A Fast Cryptographic Protocol for Anonymous Voting

L. Zahhafi, O. Khadir

Laboratory of Mathematics, Cryptography, Mechanics and Numerical Analysis. Fstm,

University Hassan II of Casablanca, Morocco

Leila.zahhafi@gmail.com

khadir@hotmail.com

Abstract

In this work, we discuss the problem of electronic voting. This notion has become widely sought in the world, which justifies the efforts made by researchers in this field. Voting by electronic means does not facilitate the task only for the organizers, but also for the voters who can send their choices from the home. Our system of binary electronic voting is based on Paillier cryptosystem. We chose this protocol as it is an additive homomorphism which will facilitate the calculation of the final vote results. The method presents a great difficulty in the decryption for attackers as it is based on the problem of factoring large numbers.

The protocol that we propose guarantees the anonymity of the vote, i.e. no one should know the vote of an elector. We also worked on the control of the parties holding the ballot. This increases the security, reliability and integrity of the vote. We have introduced several cryptographic notions to create an effective scheme.

Indexing terms/Keywords

Electronic voting scheme, Paillier cryptosystem, RSA signature, Homomorphism, Encryption, Decryption, Identification, Anonymity, Security

SUBJECT CLASSIFICATION

E.g., Mathematics Subject Classification: 94A60, 91B12.

Language: English

Date of Submission: 2018-03-01

Date of Acceptance: 2018-04-10

Date of Publication: 2018-04-30

Journal: *MATHLAB Journal*

Website: https://purkh.com

1. INTRODUCTION

Communication between people is a crucial need in their lives even before the appearance of computers. But with the development of the Internet network, it has begun having security risks. And this is where cryptography plays an indispensable role in protecting data transmitted by Internet. Today, several areas rely on this science to ensure the security of our services and online transactions.

Voting is an act allowing the expression of an opinion during a ballot to make a final decision. For a long time, we know the classic voting process, where voters move to the polling stations to mark their choices in paper envelopes. However, work on improving voting techniques has taken place over a long period of time.

In 1906, Boggiano [3] invented a voting machine to collect, count and automatically sum the results of a vote. Subsequently, the researchers do not stop on the automatization of the vote. Many of them have sought to develop methods of voting to make it more practical and effective. Based on public key cryptography, since 1981, Chaum [4] has been working on the problem of traffic analysis. Then, work on anonymous communication was integrated with the innovation of strong electronic voting protocols [2,9,11]. By observing the different conceptions of the researchers in the study of this concept, we note that several have worked on multi-authority schemes [5,6]. Which allows to generate the private key in cooperation between the different authorities. This type of model also leads to a deciphering in common. What strengthens the security of the voting system, since a single authority cannot decipher a voter's vote.

In 1994, other researchers as [10] gave the voter the opportunity to vote from a specific number of centers. As a result, the voter will not vote on a single destination. This prevents fraud and increases the level of security. It also enables the various centers to verify that all votes are considered.

The use of the homomorphic system [1,15] has facilitated the creation of simple and practical voting protocols. That allows to decipher just the product of all the votes cast to find the result of elections. This concept is considerable as it plays a very important role in ensuring the confidentiality of anonymous votes.

We relied on these ideas to create a new electronic voting scheme. Our paper presents the application of the Paillier cryptosystem [12] to model a binary electronic voting. And we note that this cryptosystem was also used by A. Acquisti in 2003 [1]. It is a homomorphic system. It means that from a single deciphering, we get the results of the vote. In our protocol, we manage a binary vote. So, electors will have only two choices yes or no. We detail below how this cryptosystem will give us exactly the number of electors who voted yes.

The paper is organized as follows: We show in the next section the different steps of the encryption and decryption by the Paillier cryptosystem, and an example to show its functioning. In the third section, we recall how the RSA signature works. Then, we will describe our voting protocol in the fourth section. In section five, we give an example and we finish by a conclusion in section six.

We denote by N the set of natural numbers. By Z the set of integers. *gcd(a,b)* expresses the greatest common divisor of *a* and *b* and *φ(n)* is the Euler function.

2. THE PASCAL PAILLIER CRYPTOSYSTEM

The Paillier cryptosystem [12] is a cryptographic algorithm proposed in 1999. This system presents an additive homomorphism. It means that we can calculate the encryption of the sum of two messages using the encryptions of each. The cryptosystem works as follow :

Alice chooses two prime numbers p and q and generates the public key $n = pq$. She selects them such that: $\gcd(n, \varphi(n)) = 1$. And she finds the secret key *d* such that: $n.d \equiv 1[\varphi(n)]$ and $\varphi(n) = (p-1)(q-1)$.

2.1. Encryption

If Bob wants to send the message M to Alice, he will do it in two steps:

1. He chooses a random number $r \in \{1,2,3,\ldots,n-1\}$, with $\gcd(r,n) = 1$.
2. He sends $C \equiv (n.M+1).r^n\ [n^2]$ to Alice.

2.2. Decryption

Theorem 1 :

If $C \equiv (n.M+1).r^n\ [n^2]$, then $M \equiv \frac{C.S-1}{n}\ [n]$. Where: $S \equiv \frac{1}{r}^n\ [n^2]$ and $r \equiv C^d[n]$.

Proof :

$C = (n.M\ +\ 1).r^n +\ K.n^2 \quad \rightarrow \quad C^d = ((n.M\ +\ 1).r^n +\ K.n^2)^d \quad \rightarrow\ C^d \equiv r^{n.d}\ [n]$ where: $K \in Z$.

Since $n.d \equiv 1[\varphi(n)]$, so $r \equiv C^d[n]$.

Alice determines the number s such as: $S \equiv \frac{1}{r}^n\ [n^2]$. And finally, she finds the message M sent by Bob as follows: $C \equiv (n.M+1).r^n\ [n^2] \ \rightarrow\ \frac{C}{r^n} \equiv (n.M+1)\ [n^2] \ \rightarrow\ C.s \equiv (n.M+1)\ [n^2] \ \rightarrow\ C.s = n.M\ + 1 + k'.n^2 \ \rightarrow\ \frac{C.s-1}{n} = M +\ k'.n$

This gives the following result: $M \equiv \frac{C.S-1}{n}\ [n]$.

Example 1 :

Suppose that Alice chooses $(p,q) = (1657{,}1057)$, and generates the public key: $n = p.q = 1817729$. So, $\varphi(n) = 1814976$ and $\gcd(n,\varphi(n)) = 1$.

We assume that Bob wants to send the message $M = 654329$ to Alice.

Bob selects the random number r= 754323 which is prime with $n = 1817729$.

He sends $C \equiv (n.M+1).r^n \equiv 1861049193970\ \ [n^2]$ to Alice.

Alice finds the message M by these following steps:

1. She calculates her secret key $d \equiv \frac{1}{n} \equiv \frac{1}{1817729} \equiv 1180097\ [\varphi(n)]$.
2. She determines the value of *r* by using her private key *d*.
   $r \equiv C^d \equiv 1861049193970^{1180097} \equiv 754323\ [n]$.
3. $S \equiv \frac{1}{r}^n \equiv \frac{1}{754323}^n \equiv 343821977573\ [n^2]$.
4. $M' \equiv \frac{C.S-1}{n} \equiv 654329\ [n]$.
   $M' = M = 654329$.

3. RSA SIGNATURE (1978)

It is based on the problem of factorization of large numbers which presents great difficulties until now. We call Alice the person who wants to sign a message, and Bob the person who verifies Alice's signature. Alice chooses two large primes p and q and makes their product $n = pq$. Then she chooses an integer $e \in N$ such as $\gcd(e, \varphi(n)) = 1$. Its secret key d verifies : $d \equiv \frac{1}{e} [\varphi(n)]$.

3.1. Signature equation

Alice signs the message M of Bob using its secret key d. She calculates: $S \equiv M^d[n]$ and sends it to the verifier Bob.

3.2. Signature verification

After receiving Alice' signature S, Bob checks this fallowing equation using the public key e: $S^e \equiv M[n]$.

4. OUR VOTING SCHEME

4.1. Description of the Protocol

In this section, we will apply the Paillier cryptosystem to implement a binary electronic voting scheme. Each elector will have only two voting choices: He either votes for *(A = Yes = 1)* or *(B = No = 0)*.

4.1.1. Elements holding the vote

We are introducing four elements to organize this protocol. Each element has a specific and indispensable role for the continuity of the voting circuit. The objective of intervening several authorities is to reduce the power of each for increasing the level of security. Their tasks are as follow :

- **Administrator :** He generates the keys used in the voting process.
- **Intermediaries :** They receive the encrypted votes of the voters and calculate their product then send it to the Decipherer.
- **Controllers :** The role of these elements is to verify that intermediaries haven't cheat by changing the electors' vote.
- **Decipherer :** He decodes the encrypted vote and calculates the final result.

4.1.2. Keys generation

As we use the Paillier cryptosystem to encrypt electors' vote, the administrator generates two prime numbers p and q, and calculates the public key $n = pq$ such that: $\gcd(n, \varphi(n)) = 1$. Then he finds the secret key d that verifies : $nd \equiv 1[\varphi(n)]$. This secret key will be used by the Decipherer in the calculation step of the result. So, the administrator gives the value of d to the Decipherer. The value of n will be public and used to encrypt the votes.

The administrator also generates the signature keys for controllers. So, he assigns each one a public exponent e_j and its secret key d_j such as: $e_j . d_j \equiv 1[\varphi(n)]$.

4.1.3. Polling steps

We suppose that there are $K \in N$ electors $\{ E_1, E_2, \ldots, E_i, \ldots E_k \}$, each of them will send his voting message M_i. The system will then encode this message using Paillier cryptosystem, and send the code C_i to an intermediary, a trusted party between electors and Decipherer.

To avoid cheating between this intermediary and the Decipherer, we introduce $l \in N$ intermediaries and l controllers. We associate every intermediary l_i to a controller Ct_i. Then we will not trust only one party. So, the ciphertext C_i of every voter will be sent to an intermediary in a random way. And the corresponding controller receives the value : $f(C_i) \equiv (C_i)^2 [n^2]$.

Every intermediary I_j will collect the coded choices of a specific number of voters, then calculates their product: $Cj \equiv \prod_{i=1}^{Sj} Ci \ [n^2]$.

With: $j \in \{1,2,\dots,l\}$ and s_j is the number of electors who send their ciphertexts to the intermediary *j*.

In the same time, the controller corresponding to this intermediary collects squares of the coded choices, then calculates their product: $f(C_j) \equiv \prod_{i=1}^{Sj} f(Ci) \equiv \prod_{i=1}^{Sj} (Ci)^2 \ [n^2]$.

At the time *t*, end of the voting period, I_j sends the value C_j to the the controller Ct_j.

In this step, every controller Ct_j calculates: $val \equiv C_j^2$ [n^2] and compares it with the value: $f(C_j)$. If they are equal he signs C_j and sends $S(C_j)$, the signed product, to intermediary l_j.

To sign the values C_j, each controller must have his secret key. there are many digital signature protocols that can be implemented in our voting scheme as the signature of Elgamal [7] and the RSA protocol based on the factorization problem [14]. In our case, the work with RSA's signature is more suitable. Indeed, we assign the same public key n^2 to all controllers but differents exponents e_j. So every controller will have a private key that allows him to sign the value received from the intermediary.

After receiving $S(C_j)$, each intermediary l_j sends the couple $(C_j, S(C_j))$ to the Decipherer.

Remark:

To participate in elections, each voter must go through an identification step. However, there are several methods of identification to limit access to the voting system such as the Fiat-Shamir [8] and Quillou-Quisquater [10] protocols which are inspired by the RSA algorithm. Also we can report the identification schemes of Shnorr [16] and Okamoto [13] which are based on the discrete logarithm problem. All of these schemes exploite the concept of zero-knowledge proof, which means that voters can identify to our voting system without disclosing their secret keys in order to guarantee the anonymity of the vote.

4.1.4. Calculation of the voting results

Now, Decipherer receive the couples { $(C_1,S(C_1))$, $(C_2,S(C_2))$,..., $(C_l,S(C_l))$ }, the product of all the electors' coded choices of each intermediary with their signatures by controllers. In the first, he verifies the validity of signatures, then he calculates: $C \equiv \prod_{i=1}^{l} C_i$ [n^2] and decodes it to find the message: $\sum_{i=1}^{k} M_i$ such as:

- M_i presents the message of the elector number i . So $M_i \in \{0,1\}$.
- *k* is the total of electors participating in this vote.
- *M* presents the voting result, and more precisely, the number of voters who voted for the choice $A = 1 = yes$.

We show that *M* presents the voting result, in other words, the number of voters who voted for the choice $A = 1 = yes$.

We used the Paillier cryptosystem to code the choices of voters. This system is an additive homomorphic cryptosystem. It means that if we code two messages M_1 and M_2 to obtain C_1 and C_2, the encryption of $(M_1 + M_2)$ will be the product of C_1 and C_2 mod n^2. Indeed:

- $C_1 \equiv (n.M_1 + 1).r_1^n \ [n^2]$
- $C_2 \equiv (n.M_2 + 1).r_2^n \ [n^2]$
- $C \equiv C_1.\ C_2 \equiv (n.(M_1+M_2) + 1).r^n \ [n^2]$

So, by decoding *C* we find $(M_1 + M_2+\dots+M_k) \bmod n$. Since $k < n$, $(M_1 + M_2+\dots+M_k) \equiv M_1 + M_2+\dots+M_k \ [n]$.

Theorem 2:

If among the k voters k′ send (Yes=1) and k″ send (No=0) then, the results M of this vote will be as follows: $M = \sum_{i=1}^{k} M_i$ = k′.1 + k″.0 = k′.

Proof:

In the binary vote, the message presenting the choice of the voter is *0* or *1*. So, we recapitulate the result of the vote like this : M voters voted by *Yes* and $k - M$ voted by *No*.

4.2. Security analysis

Attack 1: Assume that Oscar is an attacker. If he intercepts the value of C_i and tries to find M_i, the choice of the elector i. While $M_i \in \{0,1\}$, he will replace M_i by its value, and check the result by using the value of C_i in equation : $C_i \equiv (n.M_i + 1).r_i^{\,n}\ [n^2]$. But he will be confronted by the number r_i that he doesn't know. So the two tests don't allow Oscar to know the voter's choice.

Attack 2: The Decipherer is the only party that has the secret key to decipher a message. Since our voting protocol ensures the anonymity of the vote, even this Decipherer must not decipher any voter's vote. And that's why we introduced intermediaries, the trusted parties, that collect the votes and calculate their products before sending them to the Decipherer.

Attack 3: If an intermediary tries to modify an encrypted vote, the controller will not sign his wrong product C_j. Then, all intermediaries have to collect and transmit every information in an honest way.

4.3. Complexity

While k voters participate on the election, in the voting step, the system will encode k choices. So the number of operations that must be calculated is: *2k* modular multiplications and k modular exponentiations.

And to control the work of intermediaries, the system will calculate the square of encrypted votes and send the results to controllers. Then, there will be k modular exponentiations to perform. Again, in the signing step, controllers sign l values with the RSA signature which leads to perform l modular exponentiations and k modular multiplications.

In the result step, the decipherer will execute l signature's verification by calculating l modular exponentiations. Then, he executes one decryption with Paillier cryptosystem. So, there are $l+2$ modular exponentiations, tree divisions and one multiplication.

The time required to execute all the voting operations is as follows:

$$T = (3k+1)T_{mult} + (2k + 2l + 2)T_{exp} + 3T_{div}$$

$$= (3k+1)O((logn)^2) + (2k + 2l + 2)O((logn)^3) + 3O((logn)^2)$$

$$= O((logn)^2) + O((logn)^3)$$

$$= O((logn)^2 + O(logn)^3).$$

Finally, as a result, we can assume that our voting system works on a polylogarithmic time.

5. EXAMPLE

Suppose there is a company wants to make a decision to accept or refuse a proposed program. It intervenes its employees to decide, then the management organizes an electronic voting to hang the decision.

There are *40* employees participating in the vote, *3* intermediaries, *3* controllers, one decipherer and an administrator.

So, the administrator chooses the primes $(p, q) = (1861,1867)$, then generates the public key $n = p.q = 3474487$ that verifies: $\gcd(n, \varphi(n)) = 1$. He calculates the decryption key $d \equiv \frac{1}{n} \equiv \frac{1}{3474487} \equiv 793423\ [\varphi(n)]$ and sends it to the decipherer.

The administrator affects signature keys for each controller. So, controller Ct_1 receives $(e_1,d_1) = (7,6890920285783)$, controller Ct_2 receives $(e_2,d_2) = (11,5481413863691)$ and controller Ct_3 receives $(e_3,d_3) = (13,7420991076997)$. Values e_1, e_2 and e_3 are publics but d_1, d_2 and d_3 are secrets.

During the voting period the system calculates the following values:

Table 1. The encrypted votes and their squares modulo n^2

Employe E_i	V_i	C_i	$f(C_i) \equiv C_i^2\ [n^2]$
E_1	Yes	3751924240949	3196728718673
E_2	Yes	9093024508119	7298325126365
E_3	No	7741058637282	10298950875697
E_4	Yes	1394093085449	9445138624768
E_5	No	4713239518252	6785600733133
E_6	No	371065774731	9642549339190
E_7	Yes	6886882955724	10848490527469
E_8	Yes	10180990487833	4071793781160
E_9	No	1594016618288	11169617586175
E_{10}	Yes	6171570242030	7784659605423
E_{11}	No	6895010258688	1043144791444
E_{12}	Yes	2140225750492	2167483990356
E_{13}	Yes	9284881979281	1923892895064
E_{14}	No	5191366756613	4741302808315
E_{15}	Yes	1197347824403	3410101033677
E_{16}	No	6174075001691	310306008923

E_{17}	No	3691048252574	6163105694069
E_{18}	No	8491779847275	1408824491056
E_{19}	Yes	505407843581	5488423133733
E_{20}	Yes	10309754374966	8542448775151
E_{21}	No	11405127212374	10688613344789
E_{22}	No	7267494479281	6615527600107
E_{23}	Yes	3306090845272	11247684905599
E_{24}	Yes	10546257855732	767600113655
E_{25}	No	7500822411687	7994214161570
E_{26}	Yes	11212869832108	8066840857816
E_{27}	No	11201187123200	2951294666640
E_{28}	Yes	8138992439477	9015284387417
E_{29}	No	9205649525016	9287055286070
E_{30}	Yes	6577761628034	7236715002874
E_{31}	Yes	9280794787299	2446820418554
E_{32}	Yes	4118980340965	3139535635085
E_{33}	Yes	8126553658412	11716737228088
E_{34}	No	11891079006375	7024588036317
E_{35}	Yes	78455701533	11260474184520
E_{36}	Yes	5556193502501	5359577845187
E_{37}	Yes	6064836129774	9993804696757
E_{38}	No	549700468882	9646579935070
E_{39}	Yes	9700948743606	9595683981678
E_{40}	No	8428306696272	6817835099665

Suppose the system sends: *{ C_1, C_2, ... , C_{15} }* to the intermediary *I_1*. So, controller *Ct_1* receives: *{ $f(C_1)$, $f(C_2)$, ... , $f(C_{15})$ }*. Then, it sends: *{ C_{16}, C_{17}, ... , C_{30} }* to the intermediary *I_2*. Controller *Ct_2* receives: *{ $f(C_{16})$, $f(C_{17})$, ... , $f(C_{30})$ }*. The last intermediary receives: *{ C_{31}, C_{32}, ... , C_{40} }* and controller Ct_3 gets : *{ $f(C_{31})$, $f(C_{32})$, ... , $f(C_{40})$ }*.

After the end of the voting period:

- Intermediary *I_1* calculates: $X \equiv \prod_{i=1}^{15} C_i \equiv 751246028294\ [n^2]$. And sends the result to the controller Ct_1.
- Intermediary *I_2* calculates: $Y \equiv \prod_{i=16}^{30} C_i \equiv 8604416976262\ [n^2]$. And sends the result to the controller Ct_2.
- Intermediary *I_3* calculates: $Z \equiv \prod_{i=31}^{40} C_i \equiv 2355091856266\ [n^2]$. And sends the result to the controller Ct_3.

We detail the control process as follows:

Controller Ct_1 calculates: $val_1 \equiv \prod_{i=1}^{15} f(C_i) \equiv 9746762852670\ [n^2]$. And $f(X) \equiv X^2 \equiv 9746762852670\ [n^2]$ Since $val_1 = f(X)$, he signs *X*. So, he sends: $S(X) \equiv X^{d1}[n^2]$ to intermediary *I_1*.

Controller Ct_2 calculates: $val_2 \equiv \prod_{i=16}^{30} f(C_i) \equiv 3423401255655[n^2]$. And $f(Y) \equiv Y^2 \equiv 3423401255655[n^2]$ Since $val_2 = f(Y)$, he signs Y. So, he sends: $S(Y) \equiv Y^{d2}[n^2]$ to intermediary *I_2*.

Controller Ct_3 calculates: $val_3 \equiv \prod_{i=31}^{40} f(C_i) \equiv 7812734475226[n^2]$. And $f(Z) \equiv Z^2 \equiv 7812734475226[n^2]$ Since $val_3 = f(Z)$, he signs Z. So, he sends: $S(Z) \equiv Z^{d3}[n^2]$ to intermediary *I_3*.

Now, Intermediaries send values: *{ (X,S(X)), (Y,S(Y)), (Z,S(Z)) }* to the decipherer.

In the first, the decipherer checks that *S(X), S(Y)* and *S(Z)* are correct using e_1, e_2 and e_3 (see 3.2). Then, he executes $C \equiv X.Y.Z \equiv 5893815063801\ [n^2]$ and decodes this value as follows:

He calculates $r \equiv C^d \equiv 2188228\ [n]$, then he finds: $s \equiv \frac{1}{r^n} \equiv 4010950335138\ [n^2]$ and finally, he gets the result: $M \equiv \frac{C.s-1}{n} \equiv 23[n]$.

So, the final result of this vote is: *23* employees are agree with the decision of the company and $40 - 23 = 17$ are not.

6. CONCLUSION

In this paper we presented a new system of binary electronic voting based on Paillier cryptosystem. The protocol we presented is well secured as we have introduced several authorities, each one controls the work of the other. Also, we have involved solid cryptographic concepts as the homomorphe encryption system and the digital signature.

ACKNOWLEDGMENTS

This work is supported by the MMS e-orientation project.

REFERENCES :

1. A. Acquisti, 2004. Receipt-Free Homomorphic Elections and Write-in Ballots. International as sociation for cryptologic research, May 2, 2004, and Carnegie Mellon institute for software.

2. J. Benaloh, 1987. Verifiable secret-ballot elections. Ph.D. thesis, Yale university, Technical report 561.

3. Boggiano, 1906. Le pséphographe. Henry Bidou.

4. D. L. Chaum , 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 24, 84 90.

5. R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung, 1996. Multi-authority secret ballot elections with linear work. Advances in Cryptology EUROCRYPT '96, volume 1070 of Lecture Notes in Computer Science,pages 72 83, Berlin.

6. R. Cramer, R. Gennaro, B. Schoenmakers, 1997. A secure and optimally efficient multi-authority election sheme. Advances in Cryptology- EUROCRYPT' 97, LNCS 1233, 103-118.

7. T. Elgamal 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inf Theory. 31 (4): 469-472.

8. A. Fiat and A. Shamir , 1987. How to prove yourself: practical solutions to identification and signature problems. Springer-Verlag, Lecture notes in computer science, No 263, Advances in cryptology,Proceedings of Crypto '86, pp. 186-194.

9. A. Fujioka, K. Ohta, T. Okamoto , 1993. A practical secret voting scheme for large scale elections. Y.Zheng, J.Seberry, (eds.) AUSCRYPT. LNCS, vol 718, 248-259.

10. L. Guillou and J. Quisquater. A Practical Zero-Knowledge Protocol fitted to Security Micro-processor Minimizing both Transmission and Memory. Proc. of EuroCrypt 88, Springer VerlagLNCS series.

11. K. Iversen, 1992. A cryptographic scheme for computerized general elections. Proc. CRYPTO '91, Springer LNCS 576, pp: 405 - 419.

12. P. Paillier, 1999. Key Cryptosystems based on composite degree residuosity classes. Eurocrypt, 223-238.

13. T. Okamoto, 1992. Provably secure and practical identification schemes and corresponding signature schemes. Brickell E.F. (eds) Advances in Cryptology Crypto '92. Lecture Notes in Computer Science, v.740. Springer, Berlin, Heidelberg.

14. R. Rivest, A. Shamir, L. Adleman, February 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM. 21 (2): 120-126.

15. K. Sako and J. Kilian, 1994. Secure voting using partially compatible homomorphisms. CRYPTO '94: 14th International Cryptology Conference, volume 839 of LNCS, 411-424.

16. C.P. Schnorr, 1989. Efficient identification and signatures for smart cards. Brassard G (ed) CRYPTO, Lecture notes in computer science, vol 435. Springer, Berlin, pp 239-252.

Author's short biography

Dr Omar Khadir received his Ph.D. degree in Computer Science from the University of Rouen, France (1994). Co-founder of the laboratory of Mathematics, Cryptography, Mechanics and Numerical Analysis at the University of Hassan II Mohammedia, Morocco, where he is a professor in the Department of Mathematics.He was head of the deppartement of Mathematics until January 2018. He teaches cryptography for graduate students preparing a degree in computer science. His current research interests include public key cryptography, digital signature, primality, factorisation of large integers and more generally, all subjects connected to the information technology.

Leila Zahhafi holds an engineer degree in Computer Science and Mathematics from the University of Hassan II of Casablanca Mohammedia (2016). Member of the laboratory of Mathematics, Cryptography, Mechanics and Numerical Analysis, she is currently a Phd Student. Her research interest is public key cryptography.