluate any Boolean circuit on the ciphertexts. But the above schemes have constraints on the depth of the circuit: either very costly multiplications, or very limited.

In 2009, Craig Gentry [Gen09] presented the first construction guaranteeing stability, and thus an unlimited number of additions and multiplications: this allows the evaluation of any circuit on encrypted inputs, with the result encrypted under the same key. We name such a scheme a Fully Homomorphic Encryption (FHE) scheme.

2.2 Fully Homomorphic Encryption

The main idea is based on *bootstrapping*. But before describing this idea, let us consider the following simple secret-key encryption construction:

- the secret key, for both encryption and decryption, is an odd integer $p = 2m + 1$;
- the encryption of a bit b consists of $c = q \cdot p + 2 \cdot r + b$, where q is a random integer and r is chosen randomly in $\mathbb{Z}_m$;
- the decryption of c is done in two steps: first, thanks to p, we find the integer $B = 2 \cdot r + b$ by computing $c \bmod p$, while b is the parity bit of B.

We stress that $c \bmod p$ is indeed $B = 2 \cdot r + b$, as $r < m$ and so $B < p$, which is not impacted by the modular reduction. Let us now be given $c_0 = q_0 \cdot p + 2 \cdot r_0 + b_0$ and $c_1 = q_1 \cdot p + 2 \cdot r_1 + b_1$, encryptions of two bits b_0 and b_1:

$$c_0 + c_1 = (q_0 + q_1) \cdot p + 2 \cdot (r_0 + r_1) + (b_0 + b_1),$$

which is an encryption of $b_0 + b_1 \bmod 2$, with $q = q_0 + q_1$ and $r = r_0 + r_1 + (b_0 + b_1)/2$, as long as the latter remains lower than m. Similarly,

$$c_0 \cdot c_1 = (q_0 c_1 + q_1 c_0 - q_0 q_1 p) \cdot p + 2 \cdot (2 r_0 r_1 + r_1 b_1 + r_1 b_0) + b_0 b_1,$$

is therefore an encryption of $b_0 b_1 \bmod 2$, with $q = q_0 c_1 + q_1 c_0 - q_0 q_1 p$ and $r = 2 r_0 r_1 + r_1 b_1 + r_1 b_0$, provided that the latter remains less than m.

We call r the "noise" in the encryption of b, which masks a multiple $q \cdot p$ of p. Security relies on the difficulty of finding p, the approximate GCD of all noisy $q_i \cdot p$. If the noise is too small, this problem is easy, if the noise is too large (close to m), no homomorphic operation is possible. For well-chosen parameters, a compromise can be found: security is ensured, and a large number of homomorphic operations is possible. We will see below the necessary number of such homomorphic operations on the basic system, to obtain a stable fully homomorphic encryption scheme.

But first, with the secret key $\mathsf{sk} = p$, it is possible to publish a list $x_i = q_i \cdot p + 2 \cdot r_i$, for $i = 1, \ldots, t$, of 0-ciphertexts. Then, to encrypt a bit b, one can just compute $c = \sum_{i \in A} x_i + 2 \cdot r + b$, where A is a random subset of $i = 1, \ldots, t\}$ and r is an additional noise. With a good choice of parameters, we have a public-key encryption scheme, where $\mathsf{pk} = (x_i)_i$, which allows a limited number of additions and multiplications: on c_i ciphertexts of bits b_i, under the public key pk, it is possible to evaluate the circuit $C((b_i)_i)$ to obtain the result encrypted under pk. The public encryption key pk hides the secret decryption key $\mathsf{sk} = p$, under the approximate GCD problem.

A problem arises when the "noise" r becomes too important (greater than m) in a ciphertext c. Hence the idea of *bootstrapping* proposed by Craig Gentry [Gen09], on an encryption mechanism as above, but in ideal lattices, where the noise grows linearly in the number of additions and exponentially in the number of multiplications, as above: by publishing the encryption of the decryption key sk (as a bitstring, and thus with multiple ciphertexts, as above) under pk and using it to evaluate the circuit $D(\mathsf{sk}, c)$ that decrypts c under sk, but in an encrypted way, under pk: one obtains the result b, encrypted under pk. It is thus a new version of the ciphertext c, but with a lower noise. For this, it is sufficient that the decryption circuit D does not perform more homomorphic operations than the encryption scheme tolerates, before being too noisy. This has been shown to be possible, by choosing the parameters appropriately. Therefore, by performing a bootstrapping after each homomorphic operation (or after several), we end up with a ciphertext that contains an error similar to a "clean" ciphertext, hence the unlimited stability of the system: any circuit can then be evaluated.

However, without bootstrapping, depending on the chosen parameters, a certain number of operations are feasible before the noise destroys the information, we talk about "somewhat homomorphic encryption". This can be sufficient for circuits of low complexity.

2.3 Fully Homomorphic Encryption Constructions

Quickly after Gentry's first scheme, many improvements have been proposed, on the LWE (Learning with Errors) problem, or on integers based on the approximate GCD (as above). But the error kept the same type of growth: linear in the number of additions and exponential in the number of multiplications. Let us informally illustrate this second-generation family with the FV scheme [FV12]. It is based on the Ring-LWE assumption in the ring of polynomials $R =_q \mathbb{Z}_q[X]/(X^n + 1)$, for secret polynomials $\mathsf{e}, \mathsf{s} \in R_q$ with small coefficients and a public random polynomial $\mathsf{a} \in R_q$, the public key is $(\mathsf{a}, \mathsf{p} = \mathsf{a} \cdot \mathsf{s} + \mathsf{e})$.

To encrypt a message, encoded as a polynomial m with coefficients smaller than t (m can be seen in R_t), the ciphertext is $(\mathsf{c} = \mathsf{p} \cdot \mathsf{u} + \mathsf{e}_1 + \Delta \cdot \mathsf{m}, \mathsf{c}' = \mathsf{a} \cdot \mathsf{u} + \mathsf{e}_2)$, where $\Delta = \lfloor q/t \rfloor$, and $\mathsf{u}, \mathsf{e}_1, \mathsf{e}_2$ are random polynomials with small coefficients. As above with the approximate GCD illustration:

$$\mathsf{c} - \mathsf{c}' \cdot \mathsf{s} = \Delta \cdot \mathsf{m} + \mathsf{e} \cdot \mathsf{u} - \mathsf{e}_2 \cdot \mathsf{s} + \mathsf{e}_1 = \Delta \cdot \mathsf{m} + \mathsf{v}$$

with v the noise polynomial, that is small, with appropriate distributions for the "small" coefficients. This encryption scheme is clearly additively homomorphic, with additive noise. Additional information allows multiplicative property, where noise also grows multiplicatively.

In 2013, Craig Gentry, Amit Sahai, and Brent Waters [GSW13] proposed a completely new construction that exploits approximate eigenvectors: given a secret vector $\vec{s} \in \mathbb{Z}_q^n$, an encryption of $m \in \mathbb{Z}_q$ is a matrix $\mathbf{C}$ such that $\vec{s}$ is "almost" an eigenvector of $\mathbf{C}$, with m as an eigenvalue, so $\mathbf{C} \times \vec{s} = m \times \vec{s} + \vec{e}$: Given two ciphertexts $\mathbf{C}_0 \times \vec{s} = m_0 \times \vec{s} + \vec{e}_0$ and $\mathbf{C}_1 \times \vec{s} = m_1 \times \vec{s} + \vec{e}_1$, one can note that $(\mathbf{C}_0 + \mathbf{C}_1) \times \vec{s} = (m_0 + m_1) \times \vec{s} + (\vec{e}_0 + \vec{e}_1)$, as well as :

$$\mathbf{C}_0\mathbf{C}_1 \times \vec{s} = \mathbf{C}_0(m_1 \times \vec{s} + \vec{e}_1) = m_1\mathbf{C}_0 \times \vec{s} + \mathbf{C}_0 \times \vec{e}_1 = m_0 m_1 \times \vec{s} + m_1\vec{e}_0 + \mathbf{C}_0 \times \vec{e}_1.$$

In the case of addition, the noise grows linearly. In the case of multiplication, if we limit to messages in $\mathbb{Z}_2$, by controlling the norm of the matrices which constitute the ciphertexts, the noise can grow linearly too, but in an unbalanced way, becoming at most $\vec{e}_0 + \mathbf{C}_0 \times \vec{e}_1$. Thus, in the case of a "fresh" ciphertext with a low noise, it is better to put it on the right in the multiplication.

This construction can also be transposed to public keys, as well as to polynomial rings, as in [KGV14], allowing an efficient implementation. The combination of several encryption modes has thereafter led to very efficient constructions [CGGI16, CGGI17]: TFHE allows very fast bootstrapping.

2.4 Hybrid Method

There is still an issue with the amount of information to be transmitted, because all the data must be sent encrypted under this homomorphic encryption mechanism, in order to allow the evaluation of a function. And these ciphertexts under FHE have a huge expansion factor (several thousand times larger).

Thus, the idea of *bootstrapping* can be used again: the data is encrypted under a key k of a symmetric encryption scheme E, and only the secret key k is encrypted under the public key pk of the fully homomorphic encryption scheme. It is then possible to evaluate the circuit $D(k, c)$ which decrypts c under k, in order to obtain the plaintext b, encrypted under pk, that is to say under the FHE scheme. Then, no expansion factor appears here during the communication, since the symmetric encryption does not increase the size. We then talk about the hybrid method, which combines a symmetric encryption scheme and a public-key FHE scheme.

To allow good efficiency, for symmetric encryption, one can use block ciphers or stream ciphers well-suited for homomorphic encryption, with a low multiplicative depth [ARS+15, MJSC16].

2.5 Circuit Privacy

In addition to data confidentiality, the evaluated function may itself have an economic value and involve sensitive parameters. Thus, it may be important to hide all the intermediate steps performed by the homomorphic computation on the ciphertexts. While the final plaintext exclusively matches the result of the expected computation, and is independent of the intermediate steps, the ciphertext may contain additional randomness or noise that depend on these intermediate steps and may reveal critical information about the way the function has been evaluated. With an additional randomization step, it is possible to erase all information about the way the computation was performed [BdMW16].

3 Broadcast Encryption

Fully homomorphic encryption allows to externalize computations, but as any classical encryption scheme (whether with secret or public keys) targets a specific receiver: the one who has the decryption key. Of course, it is always possible to encrypt under multiple keys in order to target multiple recipients. But there remains the doubt that they do not all have the same message, and in any case, the size of the ciphertext is linear in the target set, which can be prohibitive in the case it is very large.

Broadcast encryption [FN94] makes it possible to target a set of individual recipients to efficiently share a message, and in particular with a ciphertext-size that is sub-linear in the size of the target set, or even constant.

3.1 Various Families of Broadcast Encryption

The aim of broadcast encryption is to send compact ciphertexts, regardless of the number of recipients. For this, two families of mechanisms have been proposed, depending on the number of recipients *vs.* the number of members registered in the system: either few people are concerned by the message, or few people are excluded. The main issue is to resist to a collusion of users: several members (explicitly or implicitly) excluded from the target set should not be able, by gathering their secrets, to obtain information about the broadcast message.

Of course, the most simple solution consists in encrypting a session key for each authorized member, and then the message under this session key. This solution is then linear in the target set. When almost all the users should receive the message, we instead use revocation schemes, as one would prefer the ciphertext size to depend on the number of revoked users. Combinatorial mechanisms try to cover the target set with multiple subsets, so that each authorized user is part of at least one subset and then, as above, the session key is encrypted under a key associated to each subset. The fewer subsets are required to cover the target, the more compact is the ciphertext. The most famous mechanism is NNL (Naor-Naor-Lotspiech) [NNL01] whose ciphertext size is linear in the number of revoked members. It relies on a binary tree structure to define the list of possible subsets.

Some algebraic approaches exploit mathematical properties that make the session key available to the authorized members only. The most famous solution uses polynomial interpolation in the exponents [NP01]: a master secret is shared among all the members, using the polynomial t-threshold secret sharing [Sha79]; this master secret is derived multiplicatively in the exponent to generate the session key, and in normal use $t-1$ derived shares are provided. Each user, with his own secret can generate the last required share to recover the session key. In order to revoke users (up to $t-1$), their derived shares are provided in the ciphertext, they thus cannot obtain t shares to reconstruct the session key. But then, ciphertexts are linear in the maximal number of revoked users.

Dan Boneh, Craig Gentry and Brent Waters [BGW05] proposed the first construction that provides a constant-size ciphertext, independently of the size of the target set, at the cost of linear public parameters in the number of members registered in the system and the use of pairings on elliptic curves.

3.2 Traitor Tracing

One of the key applications of broadcast encryption is Pay TV. Of course, the primary security goal is to prevent non-legitimate users from accessing the content, but registered users with access right must also be deterred from forwarding the content via another communication channel, in particular by giving their key material or by contributing to a pirate decoder with their secrets. This is the purpose of "traitor tracing": if a user with legitimate access, or even several users (called "traitors"), collude with their secrets to allow non-legitimate user to get access to the content of the encrypted stream (providing a "pirate decoder"), one wishes to be able to identify the traitors.

Any revocation mechanism as above allows some kind of tracing, as by revoking users, we can eventually exclude all the traitors to make the pirate decoder useless. However, it might require an exponential number of trials, the pirate decoder can stop working to avoid tracing, etc. One thus expects efficient and undetectable on-line tracing. Some relaxed definitions exist, such as blackbox tracing, where one has blackbox access to a resettable pirate decoder, and whitebax tracing, where one even has access to the code.

The first classical approach dates back from 1994 [CFN94], and uses codes with Identifiable Parent Property, later relaxed with binary collusion-secure codes [BS95]. Each user owns a list of secret keys according to a specific codeword. Any collusion of users, with multiple keys, will "virtually" derive multiple words, but the properties of theses codes allow to recover at least one parent of a derived word. The tracing procedure thus consists in learning a word associated to the pirate decoder, which can be extracted, bit-by-bit, by submitting specific ciphertexts. Tracing is thus linear in the length of the codewords (which is cubic in the number of users, or quadratic in the maximal size of the collusion [Tar03]).

4 Functional Encryption

On the one hand, fully homomorphic encryption allows to perform computations on the ciphertexts, but does not allow to control distribution of the computations nor the privacy of any data, since anyone able to decrypt the final result is able to decrypt the input ciphertexts. On the other hand, broadcast encryption allows to control the recipients, but does not allow computations.

Functional encryption [BSW11] provides the essential building block: it allows a fine-tuned control of the revealed information, restricted according to the key owned by the user and the constraints chosen by the data owner: security-by-design. Of course, the key may allow full decryption only under certain access conditions (identity-based encryption [BF01], attribute-based encryption [GPSW06], etc.), but may also restrict decryption to some aggregations or specific computations on the data.

Informally, functional decryption keys dk_f are generated, for specific functions f. Then on any ciphertext c of some data x, the key dk_f will provide $f(x)$ but no other information about x. Thus, the function f can test for specific recipient identity or attributes, before returning or not the clear x, which leads to a simple access control (IBE or ABE). But the function f can also do more complex computations, and in particular give access only to specific aggregations, depending on the recipients (the owned key).

4.1 Data Aggregation

The main property of functional encryption is exactly that it makes it possible to provide only partial information on the plaintext data, for example an average, an aggregation or any kind of statistics, without ever revealing additional information.

But contrary to FHE which returns the computation in an encrypted form and thus requires to possess the decryption key which allows not only to get the result in clear but also the initial data in clear, the functional decryption key performs the computation and provides the result in clear, from the encrypted inputs. And the key does not allow to decrypt the initial data.

We can therefore think to an encrypted database where each user has access to partial or aggregated information according to his functional decryption key. Although it has been shown possible (under very strong assumptions) to generate keys to evaluate any circuit on encrypted data [GGH$^+$13], the first effective construction has been provided for the family of inner products, or weighted means [ABDP15].

4.2 Inner-Product Functional Encryption

This is definitely a classical use-case: a database contains numerical values $v_{i,j}$, and the data owner wishes to limit only to some aggregations in the form of linear combinations, i.e. weighted means of each line i, $r_i = \sum_j a_j v_{i,j} = \vec{a} \cdot \vec{v}_i$, depending on the recipients. Thus, for each vector of weights $\vec{a}$, a functional decryption key can be generated, allowing the decryption providing the aggregation r_i on each row i, and nothing more on the values $(\vec{v}_i)_i$.

The intuition for a construction of a functional encryption scheme for the inner-product class of functions in $\mathbb{Z}_p$ is as follows: the secret key is a random vector $\vec{s}$ in $\mathbb{Z}_p^n$, while the functional decryption key associated to a vector $\vec{y}$ is $\mathsf{dk}_{\vec{y}} = \vec{s} \cdot \vec{y}$. To encrypt a vector $\vec{x}$ in $\mathbb{Z}_p^n$, one chooses a random scalar r in $\mathbb{Z}_p$ and defines the ciphertext as $C = (c_0 = r, \vec{c} = \vec{x} + r \cdot \vec{s})$. One can note that $\vec{c} \cdot \vec{y} = \vec{x} \cdot \vec{y} + r \cdot \vec{s} \cdot \vec{y} = \vec{x} \cdot \vec{y} + c_0 \cdot \mathsf{dk}_{\vec{y}}$. Hence, using the functional decryption key $\mathsf{dk}_{\vec{y}}$:

$$\vec{x} \cdot \vec{y} = \vec{c} \cdot \vec{y} - c_0 \cdot \mathsf{dk}_{\vec{y}}.$$

Of course, this one-time pad approach is not secure, as the same scalar r is used to mask multiple components. But this is the idea behind the construction in [ABDP15], that uses the Diffie-Hellman property and the randomness-reuse of the ElGamal encryption scheme [ElG84, Kur02]. More constructions have thereafter been proposed in [ALS16], still for the inner-product family from any additively homomorphic encryption scheme. It can then be applied with Paillier [Pai99], based on the integer factoring, and Regev [Reg05], based on lattice problems.

4.3 More Functions and Properties

As these constructions are based on additively homomorphic encryption schemes, they focus on linear computations. Whereas they are sufficient for a large class of statistics, and allow data classification with the SVM (Support Vector Machine) technique, such linear models are not very efficient, and switching to quadratic computations allows richer statistics, such as the variance, but also more efficient learning models, with non-linear activation functions in neural networks [RPB+19].

But one can all remark that most of these constructions explicitly need to include the function f in the functional decryption key for processing the decryption, whereas this function might be sensible. In order to keep it secret, one defines the notion of *function hiding*. Such a function-hiding functional encryption for inner product has been described in [BJK15].

5 Multi-Client Functional Encryption

Early constructions required that the entire database be generated by the same person at the same time. In particular, for the inner-product family, each vector must be encrypted at the same time by the same entity, as each coordinate exploits the same random randomness r. This limits the interest of aggregations, which could involve data from different sources or at various time slots. Functional encryption with multiple entries or multiple clients has been proposed [GGG+14], thus allowing entries encrypted independently, either by the same user or even by different clients.

Such a multi-client functional encryption for inner-product has been proposed in [CDG+18]: it allows various clients, that can even be competitors, to independently encrypt their own input $x_{t,i}$, for a specific time-period t, so that when, and only when, all the clients have contributed for that time-period t, the owner of the functional decryption key $\mathsf{dk}_{\vec{y}}$ for the vector $\vec{y}$ can obtain the result $\vec{x}_t \cdot \vec{y}$. Hence, multiple aggregations can be provided, without any interaction.

In [CDSG+20] we even pushed further the security-by-design, by suppressing any authority: the clients must all agree and contribute to generate a decryption functional key.

But again, efficient solutions only target inner-product families. More recent work addressed quadratic functions between two clients [dPP22] and the combination between ABE and Inner-Product families [NPP22], in order to address revocation for either specific users or specific functions.

Conclusion

Several techniques exist for protecting data privacy, with efficient and secure encryption schemes, and even post-quantum security. However, the current evolution with cloud-based storage, outsourced computation, and any on-demand online service requires more properties to remain usable.

We focused in this overview on non-interactive solutions, where the data owner can send or store the information in an encrypted way, and the recipient can operate on his own to access the final expected result, with many kinds of restrictions, by-design.

When interactions are possible, secure multi-party computation [Yao82] is also a generic approach that can always be exploited, but at a higher communication cost: all the parties can interactively compute $f(x_1, \dots, x_n)$ on their individual private inputs x_i, for any public function f. And there is the particular case where only two parties are involved, which is well-suited for machine learning applications and even federated learning [WTB+21, RTPB22].

Acknowledgements

I would like to acknowledge all my co-authors on functional encryption and multi-client functional encryption to this overview, as they contributed a lot to the state-of-the-art of computations on private data.

References

[ABDP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Heidelberg, March / April 2015.

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Heidelberg, August 2016.

[ARS$^+$15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 430–454. Springer, Heidelberg, April 2015.

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, October 1997.

[BdMW16] Florian Bourse, Rafaël del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 62–89. Springer, Heidelberg, August 2016.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 26–45. Springer, Heidelberg, August 1998.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 325–341. Springer, Heidelberg, February 2005.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 258–275. Springer, Heidelberg, August 2005.

[BJK15] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Heidelberg, November / December 2015.

[BS95] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data (extended abstract). In Don Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 452–465. Springer, Heidelberg, August 1995.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011.

[CDG$^+$18] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 703–732. Springer, Heidelberg, December 2018.

[CDSG$^+$20] Jérémy Chotard, Edouard Dufour-Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Dynamic decentralized functional encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 747–775. Springer, Heidelberg, August 2020.

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 257–270. Springer, Heidelberg, August 1994.

[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, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2016.

[CGGI17] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 377–408. Springer, Heidelberg, December 2017.

[CL15] Guilhem Castagnos and Fabien Laguillaumie. Linearly homomorphic encryption from DDH. In Kaisa Nyberg, editor, *CT-RSA 2015*, volume 9048 of *LNCS*, pages 487–505. Springer, Heidelberg, April 2015.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[dPP22] Paola de Perthuis and David Pointcheval. Two-client inner-product functional encryption, with an application to money-laundering detection. Cryptology ePrint Archive, Report 2022/441, 2022. https://eprint.iacr.org/2022/441.

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, Heidelberg, August 1984.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[FN94] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 480–491. Springer, Heidelberg, August 1994.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. https://eprint.iacr.org/2012/144.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[GGG$^+$14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Heidelberg, May 2014.

[GGH$^+$13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377. ACM Press, May 1982.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.

[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, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

[KGV14] Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: Scalable homomorphic implementation of encrypted data-classifiers. Cryptology ePrint Archive, Report 2014/838, 2014. https://eprint.iacr.org/2014/838.

[Kur02] Kaoru Kurosawa. Multi-recipient public-key encryption with shortened ciphertext. In David Naccache and Pascal Paillier, editors, *PKC 2002*, volume 2274 of *LNCS*, pages 48–63. Springer, Heidelberg, February 2002.

[MJSC16] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 311–343. Springer, Heidelberg, May 2016.

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 41–62. Springer, Heidelberg, August 2001.

[NP01] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In Yair Frankel, editor, *FC 2000*, volume 1962 of *LNCS*, pages 1–20. Springer, Heidelberg, February 2001.

[NPP22] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with fine-grained access control. Cryptology ePrint Archive, Report 2022/215, 2022. https://eprint.iacr.org/2022/215.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Heidelberg, May 1999.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[RPB$^+$19] Théo Ryffel, David Pointcheval, Francis R. Bach, Edouard Dufour-Sans, and Romain Gay. Partially encrypted deep learning using functional encryption. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *NeurIPS '19*, pages 4519–4530, 2019.

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

[RTPB22] Théo Ryffel, Pierre Tholoniat, David Pointcheval, and Francis R. Bach. Ariann: Low-interaction privacy-preserving deep learning via function secret sharing. *Proc. Priv. Enhancing Technol.*, 2022(1):291–316, 2022.

[Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.

[SYY99] Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for NC1. In *40th FOCS*, pages 554–567. IEEE Computer Society Press, October 1999.

[Tar03] Gábor Tardos. Optimal probabilistic fingerprint codes. In *35th ACM STOC*, pages 116–125. ACM Press, June 2003.

[WTB$^+$21] Sameer Wagh, Shruti Tople, Fabrice Benhamouda, Eyal Kushilevitz, Prateek Mittal, and Tal Rabin. Falcon: Honest-majority maliciously secure framework for private deep learning. *Proc. Priv. Enhancing Technol.*, 2021(1):188–208, 2021.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.