The ISC Int'l Journal of Information Security

July 2020, Volume 12, Number 2 (pp. 91–99)

http://www.isecure-journal.org

A Fast Publicly Verifiable Secret Sharing Scheme using Non-homogeneous Linear Recursions

Ali Zaghian [1,*], and Bagher Bagherpour [2]

[1] *Department of Applied Mathematics and Cryptography, Malek Ashtar University of Technology, Isfahan, Iran*

[2] *Department of Applied Mathematics and Cryptography, Malek Ashtar University of Technology, Isfahan, Iran*

ARTICLE INFO.

Article history:
Received: 21 December 2019
Revised: 7 April 2020
Accepted: 9 July 2020
Published Online: 11 July 2020

Keywords:
Cryptography
Non-homogeneous Linear Recursions
Publicly Verifiable
Secret Sharing Schemes
Threshold Access Structures
Verifiable Secret Sharing

ABSTRACT

A non-interactive (t, n)-publicly verifiable secret sharing scheme (non-interactive (t, n)-PVSS scheme) is a (t, n)-secret sharing scheme in which anyone, not only the participants of the scheme, can verify the correctness of the produced shares without interacting with the dealer and participants. The (t, n)-PVSS schemes have found a lot of applications in cryptography because they are suitable for real-life scenarios in which an external verifier is required to check the correctness of the produced shares without interacting with the dealer and participants. In this paper, we propose a non-interactive (t, n)-PVSS scheme using the non-homogeneous linear recursions (NHLRs), and prove its security with a formal method. We compare the computational complexity of our scheme with that of Schoenmakers's scheme and show that our non-interactive (t, n)-PVSS scheme runs faster than Schoenmakers's scheme when $n \geq 5$ and $n \geq t \geq \lceil \frac{2n+9}{n} \rceil$. The communicational complexity of our scheme is almost equal to that of Schoenmakers's scheme.

1 Introduction

A (t, n)-secret sharing scheme is a method of sharing a secret among a set of participants P, where $|P| = n$, in such a way that only subsets of the participants with at least t elements (so called, *qualified subsets*) are able to reconstruct the secret by pooling the received shares and subsets of the participants with at most $t-1$ elements (so called, *unqualified subsets*) are unable to obtain the secret. The collection of all qualified subsets of P is called the *access structure*. In general, access structures have the *monotone increasing* property, that is, every superset of a qualified subset is a qualified subset. For the first time, the secret sharing schemes were introduced by Shamir [21] and Blakley [3]. The introduced schemes by Shamir and Blakley are (t, n)-secret sharing schemes also they have the unconditional security, that is, the participants of the unqualified subsets cannot obtain any information about the secret in an information theoretical sense. Secret sharing schemes have found a lot of applications in cryptography since they were introduced. For example, in the distributed public key cryptography, key-escrow cryptosystems [18], electronic cash schemes [24], electronic voting schemes [22] and archive systems [12] secret sharing schemes have applications.

1.1 Publicly Verifiable Secret Sharing Schemes

In secret sharing schemes like Shamir's and Blakley's, the dealer and participants are assumed to be honest,

* Corresponding author.
Email addresses: a_zaghian@mut-es.ac.ir, bagher.bagherpour@mut-es.ac.ir
ISSN: 2008-2045

while the dealer and participants may be malicious in real applications. In other words, in real-life scenarios a malicious dealer may give incorrect shares to the participants and the malicious participants may submit incorrect shares during reconstructing the secret. To address this drawback of the secret sharing schemes, verifiable secret sharing schemes (VSS schemes) have been proposed. In non-interactive VSS schemes, the dealer publishes a witness in addition to producing the shares and any participant can verify the correctness of his share using the published witness. In such schemes, qualified subsets of participants recover the same secret using the received shares. For the first time, Chor *et al.* introduced the VSS schemes [5]. In [8], Feldman proposed a non-interactive VSS scheme. Pedersen proposed an efficient non-interactive VSS scheme whose security is based on the hardness of solving the discrete logarithm problem [19]. In [6, 7, 13, 16], the authors proposed verifiable multi-secret sharing schemes (A verifiable multi-secret sharing scheme (VMSS scheme) is a VSS scheme in which the dealer distributes several secrets among the participants and one share is kept by each of the participants).

Although, VSS schemes and VMSS schemes are secure against a malicious dealer and malicious participants, however in such schemes only the participants of the schemes can verify the correctness of the received shares while sometimes it is required that anyone, not only the participants of the scheme, is able to verify the correctness of the produced shares. For example, consider an external verifier that is going to check the correctness of the shares of participants without interacting with the dealer and participants of a VSS scheme. To address this drawback of the VSS schemes, publicly verifiable secret sharing schemes (PVSS schemes) have been proposed. In the non-interactive (t, n)-PVSS scheme, the dealer distributes a secret among a set of participants P, where $|P|= n$, such that any subset of P with at least t participants can recover the secret. As well as, the dealer publishes a proof string $Proof_D$ to show that the shares have correctly been produced. Anyone can verify the correctness of the shares using $Proof_D$. Suppose the participants of $A \subset P$, where $|A|\geq t$, are going to recover the secret in the non-interactive (t, n)-PVSS scheme. To this end, each participant $p \in A$ publishes a proof string $Proof_p$ to show that his share is correct and anyone can verify the correctness of the share of p using $Proof_p$. If each participant of A submits a correct share, then the participants of A recover the secret using their shares. Since communication over private channels is not publicly verifiable, in PVSS schemes the dealer has to use public channels to send data for the participants, and therefore the security of the PVSS scheme is computational at most.

For the first time, Stadler introduced PVSS schemes [24]. Jhanwar *et al.* proposed (t, n)-PVSS schemes in [14, 15]. Other PVSS schemes have been proposed using the bilinear pairings in [11, 15, 17, 26]. Also, Gan *et al.* proposed a PVSS scheme whose security is based on the decisional bilinear Diffie-Hellman assumption [10]. In [22], Schoenmakers proposed a simple and efficient non-interactive (t, n)-PVSS scheme using the Sigma protocols. Schoenmakers's scheme is the most efficient non-interactive (t, n)-PVSS scheme which has been proposed by now.

1.2 Motivation and Contribution

Schoenmakers's scheme is more efficient than other non-interactive (t, n)-PVSS schemes [14, 22], however the dealer of Schoenmakers's scheme must perform $nt-n+1$ group-exponentiations to produce the shares and $3n+t$ group-exponentiations to produce the proof string $Proof_D$. Also, $n(2t+1)$ group-exponentiations must be performed to verify the correctness of the produced shares in Schoenmakers's scheme. In this study, we propose a non-interactive (t, n)-PVSS scheme using the NHLRs and we prove its security with a formal method (see Scheme 1). Compared with Schoenmakers's scheme, our scheme has the following advantages:

- The dealer of our scheme performs $n + 1$ group-exponentiations to produce the shares.
- The dealer of our scheme performs $3n+1$ group-exponentiations to produce the proof string $Proof_D$.
- In our scheme, one can verify the correctness of the produced shares using $n(3 + t) + 1$ group-exponentiations.
- Our scheme runs faster than Schoenmakers's scheme when $n \geq 5$ and $n \geq t \geq \lceil \frac{2n+9}{n} \rceil$.
- In our scheme, the participants of the qualified subsets can use a faster algorithm to reconstruct the secret.

The non-interactive (t, n)-PVSS schemes have been used in designing cryptographic schemes such as threshold software key escrow schemes, revocable electronic cash schemes and electronic voting schemes [22]. Using our results, it can be said that our non-interactive (t, n)-PVSS scheme can be used in designing efficient cryptographic schemes for real-life scenarios.

1.3 Paper Structure

Section 2 presents some notations and the required background on non-interactive (t, n)-PVSS schemes. In Section 3, we present our non-interactive (t, n)-PVSS scheme and analyze its security and computational complexity. Section 4 concludes the paper.

2 Materials and Methods

In this paper, we will use the following notations:

- $[a, b] \Longrightarrow$ the set of integers $\{a, a+1, \ldots, b\}$.
- $a||b \Longrightarrow$ the bit concatenation of two positive integers a and b.
- $|a| \Longrightarrow$ the bit length of an integer a.
- $a' \in_R [a, b] \Longrightarrow$ a uniformly random choosing from $[a, b]$.
- $a \oplus b \Longrightarrow$ the bitwise xor of integers a and b.
- $|A| \Longrightarrow$ the number of elements of a set A.

A function $f(n) : (.) \longmapsto \mathcal{R}^+$ is called a negligible function if for every constant $c > 0$ there exists a constant $n_c > 0$ such that $f(n) < \frac{1}{n^c}$ for all $n > n_c$, where $\mathcal{R}^+$ is the set of positive real numbers.
Suppose p and q are two primes such that $p = 2q + 1$. Also, suppose g is a random generator of the multiplicative group Z_p^*. Decisional Deffie-Hellman assumption (DDH assumption) states that given g^α, g^β and g^γ, where $\alpha, \beta, \gamma \in_R [0, q-1]$, it is infeasible to determine whether $g^\gamma = g^{\alpha\beta}$ or $g^\gamma \neq g^{\alpha\beta}$.
Suppose $P = \{p_1, \ldots, p_n\}$ is a set of participants and $A = \{p_{i_1}, \ldots, p_{i_s}\} \subset P$. The index set of A, denoted by $\Upsilon(A)$, is the set of integers $\{i_1, \ldots, i_s\}$.

2.1 Non-homogeneous Linear Recursions

Let t be a positive integer and $c_0, \ldots, c_{t-1}, a_1, \ldots, a_t \in \mathcal{R}$. A non-homogeneous linear recursion (NHLR) of degree t is defined by the following relation:

$$NHLR : \begin{cases} u_0 = c_0, u_1 = c_1, \ldots, u_{t-1} = c_{t-1}, \\ u_{i+t} + a_1 u_{i+t-1} + \cdots + a_t u_i = f(i), \end{cases}$$

where $i \geq 0$, $c_0, \ldots, c_{t-1}$ and $a_1, \ldots, a_t$ are constants, and $f(i)$ is an arbitrary function in i . Suppose the sequence (u_i) is defined by a NHLR as:

$$NHLR : \begin{cases} u_0 = c_0, u_1 = c_1, \ldots, u_{t-1} = c_{t-1}, \\ \sum_{j=0}^{t} \binom{t}{j} u_{i+t-j} = (-1)^i ci, \ i \geq 0 \end{cases}$$

where $c_0, \ldots, c_t$ and c are constants. Then, $u_i = p(i)(-1)^i$, where $p(i)$ is a polynomial of the form $A_0 + A_1 i + \ldots + A_{t+1} i^{t+1}$. For more details about NHLRs see [2, 6].

2.2 Definition of the Non-interactive (t, n)-PVSS Schemes

Suppose $P = \{p_1, \ldots, p_n\}$ is a set of participants and $t \leq n$ is an integer. A non-interactive (t, n)-PVSS scheme consists of the following phases:

(1) **Initialization phase (Ini-phase):** In this phase, the dealer obtains system parameters SP on input (λ, P, t), where $\lambda \in \mathbb{N}$ is the security parameter. Also, each $p_i \in P$ chooses his private key sk_i and registers his public key pk_i.

(2) **Secret distribution phase (SD-phase):** In this phase, the dealer first produces the encrypted shares $(\sigma_1, \ldots, \sigma_n)$ using the Dis-algorithm.
Dis-algorithm: Take as input $K, SP, (pk_1, \ldots, pk_n)$. Then output the encrypted shares $(\sigma_1, \ldots, \sigma_n)$.
The dealer proves the correctness of the encrypted shares by running the Proof-algorithm.
Proof-algorithm: Take as input $SP, (pk_1, \ldots, pk_n), (\sigma_1, \ldots, \sigma_n)$ and the shares $(s_1, \ldots, s_n)$. Then output the proof string $Proof_D$.
After producing $(\sigma_1, \ldots, \sigma_n)$ and $Proof_D$, the dealer publishes them.
Anyone can verify the correctness of the published shares using the Ver-algorithm.
Ver-algorithm: Take as input $SP, (pk_1, \ldots, pk_n), (\sigma_1, \ldots, \sigma_n)$, and the proof string $Proof_D$. Then output 1 if the dealer has correctly produced the shares, otherwise output $(0, C)$, where C is the set of participants whose shares are inconsistent.

(3) **Decryption phase (Dec-phase):** In this phase, each participant $p_i \in P$ obtains his share s_i using his private key sk_i.

(4) **Secret reconstruction phase (SR-phase):** Suppose $A \subseteq P$ is a set of participants with at least t elements whose participants are going to compute the secret. At the first, each $p_i \in A$ produces a proof string $Proof_i$ by running the SRProof-algorithm.
SRProof-algorithm: Take as input SP, sk_i, pk_i, σ_i, and s_i. Then output a proof string $Proof_i$ to show the correctness of the share s_i.
Anyone can verify the correctness of the share of each participant $p_i \in A$ by running the SRVer-algorithm.
SRVer-algorithm: Take as input SP, pk_i, σ_i, s_i, and $Proof_i$. Then output 1 if s_i is the correct share of $p_i \in A$, otherwise output 0.
For each $p_i \in A$ if the SRVer-algorithm outputs 1, then the participants of A recover the secret K by running the Rec-algorithm, otherwise they abort the scheme.
Rec-algorithm: Take as input SP, and the shares of the participants of A, then output the secret K.

A non-interactive (t, n)-PVSS scheme Σ with computational security must satisfy the following security requirements:
Correctness: If the dealer honestly follows the SD-phase of Σ, then the Ver-algorithm outputs 1 with probability 1. If each $p_i \in P$ honestly follows the Dec-

phase and the SRProof-algorithm of Σ, then anyone accepts the share of p_i with probability 1 using the SRVer-algorithm. Furthermore, every qualified subset $A \subseteq P$ obtains the secret if dealer follows honestly the SD-phase of Σ and the participants of A honestly follow the Dec-phase and the SR-phase of Σ.

Soundness: If the Ver-algorithm outputs 1, then for every qualified subsets $B_1, B_2 \subset P$ of honest participants the following property holds: if K_1 and K_2 are the reconstructed secrets by B_1 and B_2, respectively, then $K_1 = K_2$ with high probability. Also, the SRVer-algorithm outputs 0 with high probability, if a malicious participant submits a fake share during the SR-phase of the scheme.

Privacy: The privacy of the scheme Σ is defined by the following game, say $\mathcal{G}$, between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

(1) $\mathcal{A}$ publishes a set of participants $P = \{p_1, \ldots, p_n\}$ and a threshold $t \leq n$;
(2) $\mathcal{C}$ runs the Ini-phase and produces the security parameters SP. Also, $\mathcal{C}$ chooses private keys and public keys for all participants, then sends SP and all public keys to $\mathcal{A}$;
(3) $\mathcal{A}$ publishes $B \subset P$ of the corrupted participants such that $|B| \leq t-1$;
(4) $\mathcal{C}$ sends the private keys of the participants of B to $\mathcal{A}$;
(5) $\mathcal{A}$ chooses a secret K and the secret distribution oracle sends n valid encrypted shares of the secret K to $\mathcal{A}$. Also, the secret distribution oracle sends the proof string of the shares of secret K to $\mathcal{A}$. This step can be repeated a polynomial number of times;
(6) $\mathcal{A}$ chooses two different secrets K_0 and K_1 and sends them to $\mathcal{C}$;
(7) $\mathcal{C}$ chooses $\tau \in_R \{0,1\}$, executes SD-phase on the secret K_τ and sends $(\sigma_1, \ldots, \sigma_n)$ and $Proof_D$ to $\mathcal{A}$;
(8) $\mathcal{A}$ can repeat the step 5 a polynomial number of times;
(9) Finally, $\mathcal{A}$ outputs a bit τ' and wins if $\tau = \tau'$.

The privacy property (or indistinguishably of the secrets) holds for the non-interactive (t,n)-PVSS scheme Σ if $\mathrm{Adv}^{\mathcal{G},\mathcal{A}} = |\Pr[\tau = \tau'] - \frac{1}{2}|$ is a negligible function in λ, for every polynomial-time adversary $\mathcal{A}$ [14].

2.3 Proving Equality of Discrete Logarithms

Suppose p and q are two primes, with $p = 2q+1$, and Z_p^* is a multiplicative group. In [1], the authors proposed a sigma protocol to prove that two group elements $y_1, y_2 \in Z_p^*$ satisfy the relation $y_1 = g_1^x, y_2 = g_2^x$ (see the following scheme).

Scheme DLogEq(g_1, g_2, y_1, y_2)

(1) The prover chooses $r \in_R [0, q-1]$ and computes $a = (a_1, a_2) = (g_1^r, g_2^r)$. Then, the prover sends a to the verifier;
(2) The verifier chooses $c \in_R [0, q-1]$ and sends c to the prover.
(3) The prover computes $b = (r+c)x^{-1} \mod q$ and sends b to the verifier.
(4) The verifier accepts the prover if $y_1^b = a_1 g_1^c$ and $y_2^b = a_2 g_2^c$.

The DLogEq(g_1, g_2, y_1, y_2) can be made non-interactive using the Fiat-Shamir method [9] as follows: The prover chooses $r \in_R [0, q-1]$ and computes $a_1 = g_1^r, a_2 = g_2^r, c = H(g_1||g_2||y_1||y_2||a_1||a_2)$ and $b = (r+c)x^{-1} \mod q$, where H is a collision resistant hash function. The prover sends (b, c) to the verifier. The verifier accepts the prover if $c = H(g_1||g_2||y_1||y_2||y_1^b g_1^{-c}||y_2^b g_2^{-c})$.

3 Results and Discussions

In this section, we propose a new non-interactive (t,n)-PVSS scheme using the NHLRs (see Scheme 1). In Scheme 1, we use $P = \{p_0, \ldots, p_{n-1}\}$ to denote the set of participants. All exponentiations and multiplications are performed in a multiplicative group Z_p^* of prime order q or in module q.

Scheme 1: A new non-interactive (t,n)-PVSS scheme using NHLRs.

Ini-phase: Dealer chooses two primes p and q such that $p = 2q+1$, $|q| \geq \lambda$, and computing discrete logarithms in the multiplicative group Z_p^* is infeasible. Also, dealer chooses two random generators $g, G \in Z_p^*$ and values $c_0, \ldots, c_{t-1}, c \in_R [0, q-1]$. Each participant $p_i \in P$ chooses a private key $x_i \in_R [0, q-1]$ and publishes $y_i = G^{x_i}$ as his public key. Dealer publishes $SP = (\lambda, P, t, Z_p^*, g, G, c, H)$ as the system parameters, where $H : \{0,1\}^* \to \{0,1\}^\lambda$ is a cryptographic secure hash function and $\lambda \in \mathbb{N}$ is the security parameter. Suppose $a_j = \binom{t}{j}$ for each $j \in [1, t]$.

SD-phase: To distribute the secret $K \in Z_p^*$, dealer first constitutes the following NHLR:

$$\begin{cases} u_0 = c_0, u_1 = c_1, \ldots, u_{t-1} = c_{t-1}, \\ u_{i+t} + a_1 u_{i+t-1} + \cdots + a_t u_i = ci(-1)^i \mod q, \end{cases}$$

where $i \geq 0$. After constituting the NHLR, dealer executes the Dis-algorithm.

$$Dis: \begin{cases} \text{input: } K, SP, \{y_i\}_{i\in[0,n-1]}, \{u_i\}_{i\in[0,n]}; \\ \text{compute } D = G^{u_n} K; \\ \text{compute } Y_i = y_i^{u_i}, \text{ for each } i \in [0, n-1]; \\ \text{output: } (D, \{Y_i\}_{i\in[0,n-1]}) \end{cases}$$

Now, dealer executes the Proof-algorithm (see below) and obtains the proof string. Dealer publishes

the proof string $(\{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}, \{X_i\}_{i\in[0,n+2]})$, the encrypted shares $\{Y_i\}_{i\in[0,n-1]}$ and the values D, S_{n+1} and S_{n+2}.
Anyone can verify the correctness of the produced shares using the Ver-algorithm (see below).
Dec-phase: If the produced shares are correct, then each participant $p_i \in P$ can compute his share (i.e., G^{u_i}) as $S_i = (Y_i)^{x_i^{-1}}$. Otherwise, the participants abort the scheme.
SR-phase: Suppose $A \subseteq P$ is a qualified subset of participants, i.e., $|A| \geq t$. Each $p_i \in A$ produces a proof string using the SRProof-algorithm.

$$\text{SRProof}: \begin{cases} \text{input: } SP, y_i, Y_i, S_i, x_i; \\ \text{choose } v_i \in_R [0, q-1]; \\ \text{compute} \\ w_i = H(G||S_i||y_i||Y_i||G^{v_i}||S_i^{v_i}); \\ \text{compute } z_i = (w_i + v_i)x_i^{-1} \mod q; \\ \text{output } (w_i, z_i). \end{cases}$$

Each participant $p_i \in A$ submits (w_i, z_i) and S_i. Anyone can check the correctness of the share S_i using the SRVer-algorithm:

$$\text{SRVer}: \begin{cases} \text{input: } SP, y_i, Y_i, S_i, w_i, z_i; \\ \text{compute} \\ w_i' = H(G||S_i||y_i||Y_i||y_i^{z_i}G^{-w_i}||Y_i^{z_i}S_i^{-w_i}); \\ \text{output 1 if } w_i = w_i', \text{ otherwise output 0.} \end{cases}$$

If for each $p_i \in A$ the SRVer-algorithm outputs 1, then the participants of A can recover the secret using one of the following methods, otherwise they abort the scheme:

$$\text{Rec1}: \begin{cases} \text{input: } SP, A, \{S_i\}_{i\in\Upsilon(A)}, S_{n+1}, S_{n+2}, D; \\ \text{for each } i \in \Upsilon(A) \cup \{n+1, n+2\} \text{ compute} \\ \lambda_i = \prod_{\{j\in\Upsilon(A)\cup\{n+1,n+2\}, j\neq i\}} \frac{n-j}{i-j} \mod q; \\ \text{for each } i \in \Upsilon(A) \cup \{n+1, n+2\} \text{ compute} \\ S_i' = (S_i)^{(-1)^{-i}}; \\ \text{compute } E' = \prod_{\{i\in\Upsilon(A)\cup\{n+1,n+2\}\}} (S_i')^{\lambda_i}; \\ \text{compute } E = (E')^{(-1)^n}; \\ \text{output } K = DE^{-1} \text{ as the secret;} \end{cases}$$

If A is a sequential qualified subset (i.e., $A = \{p_i, \ldots, p_{i+t-1}\}$, where $i \in [0, n-t]$), then the participants of A can compute the secret using the following algorithm:

$$\text{Rec2}: \begin{cases} \text{input: } SP, A, \{S_j\}_{j\in[i,i+t-1]}, D; \\ \text{for each } j \in [i, i+t-1] \text{ set } G^{u_j} = S_j; \\ \text{compute } G^{u_n} \text{ using the following relation :} \\ G^{u_j} = G^{((-1)^{j-t}(j-t)c)} \prod_{k=1}^{t} (G^{u_{j-k}})^{-a_k}, \\ \text{where } j \in [i+t, n]; \\ \text{output } K = D(G^{u_n})^{-1} \text{ as the secret;} \end{cases}$$

Remark 1. Note that the SRProof-algorithm of Scheme 1 uses the non-interactive version of the protocol DLogEq to show that $Y_i = S_i^{x_i}$ and $y_i = G^{x_i}$, where x_i is the private key of the participant $p_i \in A$. Also, in the SRVer-algorithm of Scheme 1, if $w_i' = w_i$ for some $p_i \in A$, then $Y_i = S_i^{x_i}$ and $y_i = G^{x_i}$. This shows that S_i is the correct share of the participant $p_i \in A$.

Algorithm 1 Proof-algorithm

1: Input: $SP, \{y_i\}_{i\in[0,n-1]}, \{Y_i\}_{i\in[0,n-1]}, \{u_i\}_{i\in[0,n+2]}$;
2: For each $i \in [0, n+2]$ compute $X_i = g^{u_i}$;
3: Compute $S_{n+1} = G^{u_{n+1}}$ and $S_{n+2} = G^{u_{n+2}}$;
4: For each $i \in [0, n+2] \setminus \{n\}$ choose $r_i \in_R [0, q-1]$;
5: For each $i \in [0, n-1]$ compute $\zeta_i = H(g||y_i||X_i||Y_i||g^{r_i}||y_i^{r_i})$;
6: For each $i \in [n+1, n+2]$ compute $\zeta_i = H(g||G||X_i||S_i||g^{r_i}||G^{r_i})$;
7: Compute $\zeta = \zeta_0 \oplus \cdots \oplus \zeta_{n-1} \oplus \zeta_{n+1} \oplus \zeta_{n+2}$;
8: For each $i \in [0, n+2] \setminus \{n\}$ compute $b_i = (r_i + \zeta)u_i^{-1} \mod q$;
9: Output $(\{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}, \{X_i\}_{i\in[0,n+2]}, S_{n+1}, S_{n+2})$.

Remark 2. The Proof-algorithm uses the non-interactive version of the protocol DLogEq to show that $Y_i = y_i^{u_i}$ and $X_i = g^{u_i}$ for all $i \in [0, n-1]$. As well as, $S_i = G^{u_i}$ and $X_i = g^{u_i}$ for all $i \in [n+1, n+2]$.

Remark 3. In the Ver-algorithm, if $\zeta_j' = \zeta_j$ for some $j \in [0, n+2] \setminus \{n\}$, then $Y_j = y_j^{u_j}$ and $X_j'' = g^{u_j}$. Therefore, u_j satisfies the relation $u_t + a_1 u_{t-1} + \cdots + a_{t-j}u_j + \cdots + a_t u_0 = 0 \mod q$ if $j \in [0, t-1]$. Also, u_j satisfies the relation $u_j + a_1 u_{j-1} + \cdots + a_t u_{j-t} = (-1)^{j-t}(j-t)c \mod q$ if $j \in [t, n+2] \setminus \{n\}$. This shows that the share of the participant $p_j \in P$ has correctly been produced by the dealer.

3.1 Security Analysis of Scheme 1

Here, we prove that our non-interactive (t, n)-PVSS scheme satisfies the correctness, soundness and privacy properties of the (t, n)-PVSS schemes.
Theorem 1. *Scheme 1 satisfies the correctness property of the non-interactive (t, n)-PVSS schemes.*

Proof. Consider the Ver-algorithm. If the dealer follows honestly the Ini-phase and the SD-phase of

Algorithm 2 Ver-algorithm

1: Input: $SP, \{y_i\}_{i\in[0,n-1]}, \{Y_i\}_{i\in[0,n-1]}, \{X_i\}_{i\in[0,n+2]}, S_{n+1}, S_{n+2}, \{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}$;
2: Set $C = \emptyset$;
3: For each $i \in [0, t-1]$ compute $X'_i = X_t \prod_{j\in[0,t-1],j\neq i} X_j^{a_{t-j}}$;
4: For each $i \in [0, t-1]$ compute $X''_i = (X'_i)^{(-a_{t-i})^{-1}}$;
5: For each $i \in [t, n+2]\setminus\{n\}$ compute $X''_i = (\prod_{j=1}^{t} X_{i-j}^{-a_j}) g^{\alpha_{i-t}}$, where $\alpha_{i-t} = (-1)^{i-t}(i-t)c$;
6: Compute $\zeta = \zeta_0 \oplus \cdots \oplus \zeta_{n-1} \oplus \zeta_{n+1} \oplus \zeta_{n+2}$;
7: For each $i \in [0, n-1]$ compute $\zeta'_i = H(g||y_i||X''_i||Y_i||(X''_i)^{b_i} g^{-\zeta}||Y_i^{b_i} y_i^{-\zeta})$;
8: For each $i \in [n+1, n+2]$ compute $\zeta'_i = H(g||G||X''_i||S_i||(X''_i)^{b_i} g^{-\zeta}||S_i^{b_i} G^{-\zeta})$;
9: For each $i \in [0, n-1]\cup\{n+1, n+2\}$ compute $C = C \cup \{i\}$ if $\zeta'_i \neq \zeta_i$;
10: Output 1 (i.e., the shares are correct) if $C = \emptyset$;
11: Otherwise output $(0, C)$ (i.e., the dealer produced incorrect shares for elements of C).

Scheme 1, then for each $p_i \in P$ it holds $Y_i = y_i^{u_i}$. Also, for each $i \in [0, n+2]\setminus\{n\}$, it holds $X''_i = g^{u_i}$. Therefore, the Ver-algorithm outputs 1. In the SR-phase, if the participant $p_i \in A$ follows honestly the Dec-phase and SRProof-algorithm, then $S_i^{x_i} = Y_i$ and $y_i = G^{x_i}$. Therefore, the SRVer-algorithm outputs 1. Now, suppose $A \subset P$ is a qualified subset of the participants. We prove that the participants of A can compute the secret by executing the Rec1-algorithm. To this end, it suffices to show that $\prod_{\{i\in\Upsilon(A)\cup\{n+1,n+2\}\}} (S'_i)^{\lambda_i} = G^{p(n)}$. Using the properties of the NHLRs, we know that there exists a polynomial $p(x)$ of degree $t+1$ such that $u_i = p(i)(-1)^i \mod q$, for each $i \in \Upsilon(A)\cup\{n+1, n+2\}$. Therefore,

$$\sum_{\{i\in\Upsilon(A)\cup\{n+1,n+2\}\}} \lambda_i u_i (-1)^{-i} = p(n) \mod q. \quad (1)$$

Using Relation 1, $S'_i = (S_i)^{(-1)^{-i}}$, and $S_i = G^{u_i}$, where $i \in \Upsilon(A)\cup\{n+1, n+2\}$, we have

$$\prod_{\{i\in\Upsilon(A)\cup\{n+1,n+2\}\}} (S'_i)^{\lambda_i} = G^{\sum_{\{i\in\Upsilon(A)\cup\{n+1,n+2\}\}} \lambda_i u_i (-1)^{-i}} = G^{p(n)}. \quad (2)$$

Hence, the participants of A can compute the secret K by executing the Rec1-algorithm. It can easily be verified that if A is a sequential qualified subset, then the participants of A can compute the secret K by executing the Rec2-algorithm. □

Theorem 2. *Scheme 1 satisfies the soundness property of the non-interactive (t, n)-PVSS schemes.*

Proof. Suppose $B_1, B_2 \subset P$ are two qualified subsets of participants. Since the Ver-algorithm outputs 1, for each $p_i \in P$ we have $X''_i = g^{u_i}$, and therefore

$$u_{i+t} + a_1 u_{i+t-1} + \cdots + a_t u_i = ic(-1)^i \mod q$$

for each $i \in [0, n+2-t]$. Suppose K_1 and K_2 are the secrets recovered by the subsets B_1 and B_2, respectively. Using the properties of the NHLRs, we know that there exists a polynomial $p(x)$ of degree $t+1$ such that $u_i = p(i)(-1)^i \mod q$ for each $i \in [0, n+2]$. Therefore,

$$\prod_{\{i:i\in\Upsilon(B_1)\cup\{n+1,n+2\}\}} (S'_i)^{\lambda_i} = \prod_{\{i:i\in\Upsilon(B_2)\cup\{n+1,n+2\}\}} (S'_i)^{\lambda_i}, \quad (3)$$

where $S'_i = (S_i)^{(-1)^{-i}}$ and $S_i = G^{u_i}$. Using the relation 3, it can be said that $K_1 = K_2$. Now, consider the SRVer-algorithm. Suppose $p_i \in P$ is a malicious participant that submits a fake share S_f during the SR-phase. Then $S_f^{x_i} \neq Y_i$, and therefore $w'_i \neq w_i$ with high probability. □

At the follows, we prove that Scheme 1 satisfies the privacy property of the non-interactive (t, n)-PVSS schemes. We will use a reduction and show that if an attacker can break the privacy of Scheme 1, then it can be used to break the DDH assumption. The heart of the reduction is a simulator $\mathcal{B}$. Given an instance of the DDH assumption as input, $\mathcal{B}$ acts as the challenger of $\mathcal{G}$ and plays $\mathcal{G}$ with an attacker that is able to break the privacy of Scheme 1. Then, $\mathcal{B}$ uses the attacker to break the DDH assumption.

Theorem 3. *Scheme 1 satisfies the privacy property of the non-interactive (t, n)-PVSS schemes.*

Proof. Suppose $\mathcal{A}$ is an attacker that can break the privacy of Scheme 1. Suppose p and q are two primes, with $p = 2q+1$, and g is a random generator of the multiplicative group Z_p^*. Given g^α, g^β and g^γ, where $\alpha, \beta, \gamma \in_R [0, q-1]$, $\mathcal{B}$ tries to determine whether $g^\gamma = g^{\alpha\beta}$ or $g^\gamma \neq g^{\alpha\beta}$. $\mathcal{B}$ plays the game $\mathcal{G}$ with $\mathcal{A}$ as follows:

The step 1 of $\mathcal{G}$: $\mathcal{A}$ publishes $P = \{p_0, \ldots, p_{n-1}\}$ as the set of participants and an integer $t \leq n$ as a threshold.

The step 2 of $\mathcal{G}$: Using the Ini-phase of Scheme 1, the simulator $\mathcal{B}$ obtains $SP = (\lambda, P, t, Z_p^*, g, G, c, H)$, where $P = \{p_0, \ldots, p_{n-1}\}$ is the set of participants and $G = g^\alpha$. For each $p_i \in P$, $\mathcal{B}$ assigns a private key $x_i \in_R [0, q-1]$ and computes $y_i = G^{x_i}$ as his public key. $\mathcal{B}$ then sends SP and $\{y_i\}_{i=0}^{n-1}$ to $\mathcal{A}$.

The step 3 of $\mathcal{G}$: $\mathcal{A}$ publishes a subset of the corrupted participants $B \subset P$ such that $|B| \leq t-1$. Without loss of the generality suppose $B = \{p_{n-t}, \ldots, p_{n-1}\}$.

The step 4 of $\mathcal{G}$: $\mathcal{B}$ sends the private keys $\{x_i\}_{i=n-t}^{n-1}$ to $\mathcal{A}$.

The step 5 of $\mathcal{G}$: $\mathcal{A}$ chooses a secret $K \in Z_p^*$ and gives it to the secret distribution oracle. The secret

distribution oracle gives K to $\mathcal{B}$ and $\mathcal{B}$ follows the following procedure:

(1) for each $i \in [n-t, n+2] \setminus \{n\}$, choose $c_i \in_R [0, q-1]$ and set $u_i = c_i$;
(2) for each $i \in [n-t, n+2] \setminus \{n\}$, compute $G_i = G^{u_i}$ and set $G_n = g^{\gamma}$;
(3) compute $\lambda_{i,j} = \prod_{e \in [n-t,n+2]\setminus\{i\}} \frac{j-e}{i-e} \bmod q$, for each $i \in [n-t, n+2]$ and $j \in [0, n-t-1]$;
(4) for each $j \in [0, n-t-1]$, compute $G'_j = \prod_{i \in [n-t,n+2]} (G_i)^{(-1)^{-i}\lambda_{i,j}}$;
(5) compute $G_j = (G'_j)^{(-1)^j}$, for each $j \in [0, n-t-1]$;
(6) for each $k \in [0, n-1]$ compute $Y_k = (G_k)^{x_k}$. Furthermore, set $S_{n+1} = G^{u_{n+1}}$ and $S_{n+2} = G^{u_{n+2}}$;
(7) for each $i \in [n-t, n+2] \setminus \{n\}$, compute $X_i = g^{u_i}$ and set $X_n = g^{\beta}$;
(8) for each $j \in [0, n-t-1]$, compute $X'_j = \prod_{i \in [n-t,n+2]} (X_i)^{(-1)^{-i}\lambda_{i,j}}$;
(9) for each $j \in [0, n-t-1]$, compute $X_j = (X'_j)^{(-1)^j}$.

$\mathcal{B}$ chooses $b_0, \ldots, b_{n-1}, b_{n+1}, b_{n+2}, \zeta_0, \ldots, \zeta_{n-1}, \zeta_{n+1}, \zeta_{n+2} \in_R [0, q-1]$. For each $i \in [0, n-1]$, $\mathcal{B}$ defines

$$\zeta_i = H(g||y_i||X_i||Y_i||(X_i)^{b_i} g^{-\zeta}||(Y_i)^{b_i} y_i^{-\zeta}),$$

where $\zeta = \zeta_0 \oplus \cdots \oplus \zeta_{n-1} \oplus \zeta_{n+1} \oplus \zeta_{n+2}$. For each $i \in [n+1, n+2]$, $\mathcal{B}$ also defines

$$\zeta_i = H(g||G||X_i||S_i||(X_i)^{b_i} g^{-\zeta}||(S_i)^{b_i} G^{-\zeta}).$$

Then, $\mathcal{B}$ saves the defined values in the H-Table. If $\mathcal{A}$ requests from the random oracle a hash query of a value e and $H(e)$ has not been previously defined, then $\mathcal{B}$ chooses a random value $e' \in [0, q-1]$ and defines $H(e) = e'$. $\mathcal{B}$ then saves $H(e) = e'$ in the H-Table and the random oracle returns e' to $\mathcal{A}$. If $H(e)$ has previously been defined, then the random oracle returns e', where $H(e) = e'$ has beforehand been saved in the H-Table.
Now, $\mathcal{B}$ computes $D = KG_n$ and sends the following values to the secret distribution oracle,

$$(\{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}, \{X_i\}_{i\in[0,n+2]}), \{Y_i\}_{i\in[0,n-1]}, D, S_{n+1}, S_{n+2}.$$

The secret distribution oracle sends them to $\mathcal{A}$. This step can be repeated a polynomial number of times.
The step 6 of $\mathcal{G}$: $\mathcal{A}$ chooses two different secrets $K_0, K_1 \in Z_p^*$ and sends them to $\mathcal{B}$.
The step 7 of $\mathcal{G}$: $\mathcal{B}$ chooses $\tau \in_R \{0, 1\}$. Like the step 5, $\mathcal{B}$ obtains a new proof string $(\{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}, \{X_i\}_{i\in[0,n+2]})$, new encrypted shares $\{Y_i\}_{i=0}^{n-1}$, and new values S_{n+1} and S_{n+2}. Also, $\mathcal{B}$ computes $D_\tau = K_\tau G_n$. Then, $\mathcal{B}$ sends the following values to $\mathcal{A}$,

$$(\{\zeta_i\}_{i\in[0,n+2]\setminus\{n\}}, \{b_i\}_{i\in[0,n+2]\setminus\{n\}}, \{X_i\}_{i\in[0,n+2]}), \{Y_i\}_{i=0}^{n-1}, D_\tau, S_{n+1}, S_{n+2}.$$

The step 8 of $\mathcal{G}$: $\mathcal{A}$ can repeat the step 5 a polynomial number of times and asks the proof string and the encrypted shares of different secrets form the secret distribution oracle.
The step 9 of $\mathcal{G}$: Finally, $\mathcal{A}$ outputs a bit τ'.
It can easily be verified that if $g^{\gamma} = g^{\alpha\beta}$, then $u_n = \beta$ and $\mathcal{A}$ will guess K_τ with probability greater than $\frac{1}{2} + \eta(\lambda)$, where $\eta(\lambda)$ is a non-negligible function. If $g^{\gamma} \neq g^{\alpha\beta}$, then u_n is a random value from $[0, q-1]$ and $\mathcal{A}$ can guess K_τ with probability $\frac{1}{2}$. Therefore, if $\tau = \tau'$ then $\mathcal{B}$ decides $g^{\gamma} = g^{\alpha\beta}$, otherwise $\mathcal{B}$ decides $g^{\gamma} \neq g^{\alpha\beta}$. This shows that $\mathcal{B}$ can break the DDH assumption if $\mathcal{B}$ has access to $\mathcal{A}$, and this contradicts the DDH assumption. Hence, Scheme 1 satisfies the privacy property of the non-interactive (t, n)-PVSS schemes. □

3.2 The Computational Complexity of Scheme 1

In this paper, the computational complexity of the schemes is measured according to the number of group-exponentiations required for running the schemes (one group exponentiation is denoted by t_{ex}), because the computational cost of other cryptographic functions, such as group-multiplication, hash function and group-inversion, is negligible compared with heavier computational cost of the group exponentiation. In Table 1, we mention the computational complexity of Scheme 1, Schoenmakers's scheme and Jhanwar's scheme (note that Schoenmakers's scheme and Jhanwar's scheme are the most efficient non-interactive (t, n)-PVSS scheme which have been proposed by now [14]). Using Table 1, it can be said that:

(1) The computational complexity of the Dis-algorithm in Scheme 1 (in Schoenmakers's scheme) is equal to $(n+1)t_{ex}$ ($(nt-n+1)t_{ex}$, respectively). Therefore, the Dis-algorithm of Scheme 1 runs almost $t-1$ times faster than that of Schoenmakers's scheme. Also, the Dis-algorithm of Scheme 1 runs $(t+3)$ times faster than the Dis-algorithm of Jhanwar's scheme.
(2) The computational complexity of the Proof-algorithm in Scheme 1 (in Schoenmakers's scheme and Jhanwar's scheme) is equal to $(3n+9)t_{ex}$ ($(3n+t)t_{ex}$, respectively). Therefore, for large n and $t \approx n$, the Proof-algorithm of Scheme 1 runs 1.33 times faster than that of Schoenmakers's/Jhanwar's scheme.
(3) The computational complexity of the Ver-algorithm in Scheme 1 (in Schoenmakers's scheme and Jhanwar's scheme) is equal to $(n(3+t)+t+6)t_{ex}$ ($n(2t+1)t_{ex}$, respectively).

Table 1. Computation cost of the scheme 1, Schoenmakers's scheme [22] and Jhanwar's scheme [14]

	Scheme 1	[22]	[14]
Dis-algorithm	$(n+1)t_{ex}$	$(nt-n+1)t_{ex}$	$(n(t+3)+2)t_{ex}$
Proof-algorithm	$(3n+9)t_{ex}$	$(3n+t)t_{ex}$	$(3n+t)t_{ex}$
Ver-algorithm	$(n(3+t)+t+6)t_{ex}$	$n(2t+1)t_{ex}$	$n(2t+1)t_{ex}$
Dec-phase	$1t_{ex}$	$1t_{ex}$	$1t_{ex}$
SRProof-algorithm	$2t_{ex}$	$2t_{ex}$	$2t_{ex}$
SRVer-algorithm	$4t_{ex}$	$4t_{ex}$	$4t_{ex}$
Rec1-algorithm	$(t+3)t_{ex}$	tt_{ex}	tt_{ex}
Rec2-algorithm	$(n-t+1)(t+1)t_{ex}$	-	-

Therefore, for large n, the Ver-algorithm of Scheme 1 runs almost twice faster than that of Schoenmakers's/Jhanwar's scheme.

(4) The computational complexity of the Dec-phase, SRProof-algorithm and SRVer-algorithm in Scheme 1 is the same as the computational complexity of the Dec-phase, SRProof-algorithm and SRVer-algorithm in Schoenmakers's scheme and Jhanwar's scheme.

(5) In Scheme 1 (in Schoenmakers's scheme and Jhanwar's scheme), the computational complexity of the Rec1-algorithm is equal to $(t+3)t_{ex}$ (tt_{ex}, respectively). Note that in the Rec1-algorithm of Scheme 1 the value $S_i' = (S_i)^{(-1)^{-i}}$, for $i \in \Upsilon(A) \cup \{n+1, n+2\}$, can be computed before running the Rec1-algorithm.

(6) Since the Rec2-algorithm needs fewer group-multiplications than the Rec1-algorithm, the participants of the sequential qualified subsets can use the Rec2-algorithm to recover the secret swiftly when $n \approx t$. Schoenmakers's scheme and Jhanwar's scheme do not have this property.

(7) In a nutshell, the computational complexity of Scheme 1 is equal to $(n(t+7)+2t+26)t_{ex}$, the computational complexity of Schoenmakers's scheme is equal to $(n(3t+3)+2t+8)t_{ex}$, and the computational complexity of Jhanwar's scheme is equal to $(n(3t+7)+2t+9)t_{ex}$.

(8) For $n \geq 5$ and $n \geq t \geq \lceil \frac{2n+9}{n} \rceil$, Scheme 1 runs faster than Schoenmakers's scheme. Also, scheme 1 runs faster than Jhanwar's scheme when $n \geq 3$ and $n \geq t \geq \lceil \frac{17}{2n} \rceil$.

3.3 The Communicational Complexity of Scheme 1

In this paper, the communicational complexity of schemes is measured according to the number of values published during running schemes. In Table 2, we state the storage and communicational complexity of our scheme, Schoenmakers's scheme and Jhanwar's scheme. Using Table 2, it can be said that the communicational complexity of our scheme is almost equal to that of Schoenmakers's scheme. Furthermore, the communicational complexity of our scheme is less than that of Jhanwar's scheme.

Table 2. Storage and communicational complexity of scheme 1, Schoenmakers's scheme [22] and Jhanwar's scheme [14]

	Scheme 1	[22]	[14]
Communicational complexity	$5n+7$	$5n+1$	$6n+t+2$
Storage for each participant	two integers	two integers	three integers

4 Conclusion

In this paper, we presented a non-interactive (t, n)-PVSS scheme using the NHLRs, and proved its security with a formal method. In our scheme, the participants of the sequential qualified subsets can recover the secret swiftly using the Rec2-algorithm, while Schoenmakers's scheme and Jhanwar's scheme do not have this property. For $n \geq 5$ and $n \geq t \geq \lceil \frac{2n+9}{n} \rceil$, our scheme runs faster than Schoenmakers's scheme. Also, our scheme runs faster than Jhanwar's scheme when $n \geq 3$ and $n \geq t \geq \lceil \frac{17}{2n} \rceil$. Since the non-interactive (t, n)-PVSS schemes are used in a lot of cryptographic schemes, using our (t, n)-PVSS scheme one can design efficient schemes for real-life scenarios.

Proactive secret sharing schemes are important cryptographic schemes that are used to save sensitive and long-lived data [12, 25]. Such schemes use Shamir's method to distribute data and to recover them. Our results show that NHLRs can easily be applied to proactive secret sharing schemes for designing efficient proactive secret sharing schemes with verification capability.

Acknowledgment

The authors would like to thank the anonymous reviewers whose valuable suggestions increased the readability and quality of the current paper.

References

[1] Bagher Bagherpour, Ali Zaghian and Mahdi Sajadieh, Sigma protocols for faster proof of simultaneous homomorphism relations, *IET information security*, pages 508–514, 2019.

[2] Norman L Biggs, Discrete Mathematics, revised ed, Oxford University Press, New York, 1989.

[3] George R Blakley, Safeguarding cryptographic keys, *International Workshop on Managing Requirements Knowledge*, pages 313–317, 1979.

[4] David Chaum and Torben P Pedersen, Wallet databases with observers, *In: Advances in Cryptology–CRYPTO 92, Lecture Notes in Computer Science*, vol. 740, pages 89–105, Springer, Berlin, 1992.

[5] Benny Chor, Shafi Goldwasser, Silvio Micali and Baruch Awerbuch, Verifiable secret sharing and

achieving simultaneity in the presence of faults, *FOCS' 85, IEEE Computer Society, Washington*, pages 383–395, 1985.
[6] Massoud H Dehkordi and Samaneh Mashhadi, New efficient and practical verifiable multi-secret sharing schemes, *Information Science*, 178, pages 2262–2274, 2008.
[7] Massoud H Dehkordi and Samaneh Mashhadi, An efficient threshold verifiable multi-secret sharing, *Computer Standards and Interfaces*, 30, pages 187–190, 2008.
[8] Paul Feldman, A practical scheme for non-interactive verifiable secret sharing, *FOCS' 87, IEEE computer society, Washington*, pages 427–437, 1987.
[9] Amos Fiat and Adi Shamir, How to prove yourself: practical solutions to identification and signature problems, *Advances in Cryptology-CRYPTO'86, Lecture notes in compute science*, 263, pages 186–194, Springer, Berlin, 1986.
[10] Yuanju Gan, Lihua Wang, Ping Pan and Yixian Yang, publicly verifiable secret sharing scheme with provable security against chosen secret attacks, *International journal of distributed sensor and networks*, pages 1–9, 2013.
[11] Somayeh Heidarvand and Jorge L Villar, Public verifiability of pairings in secret sharing schemes, *SAC 2008*, pages 294–308, 2009.
[12] Amir Herzberg, Stanislaw Jarecki, Hugo Krawczyk and Moti Yung, Proactive secret sharing or How to cope with perpetual leakage, *CRYPTO'95, LNCS 963*, pages 339–359, Springer-Verlag, 1995.
[13] Chunqiang Hu, Xiaofeng Liao and Xiuzhen Cheng, Verifiable multi-secret sharing based on LFSR sequences, *Theoretical Computer Science*, 445, pages 52–62, 2012.
[14] Mahabir P Jhanwar, Ayineedi Venkateswarlu and Reihaneh Safavi-Naini, Paillier-based publicly verifiable (non-interactive) secret sharing, *Designs, codes and Cryptography*, 18 March 2014.
[15] Mahabir P Jhanwar, A practical (non-interactive) publicly verifiable secret sharing scheme, *LNCS 6672*, pages 273–287, 2011.
[16] Yanhong Liu, Futai Zhang and Jie Zhang, Attacks to some verifiable multi-secret sharing schemes and two improved schemes, *Information Science*, 329, pages 524–539, 2016.
[17] Samaneh Mashhadi, Secure publicly verifiable and proactive secret sharing schemes with general access structures, *Information Science*, 378, pages 99–108, 2017.
[18] Silvio Micali, Fair public-key cryptosystems, *Advances in Cryptology-CRYPTO 1992, Lecture Notes in Computer Science*, vol. 740, pages 113–138, Springer, Berlin, 1993.
[19] Torben P Pedersen, Non-interactive and Information-Theoretic Secure Verifiable secret sharing, *Advances in Cryptology-CRYPTO'91* , pages 129–140, 1992.
[20] Alexandre Ruiz and Jorge L Villar, Publicly verifiable secret sharing from paillier's cryptosystem, *WEWORC 2005, LNI p-74*, pages 98–108, 2005.
[21] Adi Shamir, How to share a secret, *Communications of the ACM*, 22, pages 612–613. 1979.
[22] Berry Schoenmakers, A simple publicly verifiable secret sharing scheme and its application to electronic voting, *Advances in Cryptology-CRYPTO'99, Lecture Notes in Computer Science*, pages 148–164, vol. 1666, Springer, Berlin, 1999.
[23] Jun Shao and Zhenfu Cao, A new efficient (t, n) verifiable multi-secret sharing scheme based on YCH scheme, *Applied Mathematics and Computation*, 168, pages 135–140, 2005.
[24] Markus Stadler, Publicly verifiable secret sharing, *Advances in Cryptology-EUROCRYPT'96, Lecture Notes in Computer Science*, pages 190–199, Springer, Berlin, 1996.
[25] Theodore M Wong, Chenxi Wang and Jeannette M Wing, Verifiable secret redistribution for archive systems, *Proceedings of the first International IEEE security in storage workshop (SISW'02)*, 2003.
[26] Tsu Y Wu and Yuh M Tseng, A pairing-based publicly verifiable secret sharing scheme, *Journal of Systems Science and Complexity*, 24, pages 186–194, 2011.

Ali Zaghian Ali Zaghian was born in Isfahan, Iran in 1959. He received his Ph.D. degree in cryptography-mathematics from Tarbiat Moalem University, Tehran, Iran, in 2008. He is currently associate professor at department of Mathematics and Cryptography in Malek-Ashtar University of Technology (MUT), Isfahan, Iran. His research interests include Coding Theory and Cryptography Algorithms.

Bagher Bagherpour Bagher Bagherpour was born in 1990 in Tabriz. He received the M.Sc. degree in Cryptography from Malek Ashtar University of Technology (MUT), Isfahan, Iran, in 2015. He is currently a Ph.D. student in MUT. His research interests include Symmetric-key cryptography, Secret sharing and Lattice-based cryptography.