Journal of Computer Science and Cybernetics, V.37, N.4 (2021), 429–452
DOI 10.15625/1813-9663/37/4/16104

PRIVACY IN ADVANCED CRYPTOGRAPHIC PROTOCOLS: PROTOTYPICAL EXAMPLES

PHAN DUONG HIEU[1,*], MOTI YUNG[2]

[1]*LTCI, Telecom Paris, Institut Polytechnique de Paris, France*
[2]*Google LLC, New York, USA and Columbia University, New York, USA*

Abstract. Cryptography is a fundamental cornerstone of cybersecurity, traditionally supporting data confidentiality, integrity, and authenticity. However, when cryptographic protocols are deployed in emerging applications such as cloud services or big data, the demand for security grows beyond these requirements. Data nowadays are being extensively stored in the cloud, and users also need to trust the cloud servers/ authorities that run powerful applications. Collecting user data, combined with powerful tools (e.g., machine learning), can come with a huge risk of mass surveillance or of undesirable data-driven strategies for profit making, while ignoring users' needs. Privacy, therefore, becomes more and more important, and new techniques should be developed, first, to protect personal privacy, and, second, to reduce centralized trust in authorities or in technical solutions providers.

In a general sense, privacy is "the right to be left alone"; privacy protection allows individuals to have control over how their personal data is collected and used. Here, we discuss privacy protecting methods of various cryptographic protocols, in particular we review:

- Privacy in electronic voting systems. This is, perhaps, the most important real-world application where privacy plays a huge fundamental role.
- Private computation. This may be the widest domain in the new era of modern technologies with cloud computing and big data, where users delegate the storage of their data and its computation to the cloud. In such a situation, "how can we preserve privacy?" is one of the most important questions in cryptography nowadays.
- Privacy in contact tracing. This is a typical example of a concrete study of a contemporary scenario where one should deal with the unexpected crucial societal problem but needs not pay the cost of weakening privacy of users.

Finally, we exemplify emerging notions, and in particular one aimed at reinforcing privacy by masking the type of executed protocol; we call it *covert cryptographic primitives and protocols.*

Keywords. Cryptography, cryptographic protocols, privacy, anonymity, decentralization.

Dedicated to Professor Phan Dinh Dieu on the occasion of his 85th birth anniversary.

*Corresponding author.
E-mail addresses: hieu.phan@telecom-paris.fr (P.D. Hieu), motiyung@google.com (M. Yung).

1. INTRODUCTION

The last few decades witnessed an extremely rapid developments of new technologies: cloud services are well settled to help users to store a huge volume of data and to implement powerful algorithms that cannot be run on a single machine; new methods in machine learning geared toward analyzing big data have been introduced, etc. All of these developments give new and comfortable utilities for people, but they also contain new risks in the long run. Namely, more and more personal information is being collected, analyzed, and the privacy of people may be seriously violated. This puts an urgent demand from technologists: to invent new methods protecting privacy in all the relevant technological developments.

The main source of problems for privacy is that almost all applications of cloud computing and on big data are running on servers owned by governments or cooperates. When data and computation are centralized, the users have no choice others than to fully trust these entities. The risks can then be characterized in two categories: i) the privacy is broken because the central servers are attacked; ii) the personal data are leaked for profit of these entities holding the data. We can consider several examples:

- In the first category, as applications are now developed on a very large scale, different public databases can be correlated. Here are some examples. In medical management, the genome database can be correlated with a public voting dataset to re-identify data [1]; in electronic identity management, Taiwan's ID card is attacked employing mathematics: with a quick GCD algorithm extracting hundreds of personal secret keys [2]; camera's recording database is leaked in the US, etc.
- In the second category, the risk for privacy is more dangerous because the leakage is of huge quantities, both on corporates and governments' sides. We can see it from the Cambridge Analytica data incident, where more than 80 million users' data on their Facebook accounts have been leaked to the British consulting firm Cambridge Analytica without the users' consent, which was then to be used for political advertising [3]. On the government's side, China's social credit system tracks and then evaluates citizens. This system relies on many advanced technologies such as facial recognition systems, big data analysis, AI, etc.

Facing these very high risks for privacy in the usage of modern technologies, regulations have been studied and proposed to limit the exploit of users' personal data. On the technical side (which falls into our interests), the studies turn out to consider advanced methods to preserve the privacy of the users. The main question is, therefore:

Can we avoid the need for the total trust in central authorities so that each user can fully control the use of its personal data?

This question can be answered positively, theoretically speaking: any problem involving many parties can be implemented in a private and decentralized way so that all parties agree on the output functions and the protocol guarantees that nothing else will be leaked. Such a protocol is called multi-party computation, or MPC for short. While the general solution exists for any kind of function and the privacy of each party can be guaranteed even if all the other parties are corrupted (or minority of all other parties if we insist on robustness), it is totally impractical since it involves heavy machineries such as garbled circuits, oblivious transfer, and employs many rounds of interaction between parties.

The existence of the general solution (i.e., feasibility), though in theory, gives us the hope that one can eventually implement any application in a way that preserves privacy for all the parties. This can be the change in the way we explore data in the future. If we look back the history, the invention of public-key cryptography (encryption and signature) did change the way we exchange information: no one in the 70s could imagine that one can securely exchange secret information (credit card number, instant message, payment of tax, etc) with a party without the need to agree on a shared secret key before that. Now, analogously, the new techniques has the potential to help us to do everything without the need for trust in central authorities.

From theory to practice there is not a short way: While a general solution exists for any privacy-preserving multi-party function, the question is now to consider case-by-case problems in practice, and propose efficient solutions that fit our current computational resources of time and memory. This becomes the central question in cryptographic applications. As the full domain is very large, in this survey, we will try to explain how one can get privacy in some important prototypical protocols with relevant real-life applications.

Security Proofs. Since the birth of modern cryptography, where Diffie and Hellman introduced the notion of asymmetric cryptography (or public-key cryptography) in 1976, many primitives were introduced. Many schemes have been proposed and a number of those schemes have been broken. Provable security has been therefore considered as a very important line of research which gives a "validation" for the proposed schemes. Namely: What the validation achieves is a statement like "If one can break the cryptographic protocols, one can efficiently solve the underlying problem which is assumed to be hard to solve." The research on the hardness of underlying problems in cryptography such as factorization, discrete logarithm, shortest vector in a lattice, syndrome decoding, etc., in turn, led to a new mathematical area of computational number theory with concentration on hard to solve problems. Also, the goal of proving the security of schemes requires the need for formalization of what is called security. This latter subject is very much influenced by the spirit of the *Imitation game* (or Turing's test) in the introduction of artificial intelligence: if we interact with a robot and a human and we cannot distinguish them, then the robot effectively acts as a human, and thus is claimed to have achieved the highest level of (artificial) intelligence. As was presented by Avi Wigderson in his first public lecture [4] after receiving Abel's prize award in 2021 (with Lászlό Lovász), almost all security notions in cryptography can be represented as an analog of this imitation game, let us consider some examples:

- Security of the encryption of Goldwasser and Micali [5]: an adversary cannot distinguish between encryption of two different messages of its choice. Because public key encryption for any message can be performed by anyone, including the adversary, this notion requires that any good encryption must be probabilistic and one can only deal with computational adversaries whose resource of computation and memory are polynomially bounded (because an unlimited adversary can run an exhaustive search to break any public-key encryption). Since then, the model of probabilistic computational distinguishers is extensively used in cryptography.
- Security in general multi-party computation: as the situation becomes much more complicated than in the standard encryption, one should formalize the security in a more complex manner. However, the spirit of an imitation game predominates again: one requires that any adversary cannot distinguish the difference between a real game

with the system and an ideal game where essentially only the outputs are given and no interaction between the parties (the interactions in the game can be simulated efficiently given the ideal game and, in fact, do not contribute any added effective knowledge about the ideal game to the adversary). In such a case, the adversary gains no information more than it can gets trivially and it models what we can hope for security.

In this survey, due to lack of space, we will not discuss in details security models but would suggest that actually, it is very important in cryptography to formalize the security notions we aim at in order to understand the hardness of algorithmic problem we want to base a new scheme upon, and finally to give a rigorous security proof for any proposed scheme by reducing the formalized security game to breaking the underlying algorithmic problem.

In the following sections, we will discuss privacy on both theoretical and practical sides, ranging from voting systems, contact tracing, to theoretical tools like functional encryption and homomorphic encryption. All aimed at preserving privacy in decentralized systems where we can reduce requirements of trust in the authorities or cloud servers to a simpler situation where a participants needs merely to trust its cryptographic systems and actions. We will try to give a survey on the techniques developed in these topics, as well as some of our own related research.

2. PRIVACY IN ELECTRONIC VOTING SYSTEMS

Voting systems are necessary ingredients for a functioning democracy. In a voting system, two important features are difficult to achieve simultaneously: the privacy of the voters and the (public or universal) verifiability. Traditionally, for public verification of the result, each ballot is associated with a voter but this breaks the privacy; when it is desired to preserve privacy, the voters vote secretly in a polling station where all ballots will be shuffled, but then the public verification of the result cannot be carried out. The implementation of an electronic voting system in practice requires very careful analysis. A complete voting system involves many other ingredients such as how to implement distributed decryption, how one can prove that a vote is correctly produced, how to avoid coercion, etc. In this section, we focus on the technical side and explain how one can achieve both privacy and public verification, simultaneously.

2.1. Technique from Homomorphic encryption

A very interesting principle to achieving privacy is the following: Individual votes are never decrypted, therefore there is no possibility of linking voters to their votes. This method is implemented in the Helios voting system, a system widely used in practice. The core idea is to combine (add) all individual votes into one final tabulation tally and jointly decrypt this accumulated value later on. Because votes can be posted to a public bulletin board, anyone can verify the outcome of the election (as operation over ciphertexts is public and efficient operation). We will briefly explain how all individual votes can be combined, via the use of additive homomorphic encryption.

Exponential ElGamal. The ElGamal encryption scheme [6] was invented in 1985 (its semantic security was proven in [7]). Here we present a version where messages are encrypted in the exponential to be able to use in the voting systems. Another well-known additive encryption used in voting is the Paillier encryption [8].

ElGamal uses a prime-order group where one typically picks a prime p such that $q = \frac{p-1}{2}$ is also prime. The group $\mathbb{Z}_p^\star$ with multiplication modulo p has $p-1$ elements and because $p-1 = 2q$ where q is also prime, we pick an element $g \in \mathbb{Z}_p^\star$ of order q then the subgroup $G :=< g >\subset \mathbb{Z}_p^\star$ is itself a cyclic group of order q. As G is a cyclic group of prime order, it has no subgroups (apart from the identity) and normally it limits the attacks to exploit the extra structures of the group. It is believed and assumed that the discrete logarithm (given $g, y = g^x$ for a random $x \in \mathbb{Z}_q$, compute x) and the Diffie-Hellman problems (given $g, X = g^x, Y = g^y$ for a random $x, y \in \mathbb{Z}_q$, compute g^{xy}) in such a group are hard. The exponential ElGamal can be described as follows:

- $\mathsf{KeyGen}(\lambda)$: Pick sk at random from $\mathbb{Z}_q$ and set $pk = g^{sk}(\mod p)$, return (pk, sk).
- $\mathsf{Encrypt}(pk, m)$: Pick r at random from $\mathbb{Z}_q$ and set $c = g^r(\mod p), d = g^m \times pk^r(\mod p)$. Return (c, d).
- $\mathsf{Decrypt}(sk, (c, d))$. Compute $g^m = d/c^{sk}(\mod p)$, then recover m.

Figure 1: Exponential ElGamal encryption

The last step in the decryption looks a bit strange as the decryptor has to solve the discrete logarithm in order to get the message. The point is, however, that this is feasible as far as m is small (if m is less than, say 2^{40}, a thousand of billion, then there is no problem to recover it from g^m), and this is, in fact, the case of election. For example, in a voting system involving two candidates, we can attribute the vote for the first candidate as 0 and the vote for the second candidate as 1. In the end, if there are N voters and the sum of all the votes is k then if $k \leq \frac{N}{2}$ then the first candidate wins and if $k > \frac{N}{2}$ then the second candidate wins. The exponential ElGamal encryption exactly allows us to compute the sum of all the votes without decryption of each vote. In fact, if $(c = g^r(\mod p), d = g^m \times pk^r(\mod p))$ is the encryption of m and $(c' = g^{r'}(\mod p), d = g^{m'} \times pk^{r'}(\mod p))$ is the encryption of m' then the coordinate by coordinate multiplication of them, namely $(cc'(\mod p), dd'(\mod p))$, is the encryption of $m + m'$. Therefore, one can multiply all the ciphertexts and then get the encryption of the sum of all the votes. By a distributed decryption of this final result, one can get the result without decrypting any individual vote. Since only the encryption of the sum is made available to the decryptor, this means that individual votes remain private.

The area of additive homomorphic encryption based voting protocols has produced numerous results over the years. A robust and verifiable early proposal has been given in [9].

The idea evolved further to distribute the central tallying authority (one of the first proposals to use distributed cryptography, which we will cover more examples of later on) [10], with more works in [11, 12, 13].

2.2. Technique from Mix-nets

Another and more general method that works with any kind of votes is to randomize all the votes (re-represent the encryption in a random way, which is possible in homomorphic encryption schemes by just adding a probabilistic encryption of zero to the ciphertext)

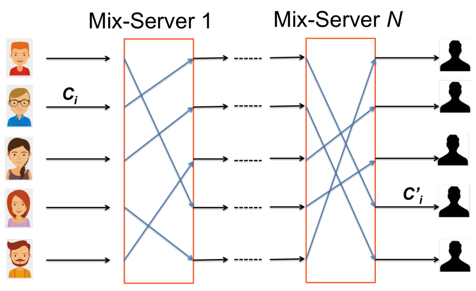

Figure 2: Shuffle of ciphertexts via a Mix-Net system

and shuffle them all. Such a method is called mix-net which preserves both privacy and accountability with public verification. At first glance, this principle seems to be similar to traditional paper-based voting: each person puts the vote into a ballot box, then the votes will be shuffled before tallying. However, in a paper-based voting when ballots are shuffled, if the input and output ballots are unlikable by any information, then there is no way to publicly verify if a ballot is counted and one has to trust the board of trustees (talliers). In cryptographic protocols, on the other hand, one can prove that the output ballots are exactly a permutation of the input ballots which were rerandomized and this verification can be publicly done, thus one can also get public verification along with privacy. We now explain mix-nets and the methods to efficiently achieve it.

Mix-Nets A shuffle of ciphertexts outputs a set of ciphertexts of the same plaintexts but in a randomly permuted order such that it is not possible to link the senders after decryption. If several mix-servers perform a shuffle successively, then one honest mix-server suffices to mask the order of the ciphertexts even if all the other ones are dishonest or corrupted. Therefore, increasing the number of mix-servers leads to a more secure protocol but also increases its cost. Chaum [14] introduced the notion of a mix-net system (as illustrated in Figure 2) which consists of a succession of shuffles and this notion has been used with many applications to anonymous emails, anonymous routing, and especially e-voting.

When it comes to electronic voting schemes, random shuffling of authenticated ciphertexts with mix-nets requires complex zero-knowledge proofs to guarantee the actual permutation of the initial ciphertexts in a privacy-preserving way. This is the bottleneck in mix-net since each mix-server has to prove that the output ciphertexts come from a permutation of the input ciphertexts of the same plaintexts without revealing any information about the underlying plaintexts. While we know how to construct a zero-knowledge proof for any NP-relation [15], such a zero-knowledge proof is very expensive as it acts on the whole set of ciphertexts. Some schemes came up with methods to make mixing robust for voting applications, see [16].

In the main two techniques we would like to point out, Furukawa and Sako [17] make proofs of permutation matrices and Neff [18] considers polynomials which remain identical with a permutation of the roots. While the latter approach produces the most efficient schemes, they need to be interactive. Groth and Ishai [19] exploited this interactive approach and proposed the first zero-knowledge argument for the correctness of a shuffle with sub-linear communication complexity, but computational complexity is super-linear which was then improved by Bayer and Groth [20].

It is interesting to notice that such a zero-knowledge proof was used in a voting system of French national education, which involves more than 430,000 votes and even though it contains only one mixer server, the running time is very long and presented a bottleneck in the verification. It was then a derived question how to improve the efficiency of such a proof so that it can work in a larger scale voting with many mix-server. Very recently, in [21], it was shown that one can use a signature with an advanced feature, namely linearly-homomorphic signatures, to design a new approach for proving correct shuffling of signed ElGamal ciphertexts: the mix-servers can simply randomize individual ballots, which means the ciphertexts, the signatures, and the verification keys, with additional global proof of constant size, and the output will be publicly verifiable. This leads to a new highly scalable technique.

We can illustrate the new approach in [21] as follows: in the shuffle, each ciphertext C_i (encrypted vote in the ballot, in the context of electronic voting) is signed by its sender and the mix-server randomizes the ciphertexts $\{C_i\}$ and permutes them into the set $\{C'_i\}$ in a provable way. The goal of the proof is to show the existence of a permutation Π from $\{C_i\}$ to $\{C'_i\}$ such that for every i, $C'_{\Pi(i)}$ is a randomization of C_i. Then, the output ciphertexts can be mixed again by another mix-server.

The main step of such proof relies on linearly-homomorphic signatures. The notion of homomorphic signatures dates back to [22], with notions in [23], but the linearly-homomorphic signatures, that allow to sign vector sub-spaces, were introduced in [24], with several follow-ups by Boneh and Freeman [25, 26] and formal security definitions in [27] (see also [28]). By using signatures that are malleable and that allow signing any linear combination of the already signed vectors [24], the approach in [21] avoids the proof of an explicit permutation Π on all the ciphertexts (per mixing step) but still guarantees the appropriate properties deeply using the linearly-homomorphic signature schemes:

- Each user is associated to a signing/verification key-pair for a linearly-homomorphic signature scheme [24], and uses it to sign his ciphertext and a way to randomize it. This guarantees that the mix-server will only be able to generate new signatures on randomized ciphertexts, which are unlinkable to the original ciphertexts, due to the new random coins. However, unchanged verification keys would still allow linkability;
- Each verification key of the users is thus also certified with a linearly-homomorphic signature scheme, that allows randomization too as well as adaptation of the above signature on the ciphertext, and provides unlinkability.

Unforgeability of the signature schemes will essentially provide the soundness of the proof of correct mixing: only permutations of ballots are possible. With the above linear homomorphisms of the signatures, we can indeed guarantee that the output C'_j is the randomization of an input C_i, and the verification keys are unlinkable.

We note that an approach which combines mix networks and homomorphic encryption has been proposed in [29]. Another approach which base voting protocols with privacy on neither homomorphic encryption nor mixnetworks, is proposing the use of blind signature (another privacy primitive), and it was proposed in [30].

Also, it is worth noting that more recent protocols deal with augmenting paper ballot typical in national elections, but adding verifiability while keeping ballot privacy and auditability, or transforming the election to be handled by remote voters yet keeping the important properties in tact, see [31, 32, 33]

We conclude this section by explicitly specifying that, in order to preserve privacy in voting systems, a number of advanced cryptographic primitives are employed, especially zero-knowledge proofs, randomizable encryption, and linearly-homomorphic signatures. Standard primitives like encryption and signature with advanced features such as randomizable and homomorphic properties also play important roles in different contexts that we will consider in the next sections.

3. PRIVATE COMPUTATION

New technologies with cloud computing, big data processing, machine learning, etc. have changed the ways data is collected and exploited. While these new platforms allow the development of very useful and innovative applications, the risk for privacy is important. In many cases, this risk for a user's privacy comes not only from himself or herself but comes from other people as well: because one is connected to many others and also stores personal information of the others (such as addresses, telephone numbers in the contact list) when one accepts to give this information for a service, it also gives information about the other people. In order to ensure privacy, it would be necessary that all the information should be encrypted and stored in a decentralized way so that no single entity (or collusion of different entities) can manipulate the data in clear.

Traditionally, private computation has had a number of directions in modern cryptography. In 1978 the first proposal for secure computing based on the idea of Homomorphic Encryption emerged in [34].

It presented computing on ciphertexts directly, without the need to decrypt. Fully homomorphic is one which allows a general computation (i.e., to operate a Turing complete operations, like Boolean NOR or Boolean NAND gates or addition and multiplication in algebraic circuits. The original work did not present any implementations of Fully Homomorphic Encryption. Then it took more than 20 years to get Somewhat Fully Homomorphic Encryption, limited to logarithmic depth circuit [35], and finally after 10 more years, the full breakthrough solution by Gentry was found [36].

Let us note that also around 1978, the era of "multiparty computation protocols" has started as a direction as well (with a protocol on various specific tasks like Mental Poker, and then Oblivious Transfer, or Coin Tossing). The attempt in thius area is to use cryptography (or information theoretic splitting of values) to allow parties to compute while hiding the inputs. Namely, rather than cryptography used for concealing communications which has been its traditional task, cryptography is used, instead, to compute while concealing partial information about private inputs. The area generated very general results, so that any randomized function which is efficient, can also be efficiently computed with input security (general secure two- or multi- party computations). The initial results showed the feasibility of the area by Yao [37] for two parties and by Goldreich, Micali, and Wigderson [15] for the multi party case. This area is currently being proposed for practical implementations, designing and implementing protocols with improved efficiency.

Currently, perhaps the two most active research directions carrying on computing over encrypted data (over ciphertexts that is) are inspired by the seminal papers on Fully Homomorphic Encryption (FHE) [36] and by the notion of Functional Encryption (FE) [38, 39, 40]. The final objective in these directions is still very far from realizable in practice but the advancements also advance very fast.

Beside the new primitives allowing computation over encrypted data, *decentralized cryptography* is also one of the main directions of current research in cryptography, especially in a concurrent environment of multi-user applications, where there is no way to trust any authority. Recently, the rise of blockchain applications also raised the feasibility and importance of decentralized applications. However, blockchain mainly addresses the decentralized validation of transactions, but it does not help in decentralizing computations per se. For the computational purpose, though general solutions can be achieved via multi-party computation, reasonably efficient solutions only exist for a limited number of protocols, as decentralization usually adds constraints to the design of protocols: in broadcast encryption [41], the decentralized protocol in [42] is much less efficient than the underlying original protocol [43]; in attribute-based encryption [44], the decentralized scheme [45] implies some constraints on the access control policy, that are removed in [46], but at the cost of the use of bilinear groups of composite order with 3 prime factors; etc. One additional area that is active and decentralised by nature is exploiting the decentralization to increase trust in cryptographic systems themselves, where a cryptographic key is shared among trustees and a majority of them have to act in order to generate the cryptographic value (decryption or a signature value). The area, called “Threshold Cryptography” was defined and implemented in full generality in [47].

In short, for the general purpose, although positive results for all of the above objectives were well settled in theory research (feasibility), no practical solution exists. However, for specific purposes, in turn, practical solutions exist and have been implemented.

We will present in the rest of this section some recent advancements in these areas of FHE, and FE, and their decentralized variants.

3.1. Fully homomorphic encryption and decentralized computing over encrypted data

In the era of cloud computing, personal and business data are stored and computed on by third parties such as Google, Apple, Facebook, Amazon, Microsoft, Dropbox, Twitter, etc. If users want to outsource computation to third parties without losing the privacy and confidentiality of data, then they should encrypt the data. If a classical encryption is used then the encrypted data are essentially just random strings to third parties and there is no way they can compute anything meaningful on the underlying data. Fully homomorphic encryption (FHE) was invented to be used to store data remotely in encrypted form, but still have the remote cloud providers able to evaluate known functions on the data without ever learning anything about either the inputs or the outputs. As all computations can be expressed as an algebraic circuit over multiplication and addition gates applied to input values, if we have an encryption that acts multiplicatively and additively which is a complete universal base for any computation (thus called fully homomorphic encryption) then we can encrypt the input data, do the desired computation on the encrypted data, and will get the encryption of the output function and finally only the users can decrypt it. The problem is natural as it was identified already in 1978 as mentioned above, but the problem evaded any solution for more than 30 years. Many people suspected that there is something inherently incompatible between homomorphic property for a complete base of operations and the security of an encryption. Therefore, Gentry’s construction of FHE in 2009 [36], in fact, shook the world of cryptography, and inspire a flurry of research results making FHE more efficient. Although there are many advancements and implementations, there is still much work to be done in order to increase the hope making FHE practical one day.

While the general solution for FHE is not yet practical, it is an interesting direction to consider real-life applications that only require the evaluation of restricted classes of functions. Boneh-Goh-Nissim [48] proposed a nice solution enabling the evaluation of quadratic polynomials (requiring only a single multiplication) with two main applications: private information retrieval schemes (PIR) and electronic voting protocols. Restricted class of FHE can also be used in machine learning. These applications are particularly useful in a decentralized setting in practice, as they deal with sensitive database from different users.

FHE was initially defined for a single data owner, and was later extended to multiple users under the name Multi-Key FHE [49]. Decentralization of FHE with multi-input from different users remains a challenge. The decentralization of BGN scheme seems hard as their solution relies on a composite-order elliptic curve and thus on the hardness of the integer factoring which makes the decentralization difficult. In fact, no efficient multi-party generation of distributed RSA modulus is presently known, only two-party ones. Even the currently most efficient construction [50] is not efficient enough as it relies on oblivious transfer in the semi-honest setting, coin-tossing, zero-knowledge and secure two-party computation protocols in the malicious setting. Fortunately, Freeman [51] proposed a conversion from composite-order groups to prime-order groups for the purpose of improving efficiency. In [52], it was showed that Freeman's conversion could be decentralized and thus 2-DNF functions can be efficiently decentralized. It remains an open problem to construct an efficiently distributed evaluation protocol for k-DNF formulas. A solution to this problem will be very useful in opening the way for a rich class of operations over encrypted database. At the first glance, it is tempting to believe that a k-somewhat homomorphic encryption scheme can be transformed into a decentralized scheme just by adding a secret sharing on top of the key generation. However, current secret sharing methods require a dealer during the setup, which is not compatible with decentralized settings.

Freeman's framework We explain the main idea in designing decentralized BGN scheme. To evaluate 2-DNF formulae on encrypted data, Boneh-Goh-Nissim described a cryptosystem [48] that supports additions, one multiplication layer, and additions. They used a bilinear map on a composite-order group and the secret key is the factorization of the order of the group. Unfortunately, composite-order groups require huge orders, since the factorization must be difficult, with costly pairing evaluations. In order to improve on the efficiency, Freeman in [51, Section 5] proposed a system on prime-order groups, using a similar property of noise that can be removed, with the general definition of subgroup decision problem. Let us recall the Freeman's cryptosystem in Figure 3.

The Freeman's scheme is also additively homomorphic. Moroever, if an homomorphism π_T exists such that, for all $g \in G, h \in H$, $e(\pi_1(g), \pi_2(h)) = \pi_T(e(g,h))$, we can get, as above, a ciphertext in G_T of the product of the two plaintexts, when multiplying the ciphertexts in G and H. The new encryption scheme in G_T is still additively homomorphic, and allows evaluations of 2-DNF formulae.

Decentralization. While Freeman [51] proposed a conversion from composite-order groups to prime-order groups for the purpose of improving the efficiency, it is interesting that the conversion allows multi-user setting, since a common setup can handle several keys. In [52]

$\mathsf{KeyGen}(\lambda)$**:** Given a security parameter λ, it generates a description of three Abelian groups G, H, G_T and a pairing $e : G \times H \to G_T$. It also generates a description of two subgroups $G_1 \subset G, H_1 \subset H$ and two homomorphisms π_1, π_2 such that G_1, H_1 are contained in the kernels of π_1, π_2 respectively. It picks $g \overset{\$}{\leftarrow} G$ and $h \overset{\$}{\leftarrow} H$, and outputs the public key $\mathsf{pk} = (G, H, g, h, G_1, H_1)$ and the private key $\mathsf{sk} = (\pi_1, \pi_2)$.

$\mathsf{Encrypt}(\mathsf{pk}, m)$**:** To encrypt a message m using public key pk, one picks $g_1 \overset{\$}{\leftarrow} G_1$ and $h_1 \overset{\$}{\leftarrow} H_1$, and outputs the ciphertext $(C_A, C_B) = (g^m \cdot g_1, h^m \cdot h_1) \in G \times H$.

$\mathsf{Decrypt}(\mathsf{sk}, C)$**:** Given $C = (C_A, C_B)$, output $m \leftarrow \log_{\pi_1(g)}(\pi_1(C_A))$ (which should be the same as $\log_{\pi_2(h)}(\pi_2(C_B))$).

Figure 3: BGN scheme converted in prime-order group by Freeman's framework

Hebant *et al.* additionally showed it is well-suited for distributed evaluation of 2-DNF formulae. Actually, working in prime-order groups, they can avoid the bottleneck of a distributed generation of RSA moduli. However, it is not enough to have an efficient distributed setup. One also needs to distribute any use of the private keys in the construction: for decryption and re-encryption. Unfortunately, the Freeman's generic description with projection matrices does not directly allow the design of a decentralized scheme, *i.e.*, with efficient distributed decryption without any trusted dealer. Hebant *et. al.* thus specify particular projections, with well-chosen private and public keys. This leads to an efficient decentralized version with distributed private computations. Decentralizing evaluation of 2-DNF formulae already gives interesting applications, including a simple machine learning's classification and group testing, as specified in [52]. It remains an open question to efficiently decentralize FHE schemes for more general function than 2-DNF formulae.

3.2. Functional encryption: Moving from revealing all-or-nothing to partial information

Public Key Encryption (PKE) enables people to securely communicate and share sensitive data with others over public channels. It allows recipients to recover in an all-or-nothing fashion encrypted data (once the recipients have the secret key then they will recover the original data, otherwise the recipients have no information about the plaintext data). Functional Encryption (FE) [38, 44], proposed by Boneh, Sahai and Waters, overcomes this all-or-nothing limitation of PKE by allowing recipients to recover encrypted data in a more fine-grained manner. Instead of revealing all-or-nothing of the original encrypted data as in PKE, recipients can get the evaluation of (statistical) functions on the data. As the function can contain an access control component that checks some relation between the identity in the functional decryption key and the authorized identity in the plaintext, this primitive generalizes Identity Based Encryption (IBE) and Attribute-Based Encryption (ABE). By allowing computation of partial data, one can aim at getting, both, the utility of analysis on large data while preserving personal information private. Functional Encryption got large interest from the cryptography community. However, there is still no efficient construction of functional encryption for general functions, and efficient constructions works mainly for linear and quadratic functions only [53, 54, 55].

In many practical applications, it is common that people only care about several specific functions on the data, for example, the mean value of the data. Linear functions are thus already interesting. We will explain in Figure 4 how one can get functional encryption on linear function, called inner-product functional encryption, or IPFE for short.

$\mathsf{KeyGen}(\lambda)$: Given a security parameter λ, it generates a cyclic $\mathbb{G}$ of prime order q, with a generator g. The master secret key MSK is a vector $\vec{s}$ with components s_i are taken randomly from $\mathbb{Z}_q$: $\mathsf{MSK} = \vec{s} = (s_1, \ldots, s_k)$. The public key $\mathsf{PK} = \Big(\mathbb{G}, \big(h_i = g^{s_i}\big)_{i\in[k]}\Big)$. The vector $\vec{x} = (x_1, \ldots, x_k)$ with components x_i is taken from $\mathbb{Z}_q$ is used to extract a functional decryption key $\mathsf{sk}_{\vec{x}} = \langle \vec{s}, \vec{x} \rangle = \sum_{i\in[k]} s_i \cdot x_i (\mod q)$

$\mathsf{Encrypt}(\mathsf{PK}, \vec{y} = (y_1, \ldots, y_k))$: To encrypt a message $\vec{y}$ using public key PK, one picks $r \xleftarrow{\$} \mathbb{Z}_q$ and outputs the ciphertext $\mathsf{CT}_{\vec{y}} = \Big(g^r, \big(h_i^r \cdot g^{y_i}\big)_{i\in[k]}\Big)$

$\mathsf{Decrypt}(\mathsf{sk}_{\vec{x}}, \mathsf{CT}_{\vec{y}})$: The $\mathsf{Decrypt}$ algorithm computes

$$\prod_{i\in[k]} \Big(h_i^r \cdot g^{y_i}\Big)^{x_i} \times \Big(g^r\Big)^{-\mathsf{sk}_{\vec{x}}} = \frac{g^{\langle \vec{s},\vec{x}\rangle r} \cdot g^{\langle \vec{x},\vec{y}\rangle}}{g^{\langle \vec{s},\vec{x}\rangle r}} = g^{\langle \vec{x},\vec{y}\rangle}$$

and gets $\langle \vec{x}, \vec{y} \rangle$, which is supposed to be relatively small, to allow the computation of the discrete logarithm.

Figure 4: The main ingredients of the IPFE of Abdalla *et. al.* [53]

For the mean value, the vector $\vec{x}$ is $(1, \ldots, 1)$. Now, we realize a shortcoming in functional encryption: if many users are interested in the mean value then they all get the same functional decryption key $\mathsf{sk}_{\vec{x}}$ and there will be no way to trace the source of the leakage if this secret key is used somewhere. Allowing many people to get access to the same function, with possible malicious users, is an important problem. As we can see in the above scheme, the functional decryption key is derived from the function and the master secret key, but independently of the user. Therefore, all the users are given the same key, and if this key is leaked, no one can identify the origin of the leakage. The tracing problem becomes critical for this situation. In [56], a new primitive, called Traceable Functional Encryption (TFE), was defined in order to deal with multi-user setting and traceability. By exploiting the similarities between the Boneh and Franklin's traitor tracing scheme [57] and the Abdalla *et al.*'s IPFE scheme [53], it was shown how to integrate the Boneh-Franklin tracing technique into the IPFE scheme of Abdalla *et al.*, which allows in particular the personalization of functional decryption keys. Interestingly, the proposed method of personalizing keys and adding traceability does not need a huge extra cost (as is usually required for others primitives such as broadcast encryption). The construction of efficient traceable functional encryption for more general functions remains an important open problem.

Decentralized functional encryption Functional Encryption is most useful when considering the multi-user case in which the inputs come from different users and the output characterizes a joint function on the input. The main different with multi-party computation is that it does not requires interaction between the users and could be thus very practical. Goldwasser *et al.* [58, 59] introduced the notion of Multi-Client Functional Encryption (MCFE) where the single input x to the encryption procedure is broken down into an input vector $(x_1, \ldots, x_n)$ where the components are independent. An index i for each client and a (typically time-based) label ℓ are used for every encryption: $(c_1 = \mathsf{Encrypt}(1, x_1, \ell), \ldots, c_n = \mathsf{Encrypt}(n, x_n, \ell))$. Anyone owning a functional decryption key dk_f, for an n-ary function f

and multiple ciphertexts for the same label ℓ can compute $f(x_1, \ldots, x_n)$ but nothing else about the individual x_i's. The combination of ciphertexts generated for different labels does not give a valid global ciphertext and the adversary learns nothing from it.

Still, MCFE requires a trusted party to generate a master key msk and to distribute the encryption keys ek_i to the clients and the functional decryption keys dk_f to the decryptors. In practical scenarios, however, the clients do not want to rely on any authority. Consider the following example: a financial firm wants to compute some statistical function on several companies' private data (profits, number of sales) so that it can better understand the dynamics of a sector. The companies may be willing to help the financial firm understand the sector as a whole, or may be offered compensation for their help, but they do not trust the financial firm or each other with their individual data. After setting up a decentralized functional encryption, each company encrypts its private data with a time-stamp label under its private key. Together, they can give the financial firm a decryption aggregation key that only reveals a sum taken over the companies' private data weighted by public information (employee count, market value) for a given time-stamp. New keys can retroactively decrypt aggregates on old data. Motivated by this decentralized setting, Chotard *et al.* [60, 61] introduced the notion of Decentralized Functional Encryption, where no authority is involved, but the generation of functional decryption keys remains an efficient process under the control of the clients themselves. It was stressed that the authority is not simply *distributed* to a larger number of parties, but that the resulting protocol is indeed *decentralized*: each client has complete control over their individual data and the functional keys they authorize the generation of.

We can briefly explain how the scheme in [60] works. Starting from the Abdalla *et al.* [53] described in Figure 4, the idea is to write $c_0 = g^r$ in the single input case and $c_0 = \mathcal{H}(\ell)$ in the Multi-Client case, we have $c_i = g^{x_i} {c_0}^{s_i}$ for $i \in [n]$ in both cases. In the public-key scheme from [53], s_i was private, and only $v_i = g^{s_i}$ was known to the encryptor. Since we are now dealing with private encryption, the encryptor can use s_i. Correctness then follows from

$$g^{\gamma} = \frac{\prod_i c_i^{y_i}}{{c_0}^{\mathsf{dk}_{\vec{y}}}} = \frac{\prod_i (g^{x_i} {c_0}^{s_i})^{y_i}}{{c_0}^{\mathsf{dk}_{\vec{y}}}} = \frac{g^{\sum_i x_i y_i} {c_0}^{\sum_i y_i s_i}}{{c_0}^{\mathsf{dk}_{\vec{y}}}} = \frac{g^{\sum_i x_i y_i} {c_0}^{\mathsf{dk}_{\vec{y}}}}{{c_0}^{\mathsf{dk}_{\vec{y}}}} = g^{\langle \vec{x}, \vec{y} \rangle}.$$

We can easily decentralize the above protocol using standard MPC techniques, but as we mentioned, the main goal is to minimize interactions during the decentralized key generation protocol. This simple protocol can illustrate our main insight: we need to provide the aggregator with the decryption key $\langle vs., \vec{y} \rangle$. Since the s_i's are owned individually by the clients, we are interested in a protocol that would let them send shares from which the decryptor would recover an agreed upon Inner Product on their individual inputs. Relying on a number of techniques, Chotard *et al.* [61] finally get a fully decentralized scheme without any interaction between the users.

The above examples touch upon computations on private data with emphasize on minimizing interactions and decentralization of the operation in the sense of decentralized inputs, distributed (threshold) cryptography, and notions that are based on specialized encryption methods which allow various computations on encrypted data. Various other directions and open questions remain in this and related areas.

4. PRIVACY IN CONTACT TRACING

As we discuss how new technologies respond to the social crisis, contact tracing is certainly one of the best examples of the year 2020. The Covid-19 pandemic demands from us solid preparation for similar natural and social risks in the future: how can technical methods contribute significantly to the control of these surprising disasters? Since in 2020 during the pandemic, global lockdown measures have been imposed all over the world and have been causing serious social and economic problems. A basic central question is: How we can use technical tools to ease lockdown measures? Among all the technical methods, contact tracing has been introduced as the most promising tool (based on users nowadays holding smartphones when they move). The usefulness of these technical methods to control the spread of the disease is still quite limited, mainly due to lack of preparation. The discussion of the privacy and effectiveness of these methods has clarified many social problems associated with the application of new technologies in a mass population. It is therefore important for an in-depth preparation, from a technical and social point of view, so that we are not surprised in the future. In this section, we'll cover contact tracing techniques, their pros and cons.

At the beginning of the pandemic, the first method included keeping logs of users' GPS location data and the users being asked to scan Quick Response (QR) codes. However, it was quickly criticized that privacy risks are important in GPS-based methods because the GPS data may be sent to centralized authorities. Almost all nations were then focused on using some other technology, namely wireless Bluetooth signals, to detect contact matches.

In Bluetooth-based approaches, the main principle is to determine who has been in close physical proximity (determined by Bluetooth Low Energy signals) to an individual who has been diagnosed with the disease. All methods require users to continually run a phone application that broadcasts RPI (pseudo-random Rolling Proximity Identifiers) representing the user and to record RPIs observed from phones in close proximity. Whenever a user is diagnosed positively with COVID-19, the user uploads the seed to generate its RPIs and the application downloads such seed and recognizes the exposure, and then alerts the device owner. In a dual method, if devices register their RPI at a central server the matching of RPIs to uploaded seed's RPIs can be done centrally, and the centyral application alerts the devices from which it had received diagnosis RPIs. Alerts are about a relevant window of time, which is during the infection window, typically 14 days for COVID-19.

Bluetooth-based proposals fall in two main categories of centralized and decentralized methods. The centralized models [62, 63, 64] rely on a trusted third-party (e.g, a government health authority) where the server generates RPIs and thus knows all the RPIs honestly used in the system. It is therefore vulnerable to many privacy issues. The decentralized models, like DP3T [65], PACT [66] and GAEN (Google Apple Exposure Notification API for building Apps on top of it) [67], allow each phone to generate its own RPIs. These RPIs are then exchanged to other phones when a close contact event is detected. This model removes the need of the trusted server who knows everything about the social graph of who met whom. Of course, the method is still vulnerable to several highly dedicated attackers who go out of their way to perform specific targetted attacks: like linkage attacks. For example, an attacker can install BLE-sniffing devices to different known physical locations and collect RPIs. By keeping track of when and where they received which tokens, the attacker can identify who has been diagnosed with the disease as well as the travel route of the individuals [68].

In [69], Vaudenay showed that centralized and decentralized proposals come with their own benefits and risks. Against a malicious authority, the risk of mass surveillance is very high in centralized systems. This risk is lower in decentralized systems because the users generate their tokens themselves. However, the decentralized systems also endanger the anonymity of diagnosed people over other users, as the tokens of diagnosed people are broadcasted to everyone. Vaudenay specified [69]: "centralized systems put the anonymity of all users in high danger, specially against a malicious authority, while decentralized systems put the anonymity of diagnosed people in high danger against anyone." Of course, the situation is that malicious players in reality are very rare among the general population, reducing the exposure of the decentralized system. Under this mostly realistic assumption based on various risk assessments, and due to the ease and suitability of its deployment as an actual product, GAEN ended up being based on the decentralized approach.

The above issues, however, still raise some cryptographic privacy challenges, and cryptographers have suggested further solutions to the general class of proximity based situations. For example, several solutions have been proposed to further prevent linkage attack, as well as to leverage the best of centralized and decentralized systems. The following protocols are in this direction:

- The Epione system [70] uses private set intersection (PSI) protocols on top of decentralized systems: the diagnosis RPIs are not broadcasted. Instead, the user's query is done with the back-end server via an interactive secure computation protocol to compute the cardinality of the intersection (PSI-CA). This system achieves both high privacy and a low volume of data to be downloaded. However, it requires each user to realize the high computation of a two-round interactive protocol with the servers.
- The Pronto-C2 [71] requires diagnosed people to send RPIs to the back-end server. It is about a system where smartphones anonymously and confidentially talk to each other in the presence of the back-end server. Essentially, the back-end server helps users establish shared Diffie-Hellman keys to check whether they are in contact with each other. The main shortcoming of this system is that the client still has to download a large database.
- The DESIRE [72] is presented as an evolution of the ROBERT protocol adopted in France [63]. In this system, for each contact between two phones, a Diffie-Hellman key exchange between is established and stored on each phone, which makes a high barrier for resource-constrained devices.
- Finally, the Catalic [73] is presented as a generalization of the Epione system. The main feature of the Catalic protocol is supporting resource-constrained devices that have limited capacities for computation and storage. The latter problem is solved via the introduction of an efficient delegated PSI-CA while maintaining the user's privacy.

We observe that theoretical tools, namely secure multiparty computation (MPC) technique, have been used in these practical applications. The Epione and Catalic rely on PSI-CA techniques that allows several parties, each holding a set of items, to learn cardinality of the intersection set without revealing anything else about the items. This fits in perfectly with the recent development of rapid PSI implementations driven by many real world applications such as contact discovery [74], botnet detection [75], human genomes testing [?]. Google also runs PSI based solution routinely in its business together with third-party data providers to analyze audiences reaction to advertising and marketing campaigns and issues of this nature without learning individual users identities and data [76, 77]. Still, PSI-CA protocols used in Epione impose a workload on lightweight end-users devices with relatively high bandwidth and heavy computational costs. The core idea of Catalic is to extend PSI-CA to *delegated* PSI-CA to supports resource-constrained devices. In Catalic, every client plays the role of a dealer by dividing each anonymous identifier beacon they collect into shares and giving each share to a cloud server of their choice. At the end, using the results of the cloud servers' computation, clients perform a simple calculation to check whether there is a match (e.g., one that indicates they are at risk). The distinguishing property of the system is that it allows the development of a collaborative and decentralized system of cloud servers all around the world, which may find applications in other contexts. These servers are available to help users who have resource-constrained devices and users can select among all available servers in the delegation. Because this choice is totally hidden from the view of any adversary, the whole system preserves privacy unless a majority of all the servers around the world are corrupted. Resource-constrained devices are thus able to run a heavy PSI-CA protocol with the backend server through a decentralized system of untrusted cloud servers. The secure matching also allows the system to prevent the linkage attack which remains in other systems including GAEN and DP3T.

5. DISCUSSION, PERSPECTIVE, AND SOME FUTURE ISSUES

5.1. Privacy in other primitives

Privacy plays an important role in all digital applications. In this work, we only address some of them, there are many other settings that privacy is primordial, such as:

- Privacy in authentication, in particular blind signatures, group signatures or anonymous credentials.
- Privacy in decentralized platforms such as blockchain, DeFi (decentralized finance) etc.

Besides these platforms, we also need to consider adversaries that use auxiliary channels to mount attacks such as side-channel attacks or subliminal channels. The latter could be a high risk in the future. In a series of works since the 90s [78, 79, 80], Young and Yung presented new threats to the computing infrastructure that are the result of combining malicious software (malware) technology with modern cryptography. They especially raised a question: what if the Trojan horse resides within a cryptographic system itself? They have shown that in many scenarios of black box cryptography (namely, when the code is inaccessible to scrutiny as in the case of tamper-proof cryptosystems or when no one cares enough to scrutinize the code), there are attacks that employ cryptography itself against cryptographic systems, a setting that they called kleptography. Interestingly, this methodology didn't go viral until many years later, but suddenly becomes a primary security threat. The first crypto Trojans just attacked the user's file system based on their initial work on the subject of "weaponizing cryptography" (namely: cryptovirology) first presented in [81]. It was relabelled by the media as ransomware with 181.5 million ransomware attacks in the first six months of 2018. In addition, the NIST Dual EC DRBG random bit generator has an asymmetric backdoor and utilizes the discrete-log kleptogram from kleptography, which makes the EC-DRBG a cryptotrojan based on their work. Because black-box cryptography is both endorsed and employed by the U.S. government, the kleptographic threats have been indeed carefully considered.

This logic of hiding cryptography inside another cryptosystem seems to have possibly other applications, escrow encryption is one immediate possibility. It has remained open to find other application to this new twist on cryptography. Recently we try another way (in fact the opposite way): using subliminal channels to prevent lack of privacy (imposed by some authorities). These unpublished works [82, 83] aim to send hidden messages on channels totally controlled by third parties or to hide the primitives we use inside another primitive in public use. In the following, we will briefly present some particular ideas about this new cryptographic approach.

5.2. Covert privacy: Towards covert cryptographic primitives and protocols

In some situations, we try to hide a message within another message or a physical object. This is the domain of steganography. If the adversary does not realize the existence of an exchanged message, it has no target to attack.

We ask the same question: can we hide a protocol Q within another protocol P? If the adversary is mistaken about our objective, it cannot launch an attack. We call this *Covert Cryptographic Primitives and Protocols (CCPP).*

Note that this is stronger than a black-box implication $P \to Q$. Evidently, if we can hide Q within P then $P \to Q$ but the inverse is not clear. The point is that we directly use P to get Q without any modification. The adversary think that we realize P but in fact we are doing Q. This is, therefore, a kind of subliminal channel, not for the message as in the literature but for the algorithm.

Unlike Kleptography, the CCPP is not modifying the implementation but re-purposing it during its running time! The adversary think it is running under the specified purpose but due to what we do, unnoticeably, we run it with a new purpose in mind (just by changing some randomness, randomness usage, message distribution, etc.). It extends subliminal channels to subliminal protocol hidden inside another protocol.

This will be an subject of an entire new article, here we just give the ideas of how to implement CCPP, to demonstrate how privacy can be achieved in a very covert fashion.

Non-PSI hidden in an anonymous broadcast encryption. We consider the following protocols:

- P is Anonymous Broadcast Encryption (AnonBE).
- Q is Private Set Intersection (PSI)

Then we achieve that: from AnonBE, then we can exploit it to get a Non-interactive PSI without revealing what we do!

We emphasize that Non-interactive PSI is not known in the literature and we can thus not only hide PSI, but also propose the first non-interactive PSI. Here is the way we proceed:

- Suppose A holds a set S_A and the associated secret keys in S_A from an AnonBE scheme.
- B holds a set $S_B = (b_1, ..., b_k)$. B simply send to A the ciphertext $C = \mathsf{AnonBE}(S_B, 0)$ (send for the set of users S_B and encrypt a constant message, say 0^λ, an all zero-bit message for a security parameter λ).
- A simply uses each key in S_A to decrypt C, if it gets 0^λ then the element is in the intersection.

- Because the AnonBE is anonymous, the ciphertext C totally hides the target set S_B which gives the privacy for the PSI scheme.

Obviously, we needs some more detailes and care, and we can use a decentralized version of AnonBE (where the key generation is run in a distributed way) to avoid any leakage to a central authority. We also note that to avoid a dictionary attack, the elements in S_A, S_B should not be predictable, an identity-based-AnonBE such as [84] can be utilized.

Signature is hiding an encryption. Next, we consider the following protocols:

- P is a signature scheme (Sign).
- Q is an encryption scheme (Enc).

Then, we achieve that: from Sign, we can exploit it to get an Enc as follow:

- Suppose A and B share a secret key s for a PRF
- When A wants to send B a message m of k bits for a low k (say 30 bits). It chooses another message x and computes $pad = \mathsf{PRF}(s, x)$.
- A signs x randomly repeatedly until the k low order bits of the signature are $[pad \oplus m]_k$, then send x and the corresponding signature σ to B.
- Upon reception of (x, σ), B computes $pad = \mathsf{PRF}(s, x)$ and then recover $m = [pad \oplus \sigma]_k$

We believe that the above examples will open up a new way to hide the use of one protocol covertly in another and, thus, strengthen privacy.

6. CONCLUSIONS

The survey covered a few typical areas of modern cryptographic research. It pointed at existing techniques, methods, and protocols which enrich our set of tools with which we take care of privacy of individuals and other entities taking part in modern transactions. The emphasize has been on using cryptographic techniques which are backed by mathematical validation to be able to claim their correctness, privacy, and other properties of the tools. We demonstrated that privacy issues are prevalent in many areas of cryptography; we did not try to cover all privacy issues, but with each issue we covered we gave enough information to make the context and the technical background clear so that the examples are self contained to a large extent (we expect the interested reader to use the reference list we provided and learn more on these and related issues). Finally, we also pointed at some developments and some key properties that have been achieved, and we also discussed open issues and coming developments as we see them.

REFERENCES

[1] L. Sweeney, A. Abu, and J. Winn, "Identifying participants in the personal genome project by name (a re-identification experiment)," *CoRR*, vol. abs/1304.7605, 2013. [Online]. Available: http://arxiv.org/abs/1304.7605

[2] D. J. Bernstein, Y.-A. Chang, C.-M. Cheng, L.-P. Chou, N. Heninger, T. Lange, and N. van Someren, "Factoring rsa keys from certified smart cards: Coppersmith in the wild," in *ASIACRYPT 2013, Part II*, ser. LNCS, K. Sako and P. Sarkar, Eds., vol. 8270. Heidelberg: Springer, Dec. 2013, pp. 341–360.

[3] R. Chan, "The cambridge analytica whistleblower explains how the firm used facebook data to sway elections," https://www.businessinsider.fr/us/cambridge-analytica-whistleblower-christopher-wylie-facebook-data-2019-10.

[4] A. Wigderson, "Imitation games," 2021. [Online]. Available: https://www.ias.edu/Wigderson-ImitationGames

[5] S. Goldwasser and S. Micali, "Probabilistic encryption and how to play mental poker keeping secret all partial information," in *14th ACM STOC*. ACM Press, May 1982, pp. 365–377.

[6] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, pp. 469–472, 1985.

[7] Y. Tsiounis and M. Yung, "On the security of elgamal based encryption," in *PKC'98*, ser. LNCS, H. Imai and Y. Zheng, Eds., vol. 1431. Heidelberg: Springer, Feb. 1998, pp. 117–134.

[8] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT'99*, ser. LNCS, J. Stern, Ed., vol. 1592. Heidelberg: Springer, May 1999, pp. 223–238.

[9] J. D. Cohen and M. J. Fischer, "A robust and verifiable cryptographically secure election scheme (extended abstract)," in *26th FOCS*. IEEE Computer Society Press, Oct. 1985, pp. 372–382.

[10] J. C. Benaloh and M. Yung, "Distributing the power of a government to enhance the privacy of voters (extended abstract)," in *5th ACM PODC*, J. Y. Halpern, Ed. ACM, Aug. 1986, pp. 52–62.

[11] R. Cramer, M. K. Franklin, B. Schoenmakers, and M. Yung, "Multi-authority secret-ballot elections with linear work," in *EUROCRYPT'96*, ser. LNCS, U. M. Maurer, Ed., vol. 1070. Heidelberg: Springer, May 1996, pp. 72–83.

[12] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," in *EUROCRYPT'97*, ser. LNCS, W. Fumy, Ed., vol. 1233. Heidelberg: Springer, May 1997, pp. 103–118.

[13] K. Sako and J. Kilian, "Secure voting using partially compatible homomorphisms," in *CRYPTO'94*, ser. LNCS, Y. Desmedt, Ed., vol. 839. Heidelberg: Springer, Aug. 1994, pp. 411–424.

[14] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[15] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract)," in *27th FOCS*. IEEE Computer Society Press, Oct. 1986, pp. 174–187.

[16] M. Abe, "Universally verifiable mix-net with verification work independent of the number of mix-servers," in *EUROCRYPT'98*, ser. LNCS, K. Nyberg, Ed., vol. 1403. Heidelberg: Springer, May/Jun. 1998, pp. 437–447.

[17] J. Furukawa and K. Sako, "An efficient scheme for proving a shuffle," in *CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Heidelberg: Springer, Aug. 2001, pp. 368–387.

[18] C. A. Neff, "A verifiable secret shuffle and its application to e-voting," in *ACM CCS 2001*, M. K. Reiter and P. Samarati, Eds. ACM Press, Nov. 2001, pp. 116–125.

[19] J. Groth and Y. Ishai, "Sub-linear zero-knowledge argument for correctness of a shuffle," in *EUROCRYPT 2008*, ser. LNCS, N. P. Smart, Ed., vol. 4965. Heidelberg: Springer, Apr. 2008, pp. 379–396.

[20] S. Bayer and J. Groth, "Efficient zero-knowledge argument for correctness of a shuffle," in *EUROCRYPT 2012*, ser. LNCS, D. Pointcheval and T. Johansson, Eds., vol. 7237. Heidelberg: Springer, Apr. 2012, pp. 263–280.

[21] C. Hébant, D. H. Phan, and D. Pointcheval, "Linearly-homomorphic signatures and scalable mix-nets," in *PKC 2020, Part II*, ser. LNCS, A. Kiayias, M. Kohlweiss, P. Wallden, and V. Zikas, Eds., vol. 12111. Heidelberg: Springer, May 2020, pp. 597–627.

[22] R. Johnson, D. Molnar, D. X. Song, and D. Wagner, "Homomorphic signature schemes," in *CT-RSA 2002*, ser. LNCS, B. Preneel, Ed., vol. 2271. Heidelberg: Springer, Feb. 2002, pp. 244–262.

[23] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, a. shelat, and B. Waters, "Computing on authenticated data," in *TCC 2012*, ser. LNCS, R. Cramer, Ed., vol. 7194. Heidelberg: Springer, Mar. 2012, pp. 1–20.

[24] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in *PKC 2009*, ser. LNCS, S. Jarecki and G. Tsudik, Eds., vol. 5443. Heidelberg: Springer, Mar. 2009, pp. 68–87.

[25] D. Boneh and D. M. Freeman, "Homomorphic signatures for polynomial functions," in *EUROCRYPT 2011*, ser. LNCS, K. G. Paterson, Ed., vol. 6632. Heidelberg: Springer, May 2011, pp. 149–168.

[26] ——, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *PKC 2011*, ser. LNCS, D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, Eds., vol. 6571. Heidelberg: Springer, Mar. 2011, pp. 1–16.

[27] D. M. Freeman, "Improved security for linearly homomorphic signatures: A generic framework," in *PKC 2012*, ser. LNCS, M. Fischlin, J. Buchmann, and M. Manulis, Eds., vol. 7293. Heidelberg: Springer, May 2012, pp. 697–714.

[28] B. Libert, T. Peters, M. Joye, and M. Yung, "Linearly homomorphic structure-preserving signatures and their applications," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Heidelberg: Springer, Aug. 2013, pp. 289–307.

[29] A. Kiayias and M. Yung, "The vector-ballot e-voting approach," in *FC 2004*, ser. LNCS, A. Juels, Ed., vol. 3110. Heidelberg: Springer, Feb. 2004, pp. 72–89.

[30] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in *Advances in Cryptology — AUSCRYPT '92*, J. Seberry and Y. Zheng, Eds. Springer Berlin Heidelberg, 1993, pp. 244–251.

[31] D. Chaum, M. Jakobsson, R. L. Rivest, P. Y. A. Ryan, J. Benaloh, M. Kutylowski, and B. Adida, Eds., *Towards Trustworthy Elections, New Directions in Electronic Voting*, ser. Lecture Notes in Computer Science. Springer, 2010, vol. 6000.

[32] P. L. Vora, B. Adida, R. Bucholz, D. Chaum, D. L. Dill, D. R. Jefferson, D. W. Jones, W. Lattin, A. D. Rubin, M. I. Shamos, and M. Yung, "Evaluation of voting systems," *Commun. ACM*, vol. 47, no. 11, p. 144, 2004.

[33] F. Zagórski, R. Carback, D. Chaum, J. Clark, A. Essex, and P. L. Vora, "Remotegrity: Design and use of an end-to-end verifiable remote voting system," in *ACNS 13*, ser. LNCS, M. J. J. Jr., M. E. Locasto, P. Mohassel, and R. Safavi-Naini, Eds., vol. 7954. Heidelberg: Springer, Jun. 2013, pp. 441–457.

[34] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of Secure Computation, Academia Press*, pp. 169–179, 1978.

[35] T. Sander, A. Young, and M. Yung, "Non-interactive cryptocomputing for nc1," in *40th FOCS*. IEEE Computer Society Press, Oct. 1999, pp. 554–567.

[36] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *41st ACM STOC*, M. Mitzenmacher, Ed. ACM Press, May/Jun. 2009, pp. 169–178.

[37] A. C.-C. Yao, "How to generate and exchange secrets (extended abstract)," in *27th FOCS*. IEEE Computer Society Press, Oct. 1986, pp. 162–167.

[38] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *TCC 2011*, ser. LNCS, Y. Ishai, Ed., vol. 6597. Heidelberg: Springer, Mar. 2011, pp. 253–273.

[39] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *54th FOCS*. IEEE Computer Society Press, Oct. 2013, pp. 40–49.

[40] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich, "Reusable garbled circuits and succinct functional encryption," in *45th ACM STOC*, D. Boneh, T. Roughgarden, and J. Feigenbaum, Eds. ACM Press, Jun. 2013, pp. 555–564.

[41] A. Fiat and M. Naor, "Broadcast encryption," in *CRYPTO'93*, ser. LNCS, D. R. Stinson, Ed., vol. 773. Heidelberg: Springer, Aug. 1994, pp. 480–491.

[42] D. H. Phan, D. Pointcheval, and M. Strefler, "Decentralized dynamic broadcast encryption," in *SCN 12*, ser. LNCS, I. Visconti and R. D. Prisco, Eds., vol. 7485. Heidelberg: Springer, Sep. 2012, pp. 166–183.

[43] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Heidelberg: Springer, Aug. 2001, pp. 41–62.

[44] A. Sahai and B. R. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT 2005*, ser. LNCS, R. Cramer, Ed., vol. 3494. Heidelberg: Springer, May 2005, pp. 457–473.

[45] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *ACM CCS 2009*, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds. ACM Press, Nov. 2009, pp. 121–130.

[46] A. B. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *EUROCRYPT 2011*, ser. LNCS, K. G. Paterson, Ed., vol. 6632. Heidelberg: Springer, May 2011, pp. 568–588.

[47] A. D. Santis, Y. Desmedt, Y. Frankel, and M. Yung, "How to share a function securely," in *26th ACM STOC*. ACM Press, May 1994, pp. 522–533.

[48] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *TCC 2005*, ser. LNCS, J. Kilian, Ed., vol. 3378. Heidelberg: Springer, Feb. 2005, pp. 325–341.

[49] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *44th ACM STOC*, H. J. Karloff and T. Pitassi, Eds. ACM Press, May 2012, pp. 1219–1234.

[50] T. K. Frederiksen, Y. Lindell, V. Osheter, and B. Pinkas, "Fast distributed rsa key generation for semi-honest and malicious adversaries," in *CRYPTO 2018, Part II*, ser. LNCS, H. Shacham and A. Boldyreva, Eds., vol. 10992. Heidelberg: Springer, Aug. 2018, pp. 331–361.

[51] D. M. Freeman, "Converting pairing-based cryptosystems from composite-order groups to prime-order groups," in *EUROCRYPT 2010*, ser. LNCS, H. Gilbert, Ed., vol. 6110. Heidelberg: Springer, May/Jun. 2010, pp. 44–61.

[52] C. Hébant, D. H. Phan, and D. Pointcheval, "Decentralized evaluation of quadratic polynomials on encrypted data," in *ISC 2019*, ser. LNCS, Z. Lin, C. Papamanthou, and M. Polychronakis, Eds., vol. 11723. Heidelberg: Springer, Sep. 2019, pp. 87–106.

[53] M. Abdalla, F. Bourse, A. D. Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," in *PKC 2015*, ser. LNCS, J. Katz, Ed., vol. 9020. Heidelberg: Springer, Mar./Apr. 2015, pp. 733–751.

[54] S. Agrawal, B. Libert, and D. Stehlé, "Fully secure functional encryption for inner products, from standard assumptions," in *CRYPTO 2016, Part III*, ser. LNCS, M. Robshaw and J. Katz, Eds., vol. 9816. Heidelberg: Springer, Aug. 2016, pp. 333–362.

[55] C. E. Z. Baltico, D. Catalano, D. Fiore, and R. Gay, "Practical functional encryption for quadratic functions with applications to predicate encryption," in *CRYPTO 2017, Part I*, ser. LNCS, J. Katz and H. Shacham, Eds., vol. 10401. Heidelberg: Springer, Aug. 2017, pp. 67–98.

[56] X. T. Do, D. H. Phan, and D. Pointcheval, "Traceable inner product functional encryption," in *CT-RSA 2020*, ser. LNCS, S. Jarecki, Ed., vol. 12006. Heidelberg: Springer, Feb. 2020, pp. 564–585.

[57] D. Boneh and M. K. Franklin, "An efficient public key traitor tracing scheme," in *CRYPTO'99*, ser. LNCS, M. J. Wiener, Ed., vol. 1666. Heidelberg: Springer, Aug. 1999, pp. 338–353.

[58] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou, "Multi-input functional encryption," in *EUROCRYPT 2014*, ser. LNCS, P. Q. Nguyen and E. Oswald, Eds., vol. 8441. Heidelberg: Springer, May 2014, pp. 578–602.

[59] S. D. Gordon, J. Katz, F.-H. Liu, E. Shi, and H.-S. Zhou, "Multi-input functional encryption," Cryptology ePrint Archive, Report 2013/774, 2013, https://eprint.iacr.org/2013/774.

[60] J. Chotard, E. D. Sans, R. Gay, D. H. Phan, and D. Pointcheval, "Decentralized multi-client functional encryption for inner product," in *ASIACRYPT 2018, Part II*, ser. LNCS, T. Peyrin and S. Galbraith, Eds., vol. 11273. Heidelberg: Springer, Dec. 2018, pp. 703–732.

[61] J. Chotard, E. Dufour-Sans, R. Gay, D. H. Phan, and D. Pointcheval, "Dynamic decentralized functional encryption," in *CRYPTO 2020, Part I*, ser. LNCS, D. Micciancio and T. Ristenpart, Eds., vol. 12170. Heidelberg: Springer, Aug. 2020, pp. 747–775.

[62] "Pan-european privacy-preserving proximity tracing," https://github.com/pepp-pt/.

[63] "Robert – robust and privacy-preserving proximity tracing protocol," https://github.com/ROBERT-proximity-tracing/.

[64] "Tracetogether, safer together, a singapore government agency website," https://www.tracetogether.gov.sg/.

[65] C. Troncoso, M. Payer, J.-P. Hubaux, M. Salathé, J. Larus, E. Bugnion, W. Lueks, T. Stadler, A. Pyrgelis, D. Antonioli, L. Barman, S. Chatel, K. Paterson, S. Čapkun, D. Basin, J. Beutel, D. Jackson, M. Roeschlin, P. Leu, B. Preneel, N. Smart, A. Abidin, S. Gürses, M. Veale, C. Cremers, M. Backes, N. O. Tippenhauer, R. Binns, C. Cattuto, A. Barrat, D. Fiore, M. Barbosa, R. Oliveira, and J. Pereira, "Decentralized privacy-preserving proximity tracing," 2020, arXiv:2005.12273.

[66] J. Chan, S. Gollakota, E. Horvitz, J. Jaeger, S. Kakade, T. Kohno, J. Langford, J. Larson, S. Singanamalla, J. Sunshine, and S. Tessaro, "Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing," 2020, arXiv:2004.03544.

[67] "Apple and google privacy-preserving contact tracing," https://www.apple.com/covid19/contacttracing, 2020.

[68] O. Seiskari, "Ble contact tracing sniffer poc," https://github.com/oseiskar/corona-sniffer.

[69] S. Vaudenay, "Centralized or decentralized? the contact tracing dilemma," 2020, cryptology ePrint Archive, Report 2020/531. [Online]. Available: https://eprint.iacr.org/2020/531

[70] N. Trieu, K. Shehata, P. Saxena, R. Shokri, and D. Song, "Epione: Lightweight contact tracing with strong privacy," *IEEE Data Eng. Bull.*, vol. 43, no. 2, pp. 95–107, 2020. [Online]. Available: http://sites.computer.org/debull/A20june/p95.pdf

[71] G. Avitabile, V. Botta, V. Iovino, and I. Visconti, "Towards defeating mass surveillance and sars-cov-2: The pronto-c2 fully decentralized automatic contact tracing system," Cryptology ePrint Archive, Report 2020/493, 2020, https://eprint.iacr.org/2020/493.

[72] "Inria 3rd-way proposal for a european exposure notification system," https://github.com/3rd-ways-for-EU-exposure-notification/project-DESIRE.

[73] T. Duong, D. H. Phan, and N. Trieu, "Catalic: Delegated psi cardinality with applications to contact tracing," in *ASIACRYPT 2020, Part III*, ser. LNCS, S. Moriai and H. Wang, Eds., vol. 12493. Heidelberg: Springer, Dec. 2020, pp. 870–899.

[74] H. Chen, K. Laine, and P. Rindal, "Fast private set intersection from homomorphic encryption," in *ACM CCS 2017*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. ACM Press, Oct./Nov. 2017, pp. 1243–1255.

[75]

[76] M. Ion, B. Kreuter, A. E. Nergiz, S. Patel, S. Saxena, K. Seth, M. Raykova, D. Shanahan, and M. Yung, "On deploying secure computing: Private intersection-sum-with-cardinality," in *IEEE European Symposium on Security and Privacy, EuroS&P 2020*. Genoa, Italy: IEEE, Sep. 2020, pp. 370–389.

[77] P. Miao, S. Patel, M. Raykova, K. Seth, and M. Yung, "Two-sided malicious security for private intersection-sum with cardinality," in *CRYPTO 2020, Part III*, ser. LNCS, D. Micciancio and T. Ristenpart, Eds., vol. 12172. Heidelberg: Springer, Aug. 2020, pp. 3–33.

[78] A. Young and M. Yung, "The dark side of "black-box" cryptography, or: Should we trust capstone?" in *CRYPTO'96*, ser. LNCS, N. Koblitz, Ed., vol. 1109. Heidelberg: Springer, Aug. 1996, pp. 89–103.

[79] ——, "Kleptography: Using cryptography against cryptography," in *EUROCRYPT'97*, ser. LNCS, W. Fumy, Ed., vol. 1233. Heidelberg: Springer, May 1997, pp. 62–74.

[80] ——, "The prevalence of kleptographic attacks on discrete-log based cryptosystems," in *CRYPTO'97*, ser. LNCS, B. S. K. Jr., Ed., vol. 1294. Heidelberg: Springer, Aug. 1997, pp. 264–276.

[81] A. L. Young and M. Yung, "Cryptovirology: Extortion-based security threats and countermeasures," in *1996 IEEE Symposium on Security and Privacy.* IEEE Computer Society Press, 1996, pp. 129–140.

[82] G. Persiano, D. H. Phan, and M. Yung, "Anamorphic encryption: Private communication against a dictator," 2021, unpublished work.

[83] D. H. Phan and M. Yung, "Covert cryptographic primitives/protocols (ccpp)," 2021, unpublished work.

[84] X. T. Do, D. H. Phan, and M. Yung, "A concise bounded anonymous broadcast yielding combinatorial trace-and-revoke schemes," in *ACNS 20, Part II*, ser. LNCS, M. Conti, J. Zhou, E. Casalicchio, and A. Spognardi, Eds., vol. 12147. Heidelberg: Springer, Oct. 2020, pp. 145–164.

Received on June 01, 2021
Accepted on August 13, 2021