Applying General Access Structure to Proactive Secret Sharing Schemes

Ventzislav Nikov[1], Svetla Nikova[2] ⋆, Bart Preneel[2], and Joos Vandewalle[2]

[1] Department of Mathematics and Informatics,
Veliko Tarnovo University,
5000 Veliko Tarnovo, Bulgaria
vnikov@mail.com
[2] Department Electrical Engineering, ESAT/COSIC,
Katholieke Universiteit Leuven, Kasteelpark Arenberg 10,
B-3001 Heverlee-Leuven, Belgium
svetla.nikova, bart.preneel, joos.vandewalle@esat.kuleuven.ac.be

Abstract. Verifiable secret sharing schemes (VSS) are secret sharing schemes (SSS) dealing with possible cheating by participants. In this paper we use the VSS proposed by Cramer, Damgard and Maurer [6, 7, 5]. They introduced a purely linear algebraic method to transform monotone span program (MSP) based secret sharing schemes into VSS. In fact, the monotone span program model of Karchmer and Wigderson [13] deals with arbitrary monotone access structures and not just threshold ones. Stinson and Wei [16] proposed a proactive SSS based on threshold (polynomial) VSS. The purpose of this paper is to build unconditionally secure proactive SSS over any access structure, as long as it admits a linear secret sharing scheme (LSSS).

1 Introduction

Proactive security for secret sharing was first suggested by Ostrovski and Yung in [14], where they presented, among other things, a proactive polynomial secret sharing scheme. The proposed polynomial secret sharing proactive scheme in [14] uses the verifiable secret sharing scheme of [15]. Proactive security refers to security and availability in the presence of a mobile adversary. Herzberg et al. [11] further specialized this notion to robust secret sharing schemes and gave a detailed efficient proactive secret sharing scheme. "Robust" means that in any time period, the shareholders can reconstruct the secret value correctly. There are also many papers that discuss proactive security, see e.g. the references in [15, 14, 11, 16].

We call the groups who are allowed to reconstruct the secret *qualified*, and the groups who should not be able to obtain the information about the secret *forbidden*. The collection of all qualified groups is denoted by Γ, and the collection

⋆ The author was partially supported by NATO research fellowship and Concerted Research Action GOA-MEFISTO-666 of the Flemish Government.

of all forbidden groups is denoted by Δ. In fact Γ is *monotone increasing* and Δ is *monotone decreasing*. The tuple (Γ, Δ) is called *access structure* if $\Gamma \cap \Delta = \emptyset$. If $\Gamma \cup \Delta = 2^P$, where P is the set of participants, then we say that (Γ, Δ) is *complete* and we denote it by Γ. Otherwise we say that (Γ, Δ) is *incomplete*. By Γ^- we denote the collection of *minimal sets* of Γ and by Δ^+ we denote the collection of *maximal sets* of Δ. It is obvious that the (Γ^-, Δ^+) generate the (Γ, Δ). We will consider general monotone access structure (Γ, Δ), which describes subsets of participants that are qualified to recover the secret $s \in K$ (K - finite field) in the set of possible secret values.

There exists an adversary A which can *corrupt* a set of servers during any time period. Corrupting a server means learning the secret information in the server, modifying its data, sending out wrong message, and so on. Since the server can be rebooted, the adversary is a mobile one. The collection of all possible corrupted servers for fixed time period we call *bad* and is denoted by Δ_A, and the collection of all possible uncorrupted servers for the same period of time we call *good* and is denoted by Γ_A. It is obvious that Δ_A and Γ_A are monotone and $\Gamma_A \cap \Delta_A = \emptyset$. So we can consider a second access structure (Γ_A, Δ_A), which is called *adversary access structure* [12]. The adversary access structure is complete, so we will denote it only by Γ_A.

The simple example of adversary access structure is to set a number b to be the maximum number of broken (corrupt) servers by adversary for fixed time frame (i.e. threshold case).

The contribution of this paper is twofold: First, we introduce new operation for the access structures which extends the notion of $Q^2(Q^3)$ adversary structure introduced by Hirt and Maurer [12]. This operation characterize which adversary structure can be tolerated. Also this operation allows us to study how the participants and the adversary structures are linked. Second, we propose a proactive SSS for general access structures for both, participants and adversary, as long as the participants access structure admits a LSSS and some conditions for both access structures hold.

2 Preliminary

2.1 Notations

Let K be finite field. An (n, t)-Vandermonde matrix (over K) with $t < n$, is a matrix whose i-th row is of the form $(1, \alpha_i, ..., \alpha_i^{t-1})$, where $\alpha_1, ..., \alpha_n \in K$. For an arbitrary matrix M over K, with m rows labeled by $1, \ldots, m$ and for arbitrary non-empty subset N of $\{1, \ldots, m\}$, let M_N denote the matrix obtained by keeping only those rows i with $i \in N$. If $\{i\} = N$ we write M_i. Consider the set of row-vectors $v_{i_1}, \ldots, v_{i_k}$ and let $N = \{i_1, \ldots, i_k\}$ be the set of indices, then we denote by v_N the matrix consisting of rows $v_{i_1}, \ldots, v_{i_k}$. Instead of $< \epsilon, v_i >$ for $i \in N$ we will write $< \epsilon, v_N >$. Let M_N^T denote the transpose of M_N, and let ImM_N^T denote the K-linear span of the rows of M_N. We use $KerM_N$ to denote the kernel of M_N, i.e. all linear combinations of the columns of M_N, leading to 0.

It is well known that any square Vandermonde matrix has non-zero determinant. If M is an (n,t)-Vandermonde matrix over K and N is non-empty subset of $\{1,...,n\}$, then the rank of M_N is maximal (i.e. is equal to t, or equivalently, $ImM_N^T = K^t$) if and only if $|N| \geq t$. More over: Let ε denote the column vector $(1,0,...,0) \in K^t$. If $|N| < t$, then $\varepsilon \notin ImM_N^T$, i.e. there is no $\lambda \in K^{|N|}$ such that $M_N^T \lambda = \varepsilon$.

Let us define the standard scalar product $< x,y >$ and $x \perp y$, when $< x,y >= 0$. For a K-linear subspace V of K^t, $V^\perp$ denotes the collection of elements of K^t, that are orthogonal to all of V (the orthogonal complement), which is again a K-linear subspace. For all subspaces V of K^t we have $V = (V^\perp)^\perp$, $(ImM_N^T)^\perp = KerM_N$ or $ImM_N^T = (KerM_N)^\perp$, $< x, M_N^T y >=< M_N x, y >$ Hence from $ImM_N^T = (KerM_N)^\perp$ follows the lemma.

Lemma 1. *[5] The vector $\varepsilon \notin ImM_N^T$ if and only if there exists $\mathbf{k} \in K^t$ such that $M_N \mathbf{k} = 0$ and $\mathbf{k}_1 = 1$.*

Let $v = (v_1, \ldots, v_t) \in K^t$; $w = (w_1, \ldots, w_t) \in K^t$; The tensor product $v \otimes w$ is defined as a matrix $t \times t$ that the j-column is equal to $v_j w$.

2.2 Definition

Now, generalizing the notion of $Q^2(Q^3)$ adversary structure introduced by Hirt and Maurer [12], we introduce a new operation for the access structure.

Definition 1. *For the access structure (Γ, Δ) we define the operation $*$ as follows: $n * \Delta = \{A = A_1 \cup A_2; A_1 \in (n-1) * \Delta, A_2 \in \Delta\}$, for $n = 2, 3, \ldots$.*

Let us consider the tuples $(\Gamma, \Delta), (\Gamma, 2*\Delta), \ldots, (\Gamma, n*\Delta)$. They are access structures if and only if $\Gamma \cap n * \Delta = \emptyset$.

Definition 2. *For the complete access structure Γ we define the operation $*$ as follows: First we set $\Delta = 2^P \setminus \Gamma$ and (as in Definition 1) calculate $n * \Delta$. Then we define $n * \Gamma = 2^P \setminus n * \Delta$, for $n = 2, 3, \ldots$*

Now we can consider the sequence $\Gamma, 2 * \Gamma, \ldots, n * \Gamma$, of access structures if and only if $n * \Gamma \neq \emptyset$, i.e. if $n * \Gamma$ is *non-trivial* one.

Lemma 2. *Let Γ be a complete access structure, then $n * \Gamma \neq \emptyset$ for every n if and only if there exists a $P_i \in P$ such that $\{P_i\} \notin \Delta$.*

2.3 The settings

We will follow the settings of the scheme in [14, 11, 16]. Consider system of n servers $P = \{P_1, P_2, \ldots, P_n\}$, which are connected to a common broadcast channel. We can also assume that the system is synchronized. To make things simpler, we assume that there are private channels between each pair of servers and that the messages sent by broadcast are safely authenticated. With this assumptions, we are able to focus on the proactive scheme itself.

There is also a dealer D who should share the secret $s \in K$ and a mobile adversary A. For the system of servers P, we consider the access structure (Γ, Δ)

of qualified and forbidden groups. Since this access structure is set up in the beginning of the procedures and is not changed during the all life of the system we will call it *static*. On the other hand the adversary access structure Γ_A is dynamic. We prove in this paper that if both access structures fulfill some requirements it is possible to build an unconditionally secure proactive scheme.

3 VSS

Since secret sharing was proposed initially by Shamir [17] and Blakley [2], research on this topic has been extensive. In the "classic" secret sharing schemes, there are assumed to be no faults in the system. Chor et al. [8] first defined the complete notion of VSS. There are two aspects of the security in a VSS. One is the security of the secret and the other is the security of the verification. In [1] it was shown that in any unconditionally secure threshold VSS, $b < n/3$. In [16] Stinson and Wei proposed more efficient unconditionally secure VSS with threshold t and with $b \leq n/4 - 1$.

In this section we provide an unconditionally secure VSS which will be used in the proactive scheme later.

3.1 LSSS and MSP

Brickell [4] points out how the linear algebraic view leads to a natural extension to a wider class of secret sharing schemes that are not necessarily of the threshold type. These have later been generalized to all possible so-called monotone access structures by Krachmer and Wigderson [13] based on a linear algebraic computational device called *monotone span program* (MSP).

Definition 3. *[13, 5] The quadruple $M = (K, M, \varepsilon, \psi)$ is called a monotone span program, where K is a finite field, M is a matrix (with m rows and $d \leq m$ columns) over K and $\psi : \{1, \ldots, m\} \to \{1, \ldots, n\}$ is a surjective function. The size of M is the number of rows m.*

Here ψ labels each row with a number from $[1, \ldots, m]$ corresponding to a fixed player, so we can think of each player as being the "owner" of one or more rows. And for every player we consider a function φ which gives the set rows owned by the player. In some sense φ is the inverse of ψ.

Theorem 1. *[3, 9] MSP is said to compute an access structure (Γ, Δ) if and only if it is the case that:*
a) $\varepsilon \in ImM_N^T$ when N is a member of Γ.
b) $\varepsilon \notin ImM_N^T$ when N is a member of Δ.

A SSS is linear if the dealer and the participants use only linear operations to compute the shares and the secret. Each *linear SSS* (LSSS) can be viewed as derived from a monotone span program computing its access structure. On the other hand, each monotone span program gives rise to an LSSS. Hence, one can identify an LSSS with its underlying monotone span program. Note that the size of M is also the size of the corresponding LSSS. Now we will consider any access structure, as long as it admits a linear secret sharing scheme.

3.2 Definition

Now a formal definition of VSS follows.

Definition 4. *[16] Suppose there are a dealer D and participants $P_1, \ldots, P_n$ connected by private channels. They also have access to broadcast channel. There is a static adversary A, that can corrupt a set of the participants from Δ_A including the dealer D. Here static means that the participants controlled by the adversary are fixed.*
Let π be a protocol, consisting of two phases: **Share** *and* **Reconstruct**.
At the beginning of the **Share** *the Dealer inputs a secret $s \in K$.*
At the end of **Share** *each participant P_i is instructed to output a Boolean value ver_i.*
At the end of **Reconstruct** *each participant is instructed to output a value in K.*

Now we are ready to define unconditionally secure VSS, as follows.

Definition 5. *[16] The protocol π is unconditionally secure Verifiable Secret Sharing protocol if the following properties hold:*

1. *If a good player P_i outputs $ver_i = 0$ at the end of Share then every good player outputs $ver_i = 0$;*
2. *If the dealer D is good, then $ver_i = 1$ for every good P_i;*
3. *If a group of good players P_i output $ver_i = 1$ at the end of Share, then there exists an $s' \in K$ such that the event that all good P_i output s' at the end of Reconstruct is fixed at the end of Share and $s' = s$ if the dealer is good;*
4. *If $|K| = q$ and s is chosen randomly from K, and the dealer is good, then any forbidden coalition cannot guess at the end of Share the value s with probability better than $1/q$.*

One obvious requirement for the access structures is the following.

Lemma 3. *Let the adversary access structure be Γ_A and the considered access structure for participants P be (Γ, Δ). In order to build the VSS based on linear SSS the following conditions should hold:*
i) $\Delta_A \subseteq \Delta$;
ii) $\Gamma \subseteq \Gamma_A$;
i.e. the set of bad servers is a subset of the set of forbidden ones, and the set of qualified servers is a subset of the set of good ones.

3.3 VSS with a dealer

We first state the share phase as follows.

Distribution (Share) Phase Let $s \in K$ be a secret.

1. The dealer D chooses a random symmetric matrix $R \in K^{d,d}$, subject to s in its upper left corner. He sends $v_{\varphi(k)} = M_{\varphi(k)}R$ (the row-vectors) to P_k.
2. After receiving $v_{\varphi(k)}$, each P_k sends $M_{\varphi(e)}v^T_{\varphi(k)}$, to P_e for $1 \leq e \leq n$, $(e \neq k)$.

3. Each P_e checks whether $M_{\varphi(e)}v^T_{\varphi(k)} = v_{\varphi(e)}M^T_{\varphi(k)}$, for $k = 1, \ldots, n, (k \neq e)$. If P_e finds that this is not true then P_e broadcasts accusation to P_k in the form (e, k).

4. Each P_i computes the minimum subset $G \subset \{P_1, \ldots, P_n\}$, such that any ordered pair $(e, k) \in G \times G$ is not broadcasted. If $G \in \Gamma_A$, then P_i outputs $ver_i = 1$ otherwise P_i outputs $ver_i = 0$.

It is obvious that every good participant computes the same subset G at the end of *Share*. Next we consider the reconstruction phase. Note that although the adversary is static, he could provide correct information in *Share* phase but wrong information in *Reconstruction* phase. It means that the adversary access structure in reconstruction phase is $2 * \Gamma_A$.

Reconstruction Phase 1. Each player P_i sends the $< \varepsilon^T, v_{\varphi(i)} >$ to P_k, where $i, k \in G$.

2. After receiving the information, P_k computes λ, such that $M^T_{\varphi(\tilde{G})}\lambda = \varepsilon$, for some group $\tilde{G} \subset G$ and $\tilde{G} \in 2 * \Gamma_A$.

3. Denote by R_1 the first column in R. So, $s =< R_1, \varepsilon >=< R_1, M^T_{\varphi(\tilde{G})}\lambda >= < M_{\varphi(\tilde{G})}R_1, \lambda >=< (M_{\varphi(\tilde{G})}R)_1, \lambda >= < (S_{\varphi(\tilde{G})})_1, \lambda >$, where $(S_{\varphi(\tilde{G})})_1$ is the column-vector of the first coordinates of each share, i.e. $< \varepsilon^T, v_{\varphi(\tilde{G})} >$.

Note that joint information held by the players in G is $S_{\varphi(G)} = M_{\varphi(G)}R$.

We are now in position to prove the following theorem.

Theorem 2. *The scheme of this section is an unconditionally secure verifiable secret sharing scheme if the following condition is satisfied:*

iii) $\Gamma = 2 * \Gamma_A$.

Proof. We prove that the above scheme satisfies the conditions of the VSS as follows.

1. If a good player P_i outputs $ver_i = 0$, then all players in $\tilde{G} \in 2 * \Gamma_A$ will also output $ver_i = 0$.

2. If the dealer is good, then since R is symmetric we have $(M_iR)^T = RM_i^T$ and hence $M_jv_i^T = v_jM_i^T$ holds for all good players P_i, P_j. Thus all good players are in $\tilde{G}$. Therefore $ver_i = 1$ for each good player P_i.

3. Suppose that all players from $G \in \Gamma_A$ output $ver_i = 1$ at the end of the *Share*. Then in G no one complained to the others. Since we assume that there are Δ_A bad players, there are at least $\tilde{G} \in 2 * \Gamma_A$ good players in G. Further because of condition iii) the players in $\tilde{G}$ can determine the secret $s' \in K$. Of course $s' = s$ if the dealer is good.

4. Regarding *privacy* let N be the "rejected" set and let us consider joint information held by the players in N, i.e. $S_{\varphi(N)} = M_{\varphi(N)}R$. Let $u \in K$ be arbitrary and $\mathbf{k}$ satisfy $M_{\varphi(N)}\mathbf{k} = 0$ and $\mathbf{k}_1 = 1$. Then $\mathbf{k} \otimes \mathbf{k}$ is a symmetric matrix, which has 1 in its upper left corner and satisfies $M_{\varphi(N)}(\mathbf{k} \otimes \mathbf{k}) = 0$. This is enough to show that for each possible secret, the number of symmetric matrices with that secret in its upper left corner and consistent with the joint information of N are the same. Consider the equation $M_{\varphi(N)}(R + (u - s)\mathbf{k} \otimes \mathbf{k}) = S_{\varphi(N)}$ and let the first coordinate of the first column vector be equal to u. This means that from

the point of view of the players in N, $S_{\varphi(N)}$ can be consistent with the secret u. The number of $\tilde{R} \in K^{d,d}$ with u in its upper left corner is clearly equal to $|KerM_{\varphi(N)}|$ (which is independent of u) and the players in N have no information about s (here we must take into account that all elements of R, except possibly the first one, have been chosen at random). Note that from iii) it follows that $\Gamma \cap 2 * \Delta_A = \emptyset$. □

3.4 VSS without dealer

Secret sharing without dealer means that there is no dealer in the scheme, who knows and distributes the secret. We can remove the dealer from our scheme as follows. The other properties of the scheme are the same as in the previous subsection.

Distribution Phase 1. Each P_k chooses an independent symmetric matrix $R^{(k)}$ subject to s_k be in the upper left corner. Then P_k sends $v^{(k)}_{\varphi(e)} = M_{\varphi(e)}R^{(k)}$ to P_e through a private channel.

2. After receiving $v^{(k)}_{\varphi(e)}$ each P_e sends $M_{\varphi(j)}(v^{(k)}_{\varphi(e)})^T$ to P_j for $1 \leq j \leq n$ through a private channel.

3. P_j checks whether $M_{\varphi(j)}(v^{(k)}_{\varphi(e)})^T = v^{(k)}_{\varphi(j)}M^T_{\varphi(e)}$ for $1 \leq e \leq n$. If P_j finds that this is not true, then P_j broadcasts accusation $(k; j, e)$.

4. For every $k \neq j$, each player P_j computes the maximum subset $G^{(k)}$, such that for any pair $(j, e) \in G^{(k)} \times G^{(k)}, (k; j, e)$ is not broadcasted. If $G^{(k)} \in \Gamma_A$, then P_j says P_k is a honest dealer and puts the value k in a list G.

5. If $G \in \Gamma_A$, then P_j outputs $ver_j = 1$ and computes his share as $v_{\varphi(j)} = \sum_{e \in G} v^{(e)}_{\varphi(j)}$. Otherwise, P_j refuses the shares and outputs $ver_j = 0$.

Reconstruction Phase This phase is the same as in the previous scheme. Note that in this case the shared secret is $s = \sum_{i \in G} s_i$ In this scheme each player in turn plays the role of the dealer. Thus the security of the scheme follows from the security of previous scheme. We need only to show that each good player has the same list G, which is obvious. So, we obtain a key predistribution scheme without dealer.

4 Proactive scheme

The secret value needs to be maintained for a long period of time. The life time is divided into time periods which are determined by the global clock. At the beginning of each time period the servers engage in an interactive update protocol. The update protocol will not reveal the value of the secret. At the end of the period the servers hold new shares of the secret. We distinguish the following phases in each time period [11]. At the beginning we have *Distribution* or *Recovery*, during the period *Renewal* and at the end *Reconstruct* or *Detection* followed of *Recovery* for the beginning of next period.

It is a common expectation that once we have the concept of proactivity very often it is quite easy to add it on top of an existing distributed protocol as VSS, notwithstanding many known VSS are not easy to adapt for proactive property. All proactive schemes known to the authors are for the threshold case; in this section we propose a scheme applying general access structure.

4.1 Distribution Phase

In the initial step, we assume that there is a dealer to set up the scheme. After the initialization phase the dealer will no longer be needed.

In the initialization, we use the share phase of the VSS described before. The first four steps are the same. Then the last one is as follows:

5. If the set of the servers with output $ver_i = 1$ is from Γ_A, then the dealer D erases all the information about the scheme on his end. Otherwise the dealer reboots the whole system and initializes the system again.

4.2 Share Renewal

In this phase we will use one additional row to the matrix M and denote it by $M_0 = (1, 0, \ldots, 0)$. In the Share Renewal phase, all good servers G from the distribution phase do the following:

1. Each server $P_e \in G$ selects a random symmetric matrix $R^{(e)}$, subject to 0 being in upper left corner.

2. P_e sends $v^{(e)}_{\varphi(k)} = M_{\varphi(k)} R^{(e)}$ to all P_k by a private channel and broadcasts $v^{(e)}_0 = M_0 R^{(e)}$.

3. P_k checks whether $v^{(e)}_{\varphi(k)} M_0^T = M_{\varphi(k)} (v^{(e)}_0)^T$ and $< v^{(e)}_0, \varepsilon^T >= 0$. If the conditions are satisfied, then P_k computes and sends to P_j the values $M_{\varphi(j)} (v^{(e)}_{\varphi(k)})^T$. Otherwise P_k broadcasted an accusation of P_e.

4. P_j checks whether $M_{\varphi(j)} (v^{(e)}_{\varphi(k)})^T = v^{(e)}_{\varphi(j)} M^T_{\varphi(k)}$ for the values of e not accused by some set of servers from $2 * \Gamma_A$ (in step 3). If the set of values of k for which equations are not true is from $2 * \Gamma_A$, then P_j broadcasts an accusation of P_e.

5. If P_e is accused by some set of servers from $2 * \Gamma_A$ (from steps 3 and 4), then he can defend himself as follows. For those P_i that P_e is accused by, P_e broadcasts $v^{(e)}_{\varphi(i)}$. Then all servers P_k check whether $v^{(e)}_{\varphi(k)} M^T_{\varphi(i)} = M_{\varphi(k)} (v^{(e)}_{\varphi(i)})^T$ and broadcast *"yes"* or *"no"*. If the set of servers broadcasting *"yes"* is from $2 * \Gamma_A$, then P_e is not a bad server.

6. P_j updates the list of bad servers L by including all values e for which P_e is accused by at least one set from $2 * \Gamma_A$ or found bad in the previous step. Then P_j updates its shares as $v_j \longleftarrow v_j + \sum_{e \notin L} v^{(e)}_j$.

In this phase the real shares are not involved, so no information about them could be revealed. Secondly, from the protocol we know that every good server should have the same list L. Therefore, the good server will keep consistent shares after renewal.

Note that a good server P_e can be accused by at most $2 * \Delta_A$ servers. In this case, P_e will broadcast its defense. On the other hand, suppose P_e gives to P_i a

wrong share, i.e the received share P_i is not consistent with some set of servers from $2 * \Gamma_A$. Then P_i will accuse P_e in step 4. If P_e broadcasts a correct share in the defense, then P_i can correct his share. Otherwise P_e will be found to be bad.

4.3 Recover a share

When a server is corrupted or replaced, it needs to be rebooted and thus it needs to recover the secret shares.

Detection First we provide a protocol, to detect the corrupted servers, which we call detection.

1. P_e computes and sends $M_{\varphi(k)}v^T_{\varphi(e)}$ to P_k for $k = 1, 2, \ldots, n$ by private channels.

2. P_k checks whether $M_{\varphi(k)}v^T_{\varphi(e)} = v_{\varphi(k)}M^T_{\varphi(e)}$. Then broadcasts an accusation (k, e), which contains those e, such that the equations are not true or $M_{\varphi(k)}v^T_{\varphi(e)}$ was not received.

3. Each good server (e.g. not in L from renewal phase) updates the list L so that it contains those e accused by some set of servers from $3 * \Gamma_A$.

At the end of the detection we have a set of bad servers L and corresponding set of good servers $G = P \setminus L$.

Recovery After running detection in the end of the time period the system will recover the shares for all servers P_e, $e \in L$ which is in fact the beginning of the new time period for the system. The recovery protocol is as follows.

1. For each $e \in L$ the server P_e is rebooted.

2. Every good server P_i $(i \notin L)$ computes and sends $M_{\varphi(e)}v^T_{\varphi(i)}$ to P_e.

3. Upon receiving the data, P_e computes rows $v_{\varphi(e)}$, such that $M_i v^T_{\varphi(k)} = v_i M^T_{\varphi(k)}$, $i \in \varphi(e)$ for some set of indices k from $3 * \Gamma_A$ it received. P_e sets $v_{\varphi(e)}$ as its shares.

Remark: Let us denote by $c_{\varphi(k)} = M_i v^T_{\varphi(k)}$, for $i \in \varphi(e)$ and $k \notin L$. So P_e is searching for vectors (shares) $v_i \in K^d$, $i \in \varphi(e)$ such that the following system of equations holds $v_i M^T_{\varphi(G)} = c_{\varphi(G)}$. Now we conclude that from the pairwise checking protocol and the requirement v) from Theorem 3 such a vector v_i always exists and is unique.

Recall that $c_{\varphi(G)} = M_i v^T_{\varphi(G)} = M_i R M^T_{\varphi(G)}$ hence there exists always a solution v_i of the equations $v_i M^T_{\varphi(G)} = c_{\varphi(G)}$. So, the question is whether the solution is unique. M. van Dijk [10] observes in the proof of Theorem 3.1.2 (p.56) the equality of the following quantities, when $G \in \Gamma$: the row rank of $M_{\varphi(G)}$, the column rank of $M_{\varphi(G)}$ and the dimension of the row space of $M_{\varphi(G)}$. Let us denote by $d_2 = |\varphi(G)|$ the number of rows that the group G uses for their shares. Thus it follows that $d \geq d_2$ and the number of linearly dependent columns is $d - d_2$. Therefore in order to have an unique solution for v_i we should have $d = d_2$.

4.4 Reconstruct the secret

The reconstruction protocol is similar to the reconstruction of VSS given above. We need only to change the first two steps as follows:

1′. For all good servers $P_i, P_k \in G$, P_i sends the value $< \varepsilon^T, v_{\varphi(i)} >$ to P_k.

2′. P_k computes λ, s.t. $M^T_{\varphi(\tilde{G})}\lambda = \varepsilon$, for some group $\tilde{G} \subset G$ and $\tilde{G} \in 3 * \Gamma_A$.

3. As in VSS case the secret $s =< (S_{\tilde{G}})_1, \lambda >$.

We are now in a position to prove the following theorem.

Theorem 3. *The scheme described in this section is an unconditionally secure proactive secret sharing scheme if the following conditions are satisfied:*
iv) $\Gamma = 3 * \Gamma_A$.
v) For each group $N \in \Gamma^-$ *the number of rows for the group* $|\varphi(N)|$ *is equal to the number of columns of matrix* M.

Note that the requirement *v)* is satisfied in the threshold case.
The proof follows from Theorem 2 and from the description of the protocol.

References

1. **M.Ben-Or, S.Goldwasser, A.Wigderson**, Completeness theorems for Non-Cryptographic Fault-Tolerant Distributed Computation, *Proc. ACM STOC'88*, 1988, 1-10.
2. **G.R.Blakley**, Safeguarding cryptographic keys, *AFIPS Conference Proc.* 48, 1979, 313-317.
3. **G.R.Blakley, G.A.Kabatianskii**, Linear Algebra Approach to Secret Sharing Schemes, Springer Verlag LNCS 829, 1994, 33-40.
4. **E.F.Brickell**, Some ideal secret sharing schemes, *J. of Comb. Math. and Comb. Computing* 9, 1989, 105-113.
5. **R.Cramer**, Introduction to Secure Computation, *Secure Computation 2000.*
6. **R.Cramer, I.Damgard, U.Maurer**, General and Efficient Secure Multy-Party Computation from any Linear Secret Sharing Scheme, Weizmann Workshop on Cryptography, Weizmann Institute of Science, Rehovot, Israel, June 1998.
7. **R.Cramer, I.Damgard, U.Maurer**, General Secure Multy-Party Computation from any Linear Secret-Sharing Scheme, *Proc. EUROCRYPT 2000*, Springer Verlag LNCS 1807, 316-335.
8. **B.Chor, S.Goldwasser, S.Micali, B.Awerbuch**, Verifiable secret sharing and achieving simultaneity in the presence of faults, *Proc. of the IEEE 26th Annual Symp. on Foundations of Computer Science* 1985, 383-395.
9. **M.van Dijk**, A Linear Construction of Secret Sharing Schemes, *DCC* 12, 1997, 161-201.
10. **M.van Dijk**, Secret Key Sharing and Secret Key Generation, *Ph.D. thesis*, 1997, TU Eindhoven.
11. **A.Herzberg, S.Jarecki, H.Krawczyk, M.Yung**, Proactive secret sharing or: How to cope with perpetual leakage, *Proc. CRYPTO 1995*, Springer Verlag LNCS 963, 339-352.
12. **M.Hirt, U.Maurer**, Player Simulation and General Adversary Structures in Perfect Multiparty Computation, *J. of Cryptology* 13, 2000, 31-60.

13. **M.Karchmer, A.Wigderson**, On Span Programs, *Proc. of 8-th Annual Structure in Complexity Theory Conference*, San Diego, California, 18-21 May 1993. IEEE Computer Society Press, 102-111.
14. **R.Ostrovsky, M.Yung**, How to withstand mobile virus attack, ACM Symposium on principles of distributed computing, 1991, 51-59.
15. **T.Rabin, M.Ben-Or**, Verifiable secret sharing and multiparty protocols with honest majority, *Proc. of the 21st Annual ACM Symp. on Theory of Computing* 1989, 73-85.
16. **D.R.Stinson, R.Wei**, Unconditionally Secure Proactive Secret Sharing Scheme with combinatorial Structures, *SAC'99*, Springer Verlag LNCS 1758, 200-214.
17. **A.Shamir**, How to share a secret, *Communications of the ACM* 22, 1979, 612-613.