A ROBUST THRESHOLD ELLIPTIC CURVE DIGITAL SIGNATURE PROVIDING A NEW VERIFIABLE SECRET SHARING SCHEME

Maged H. Ibrahim I. A. Ali I. I. Ibrahim A. H. El-sawi
Communication department, Faculty of engineering
Helwan University
Helwan, Cairo,Egypt

Abstract - *Robust threshold digital signature schemes are group signature schemes aiming to depart from the classical one person signer schemes. The term 'robust' means that such schemes can tolerate errors attempted by malicious adversary and the term 'Threshold' means that given a total of n players, no coalition of players with cardinality less than or equal the threshold value can perform the signature while any coalition of players exceeding the threshold value can perform the signature correctly. The contributions in this paper are two fold. First, we propose a new verifiable secret sharing scheme (VSS) other than Feldman's [1] and Pedersen's [2] schemes suitable to protect elliptic curve secret keys. The proposed scheme utilizes a strong one way function provided by the elliptic curve cryptography based on a different type of group mathematics. Next, we employ the elliptic curve VSS to propose a robust threshold elliptic curve digital signature scheme that can withstand an n/2 eavesdropping, n/3 halting and an n/4 malicious adversary. The scheme is able to tolerate n/3 malicious adversary with the cost of higher complexity.*

I. INTRODUCTION

The tremendous development in computer networks and the internet opens up terrific opportunities for cooperative computations where the output depends on private inputs of separate entities. These computations may occur among mutually entrusted entities. The solution is trivial if there is some trusted party that collects all the private inputs, perform the computation and provide the correct outputs. However, trusted parties do not exist, at least all the time. Secure multiparty computations and secret sharing provide useful solutions enabling honest entities to perform correctly even in the existence of some cheaters in the neighborhood.

Elliptic curves can provide versions of public-key methods that are faster and use much smaller keys, while providing an equivalent level of security. Their advantage comes from using a different kind of mathematical group for public-key arithmetic. All practical public-key systems today exploit the properties of arithmetic using large finite groups. For many methods, like Diffie-Hellman, El-Gamal, and DSS, the security depends directly on the relative difficulty of performing two group operations: Exponentiation vs. Discrete logarithm. Exponentiation must be easy, compared to discrete log, its inverse operation. In the commonly used groups, using modular multiplication of integers, discrete log is only hard when the modulus is very large. This makes exponentiation a "one-way" function. But large exponentiation is expensive, so it is important to find groups that work efficiently. The elliptic curve discrete log problem is different and harder than discrete log in ordinary groups. Implementations can exploit this difference to provide both increased speed and decreased key size for a given level of security. Many public-key methods can easily work with elliptic curves, or any suitable group, to take advantage of the best available implementations. Elliptic curve cryptography (ECC) offers secure and efficient solutions for the new communication technologies. It requires fewer bits than the RSA for similar amount of security. While the ECC provides shorter key sizes, the time and code size requirements may still be excessive. Thus, efficient and optimized implementations are required for the restricted platforms particularly found in wireless communication. To achieve reasonable security, RSA and DSA should employ 1024 bit moduli, while a 161 bit modulus should be sufficient for ECC. Moreover, the security gap between the systems increases dramatically as the moduli sizes increases. For example, 300 bit ECC is dramatically more secure than 2048 bit RSA or DSA [3, 4, 5, 6, 7, 8].

The idea of distributed signature schemes is to allow a group of people to hold the key in such a manner that they can as a group produce signatures yet no person on his own can generate a signature. The signature which is generated by the group is the same as if it was generated by a single signer and the group can perform the signature as many times as they wish. More in detail the secret key is shared by a group of *n* players. In order to produce a valid signature on a given message *m*, players engage in a communication protocol whose result will be a full signature on *m*. The signature scheme must be transparent to the verifier. Also, if *t* is the threshold then no *t* or less coalition of the *n* players can perform the signature.

II. PREVIOUS WORK, MOTIVATIONS AND CONTRIBUTIONS

Secret sharing was first introduced by Shamir in [9] and since then many papers and researches appeared in this field discussing wide variety of applications and analysis. Threshold signatures represent one of the important applications of secret sharing enabling existing digital

signature schemes to depart from the classical one person signer schemes to the group signature in a fault tolerable manner. Solutions to threshold signatures can be found in [10, 11] for the case of RSA signatures and in [12, 13] for the case of El-Gamal type signatures. [14] studied the threshold DSS signature scheme. However, the schemes presented were not robust i.e. the schemes were not able to tolerate malicious adversary. Lately, [15, 16, 17] studied the threshold DSS signatures and provided efficient, fast, practical and robust schemes to tolerate $n/2$ eavesdropping and $n/3$ halting adversary, the scheme is also able to tolerate $n/3$ malicious adversary where n is the total number of players involved in the system.

ECC is looming at the horizon to be the next generation public key cryptosystem and digital signature scheme, also providing an excellent one way function relying on a different type of computations. The contributions of this paper are two fold. First we propose a new verifiable secret sharing scheme (VSS) other than Feldman's [1] and Pedersen's [2] VSS schemes utilizing the strength of the EC one way function to protect shorter key lengths. The motivation behind employing the EC one way function is that Feldman's and Pedersen's schemes which depend on the discrete log problem become risky and inefficient if they are used to protect secret keys of the ECC. Numerically, 160 bits secret key requires 161 bits EC public parameters, where it requires 1024 bits public parameters if Feldman's or Pedersen's schemes are employed which consequently destroys the merit of the ECC. Next, we propose a robust threshold ECDSA secure against eavesdropping, halting and malicious adversary. The techniques have much similarities to that of the robust threshold DSS but with higher information rates arising from the merits of the EC one way function over the exponentiation and discrete log problem functions.

III. THE MODEL

In the communication model, we assume a system of n servers (players). The servers are connected by a complete network of private point to point channels. In addition, the players have access to a dedicated broadcast channel" by dedicated we mean that if player P_i broadcasts a message it will be recognized by the other players as coming from P_i. Also the system must be synchronous that is all the players are tight to a master clock.

In the adversary model, the adversary we are dealing with throughout the work in this paper is a static adversary, that is, the players corrupted by the adversary are decided upon once and do not change during the execution of the protocol. However, we will give notes on how to deal with the so called mobile adversaries later. In general, there is an upper bound on the number of players that can be corrupted by the adversary. Let this upper bound to be t. we distinguish three types of a static adversary: An eavesdropping (passive) adversary, a freezing (halting) adversary and a malicious (active) adversary. This categorization enables us to provide better efficiency for weaker adversaries. A passive adversary sees and learns all information stored at the corrupted nodes but will not compromise or prevent the player from behaving correctly during the protocol. A freezing adversary is also an eavesdropping one, plus her capabilities to disconnect and crash a player to stop him from contributing in the protocol. Finally the malicious adversary that is able to do almost anything to the corrupted player such as eavesdropping, halting, changing values, substituting values, sending false values…etc.

IV. ELLIPTIC CURVE VERIFIABLE SECRET SHARING SCHEME

A. *Compatibility with Shamir's SSS*

The number of rational points that satisfies the elliptic curve equation represents the order of the curve and is divisible by a large prime r. Note that $rP = O$ for any basic point P. It is important to notice that for any set of integers k, k_1, k_2, …; if $kP = k_1P + k_2P + \ldots \bmod p$ then $k = k_1 + k_2 + \ldots \bmod r$. Also if $kP = (k_1 k_2 \ldots.)P \bmod p$ then $k = k_1 k_2 \ldots \bmod r$. We press on the fact that Z_r is a field, and it is possible to perform Lagrange interpolation modulo r and hence Shamir's SSS can be directly designed over Z_r which is not the case for other public encryptions such as RSA.

B. *ECVSS with a dealer*

Given an elliptic curve E defined over Z_p. The number of points in $E(Z_p)$ should be divisible by a large prime r. Select a base point $G \in E(Z_p)$ of order r. for a randomly selected integer q where $1 \le q \le r-1$, it is extremely difficult to compute q from qG. Given a threshold t and the total number of players $n = 2t + 1$, the proposed ECVSS is as follows:

Secret distribution:

- The dealer chooses a random polynomial of degree t subject to the secret as its free term, $f(x) = \sum_{i=0}^{t} a_i x^i \bmod r$ where a_o is the secret.
- The dealer secretly passes $f(i)$ to player P_i $\forall i = (1,\ldots,n)$.

Generating commitments:

- The dealer publicizes an elliptic curve E defined over Z_p, the prime p and the base point $G \in E(Z_p)$ of order r.
- The dealer broadcasts $a_i G \bmod p \; \forall i = (0,\ldots,t)$.

Verification:

- Each player P_i verifies that $f(i)G = \sum_{j=0}^{t} x_i^j (a_j G) \bmod p \; \forall i = (1,\ldots,n)$.

Secret reconstruction:

- Each player P_i broadcasts $f(i)$.

- Each player P_i verifies that, $f(i)G = \sum_{j=0}^{t} x_i^j (a_j G) \mod p \; \forall i = (1,...,n)$.

It is also possible that the dealer broadcasts a commitment for every distributed share $f(i)$ as $f(i)G \;\; \forall i = (1,...,n)$. If any player sends an accusation against the dealer, the dealer opens the commitments for this player enabling all the other players to decide on the guilty one. The players then vote on either, disqualifying the dealer or the player.

C. *ECVSS without a dealer (JRVSS)*

The players agree on an elliptic curve *E*, the prime *p*, the base point $G \in E(Z_p)$ of order *r*. Given a threshold *t* and the total number of players $n = 2t + 1$, each player P_i do:

- Select a random polynomial $f_i(x)$ of degree *t* subject to his chosen secret $a_0^{(i)}$ as its free term.
- Secretly sends $f_i(j)$ to player $P_j \; \forall j = \{1,...,n\}$.
- Broadcasts $a_k^{(i)} G \;\; \forall k = \{0,...,t\}$.
- Broadcasts $f_i(j)G \;\; \forall j = \{1,...,n\}$.
- Each $P_{j \neq i}$ verifies that $\sum_{k=0}^{t} j^k a_k^{(i)} G \stackrel{?}{=} f_i(j)$ and that $f_i(j)G$ is consistent with his share.
- Each $P_{j \neq i}$ verifies that his share is consistent with other shares i.e., $a_0^{(i)} G = \sum_{j \in B} b_j f_i(j) G$.

The decision on the guilty player is taken according to majority voting. Once the above protocol is completed each player P_i safely calculates his share as $\sum_{j=1}^{n} f_j(i) \mod r$.

D. *Elliptic curve joint zero VSS (JZVSS)*

In the joint random zero secret sharing each player selects a random polynomial of degree t subject to zero as its free term, the protocol is similar to the JRVSS except that the players must validate that the chosen secret by every other player is really zero. Not that $0G = rG = O$ where O is the point at infinity of the elliptic curve. Hence, the difference between the JRVSS and JZVSS is that for any player P_i, all other players check that $a_0^{(i)} G = O \;\; \forall i = \{1,..., n\}$. This check is analogues to that of Feldman's scheme when all that players must consider that $g^{a_0^{(i)}} = 1 \;\; \forall i = \{1,..., n\}$.

V. MULTIPARTY COMPUTATION TOOLS

A. *Simple multiplication protocol*

Given two secrets *x* and *y*, which are both shared among players, compute the product *xy*, while maintaining both x and y secret [21].

Given that *x* and *y* are each shared by a polynomial of degree *t*, each player can locally multiply his shares of *x* and *y* and the result will be automatically a share of *xy* on a polynomial of degree 2*t*. Hence the value of *xy* can be reconstructed from a set of 2*t*+1 correct shares. It is important to state that the resulting polynomial is not completely random which may weaken the security of the scheme. Consequently, the JZSS is employed to add a sort of randomization to the process. More concretely, each player P_i has a share x_i of *x* on a polynomial of degree *t* and a share y_i of *y* on a polynomial of degree *t*. The players run the JZSS scheme such that each player P_i has a valid share z_i of zero on a polynomial of degree 2*t*. Each player P_i locally computes $x_i y_i + z_i$ which represent a valid share of *xy* on a polynomial of degree 2*t*. Extra work have been done in this field to reduce the degree of the 2*t* polynomial back to *t*, however for simplicity we will consider the simple case.

B. *Simple reciprocal protocol*

Given a secret *x* mod *r* which is shared among *n* players, it is required to generate shares of $x^{-1} \mod r$ with out revealing any information about *x* or x^{-1} [15]. The protocol employs the simple multiplication protocol. Initially, each player P_i has a share x_i of *x* on a polynomial of degree *t*. The players run the JRSS protocol, ending up with each player P_i has a share e_i of a random secret *e* on a polynomial of degree *t*. Also, the players run the JZSS such that each player P_i has a share z_i of a zero secret on a polynomial of degree 2*t*. Each player P_i locally computes and broadcasts, $u_i = x_i e_i + z_i$. Players can interpolate the polynomial of degree 2*t* and compute *u*. All the players can compute $u^{-1} \mod r$. Each player P_i computes his share of x^{-1} as $\zeta_i = e_i u^{-1}$ on a polynomial of degree 2*t*.

Of course, when dealing with malicious adversary, ECVSS can be injected in the above protocols to ensure correct dealing of the shares besides the Berlekamp-Welch decoding scheme.

VI. ROBUST ECDSA: EAVESDROPPING AND HALTING ADVERSARY

In this section we will present a robust ECDSA against an *n*/2 eavesdropping and *n*/3 halting adversary. Since it is a requirement for a (*t*, *n*)-secret sharing scheme to hold, that the adversary can attack no more than *t* players; the total number of players is $n = 2t + 1$ for an eavesdropping adversary and $n = 3t + 1$ for a halting adversary. The protocol is as follows:

Public information:

The elliptic curve *E*, the prime *p*, the order *r* and the basic point *G* in addition to the public key *W* are all public information.

Initialization by the dealer:

The dealer distributes shares for the secret key s where $1 \le s \le r-1$ among the players using a random polynomial of degree t. Hence each player P_i is assigned a share s_i of s.

Signature generation:

- Players run the JRSS scheme to share a random value k modulo r on a polynomial of degree t.
- Each player P_i is assigned a share k_i of k. P_i computes: $b_i = \prod_{j \in B, j \neq i} j/(j-i)$ and sets $y_i = b_i k_i$, $V_i = y_i G$ and broadcasts V_i. Notice that it is infeasible to predict y_i from G and V_i.
- Each player computes $(x_V, y_V) = \sum_{i \in B} V_i$ and computes $c = x_V \bmod r$.
- The Players run the reciprocal protocol on the shares of k to compute shares of $k^{-1} \bmod r$ on a polynomial of degree $2t$.
- Let ζ_i be the share given to P_i for $k^{-1} \bmod r$. Players run the multiplication protocol to compute shares of sk^{-1} since each P_i holds a share s_i of s and a share ζ_i of k^{-1}.
- Finally each player P_i is holding a share x_i of $x = k^{-1}(m + sc) \bmod r$.
- Each player P_i broadcasts his x_i (at least $2t+1$ players) so that it is now possible to compute x by interpolation.

Signature verification:

Since the process of sharing the signature is transparent to the verifier, the signature verification process is similar to that if there is a one person signer.

VII. Robust ECDSA: malicious adversary

It is known from error correcting codes theory that if one evaluates a polynomial f of degree m over n different points x_i for i = 1, ... , n, then given the sequence $f(x_i)$ one can reconstruct the coefficients of f in polynomial time even if up to t elements in the sequence are in error provided that $n > m + 2t$ [22]. Since in our proposal described below it is required to interpolate a polynomial of degree $2t$ then we have $n = 2t + 2t + 1 = 4t + 1$. Hence the protocol described below can face an $n/4$ malicious adversary. The signature protocol is as follows:

Public information:

The elliptic curve E, the prime p, the order r and the basic point G in addition to the public key W are all public information.

Initialization by the dealer:

- The dealer distributes shares for the secret key s where $1 \le s \le r-1$ among the players using a random polynomial of degree t. Hence each player P_i is assigned a share s_i of s.
- The dealer also provides commitments to these shares by applying ECVSS. These commitments will be used to ensure honest behavior of the players later on.

Signature generation:

- The players run the JRVSS scheme to share a random value $k \bmod r$ on a polynomial of degree t.
- Each player P_i is assigned a share k_i of k. Notice that $k_i = \sum_{j=1}^{n} k_i^{(j)}$ where $k_i^{(j)}$ is the sub-share submitted to player P_i from player P_j during the execution of the JRVSS.
- P_i now computes $V_i^{(j)} = k_i^{(j)} G$ and broadcasts $V_i^{(j)}$ $\forall j = (1,...,n)$.
- Since $k_i^{(j)}$ is known to P_j, each player P_j can check the validity of $V_i^{(j)}$.
- P_i computes $y_i = b_i k_i$ then $V_i = y_i G$ and broadcasts V_i. Note that any player can compute any of the b_i's. Hence, each player can verify that $V_i = b_i \sum_{j=1}^{n} V_i^{(j)}$. If everything goes ok then the players accept the point V_i.
- Once the players agree on the valid points they safely compute $(x_V, y_V) = \sum_{i \in B} V_i$ and then $c = x_V \bmod r$.
- The Players run the reciprocal protocol on the shares of k to compute shares of $k^{-1} \bmod r$ on a polynomial of degree $2t$. The interpolation is performed using Berlekamp-Welch fault tolerant decoding scheme. As a result, each player P_i has a share ζ_i of $k^{-1} \bmod r$.
- Players run the multiplication protocol to compute shares of sk^{-1} since each P_i holds a share s_i of s and a share ζ_i of k^{-1}.
- Each player P_i broadcasts his x_i (at least $2t+1$ players) so that it is now possible to compute x by Berlekamp-Welch interpolation.

Signature verification: As in the case of one person signer

VIII. A tolerability improvement

The issue here is a complexity – tolerability tradeoff, it is possible to increase the number of cheating players that can be tolerable during the execution of the protocol if one can afford the increase in complexity. The classical results of [23, 24] states that secure multiparty computations are possible against an $n/2$ passive and an $n/3$ active adversary. Hence the scheme described in the previous section is not optimum from the point of view of the number of cheating players that can be tolerated; however, it is low in complexity.

A way to improve on the fault tolerance is to avoid Berlekamp-Welch decoding, but to use some other way to detect and discard incorrect shares. This is indeed possible and it brings the fault tolerance to $(n-1)/3$ as one still needs

$2t$+1 good shares. The gain in fault tolerance however comes at the expenses of an increased amount of computation required from the players in order to compute a single signature. Let a be the secret shared among the n players on a polynomial $A(x)$ of degree t. Each player P_i has a share a_i of a and a_iG for $i = (0,..., n)$ are public knowledge. The players share a secret b using the JRVSS protocol such that each player P_i is assigned a share b_i of b on a polynomial of degree t. Notice that $b_i = \sum_{j=1}^{n} b_i^{(j)}$, where $b_i^{(j)}$ is the sub-share submitted to player P_i from player P_j. The players also run the JZVSS protocol such that each player P_i is assigned a share z_i of zero on a polynomial of degree $2t$ subject to zero as its free term. Each player P_i broadcasts $a_i b_i^{(j)} G \ \forall j = (1,..., n)$ and $a_i b_i G$. Each player P_j can verify the validity of $a_i b_i^{(j)} G$ from $b_i^{(j)}$ and a_iG. Also P_j can verify that $a_i b_i G = \sum_{j=1}^{n} a_i b_i^{(j)} G$. Players are able to sieve bad shares by this method and the number of players required can drop down to 3t+1; however, the complexity is high.

IX. Conclusions

This paper is the first to deal with robust threshold ECDSA bringing the attention to its excellent one way function; it's compatibility with shamir's secret sharing scheme and its impact on providing efficient verifiable secret sharing scheme. We were able to employ the EC one way function to come up with a new VSS scheme based on a different type of mathematics which is more efficient than the Feldman's and Pedersen's schemes that both dramatically fails to protect EC secret keys regarding the size of the public parameters required which destroys one of the merits of the elliptic curve cryptosystem. Then we employed the ECVSS scheme to propose a robust threshold ECDSA to face an $n/2$ eavesdropping, $n/3$ halting and $n/4$ malicious adversary. The tolerability against malicious adversary can be improved to $n/3$ with the expense of higher complexity.

X. References

[1] P. Feldman. A practical scheme for non-interactive veri able secret sharing. *Proc. of the 28th IEEE Symp. on the Found. of Computer Science. IEEE Press*, pp. 427-437 (1987).

[2] T. P. Pedersen. Non-interactive and information-theoretic secure veriable secret sharing. *In J. Feigenbaum, editor, Proceedings CRYPTO '91*, pages 129-140. Springer, 1992. Lecture Notes in Computer Science No. 576.

[3] N. Koblitz, An Elliptic Curve Implementation of the Finite Field Digital Signature Algorithm. *In: Advances in Cryptology -- Crypto '98. Lecture Notes in Computer Science*, Vol. 1462. Springer-Verlag, Berlin Heidelberg New York (1998) 327—337

[4] A. J. Menezes, T. Okamoto, and S. A. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, *IEEE Trans. Inform. Theory, 39 (1993*), pp. 1639--1646

[5] R. Schroeppel, H. Orman, and S. O'Malley. Fast key exchange with elliptic curve systems. *Technical Report 95-03, Department of Computer Science, University of Arizona*, Feb. 1995.

[6] G. B. Agnew, R. C. Mullin, and S. A. Vanstone, "An Implementation of Elliptic Curve Cryptosystems over F 2^155 ", *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 5, June, 1993.

[7] N. Koblitz, Elliptic curve cryptosystems, Math. Comp. 48, pp. 203--209, 1987

[8] V.S. Miller, Use of Elliptic Curve in Cryptography, *In Advances in CryptologyCRYPTO '85*(Santa Barbara, Calif.,1985), LNCS.218, *Spring-Verlag*, pp.417-426, 1986.

[9] A. Shamir. How to Share a Secret. *Communications of the ACM, 22(11),* pp. 612--613, November 1979.

[10] Y. Desmedt and Y. Frankel, Shared generation of authenticators and signatures," *In Advances in Cryptology|CRYPTO '91*, volume 576 of Lecture Notes in Computer Science, pages 457-469, Springer-Verlag, 1992.

[11] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. How to share a function securely. *In 26th Annual ACM Symposium on Theory of Computing*, pages 522--533, 1994.

[12] M. Cerecedo, T. Matsumoto, and H. Imai. Efficient and secure multiparty generation of digital signatures based on discrete logarithm. *IEICE Trans. on Fund. Electr. Comm. and Comp. Sci.,* E76-- A(4):532--545, 1993.

[13] L. Harn, "Group-oriented (t,n) threshold digital signature scheme and digital multisignature ", *IEE Proc.-Comput. Digit. Tech.*, Vol. 141, No. 5, September, (1994), pp. 307 -- 313.

[14] Langford, Threshold DSS Signatures without a Trusted Party, *Proc. CRYPTO'95, LNCS 963, Springer-Verlag*, 1995.

[15] R. Gennaro, Theory and Practice of Verifiable Secret Sharing. *PhD thesis,* Massachusetts Institute of Technology (MIT), May 1996.

[16] R. Gennaro, M. O. Rabin, and T. Rabin. Simplified VSS and fasttrack multiparty computations with applications to threshold cryptography. *In Proc. 17th ACM Symposium on Principles of Distributed Computing (PODC),* 1998.

[17] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, "Robust Threshold DSS Signatures ", *Advances in Cryptology: Proc. Eurocrypt'96, Lecture Notes in Computer Science 1070, Springer*, (1996), pp. 354-371.

[18] A. Boldyreva, "Threshold Signature, Multisignature and Blind Signature Schemes Based on the Gap-Diffie-Hellman-group Signature Scheme", *Public Key Cryptography - PKC 2003*.

[19] T. El Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Trans. Inform. Theory* 31:469-472, 1985.

[20] C. P. Schnorr, Efficient signature generation by smart cards, *Journal of Cryptology* 4:161-174, 1991.

[21] J. Bar-Ilan, and D. Beaver, "Non-Cryptographic Fault-Tolerant Computing in a Constant Number of Rounds", *Proc. of 8th PODC*, pp. 201--209, 1989.

[22] E. Berlekamp and L. Welch. Error Correction of Algebraic Block Codes. US Patent Number 4,633,470.

[23] M. Ben-Or, S. Goldwasser and A. Wigderson, " Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation", *20th STOC*, pp. 1-10, 1988.

[24] T. Rabin and M. Ben-Or. "Verifiable secret sharing and multiparty protocols with honest majority". *In Proc. 21st ACM Symposium on Theory of Computing*, pages 73--85, 1989.

[25] A.Herzberg, S.Jarecki, H.Krawczyk, M.Yung, "Proactive secret sharing or: How to cope with perpetual leakage", *LNCS 963, Proc. Crypto'95, Springer Verlag*, (1995), pp. 339--352.

[26] J.C. Benaloh, "Secret sharing homomorphisms: Keeping a secret secret"; *Proc. of Crypto'86, Lecture Notes on Comput. Sci., 263, Springer Verlag* (1986) 251-260.

[27] J.C. Benaloh, "Secret sharing homomorphisms: Keeping a secret secret"; *Proc. of Crypto'86, Lecture Notes on Comput. Sci., 263, Springer Verlag* (1986) 251-260.