

Efficient Perfectly Secure Verifiable Secret Sharing and Distributed Commitment Schemes

Svetla Nikova¹ and Ventsislav Nikov²

¹ ESAT-COSIC, Katholieke Universiteit Leuven, Belgium and
EEMCS-DIES, University of Twente, The Netherlands

² NXP, Belgium

svetla.nikova@esat.kuleuven.be, venci.nikov@gmail.com

Abstract. This paper deals with perfectly secure *distributed commitment* and *verifiable secret sharing* schemes. We consider *linear secret sharing* schemes induced by *monotone span programs* with size m , as well as non-homomorphic schemes. Our achievements are as follows. We improve the efficiency of several (fundamental) distributed perfectly secure protocols, and generalize some existing protocols by proposing non-homomorphic variants. We also show that an existing scheme possesses the property of public verifiability.

More precisely, we first propose a new linear verifiable secret sharing scheme. We prove that this scheme is perfectly secure and that its communication complexity is $\Omega(m)$ times better than the scheme published by Cramer *et al.* at EC'00. We also present a more efficient general treatment of perfectly secure verifiable secret sharing schemes (not necessarily linear), following the idea of Cramer *et al.* at STOC'00 and generalizing the known schemes. In addition, we design a non-homomorphic *commitment transfer protocol*. Next, we propose a *commitment sharing protocol* with communication complexity $\Omega(m)$ times better than the scheme given by Cramer *et al.* in EC'00. The proposed commitment sharing protocol works even if the underlying secret sharing scheme is non-homomorphic. Finally, we point out that the verifiable secret sharing scheme with reduced round complexity (by Gennaro *et al.* in STOC'01), is in fact, a *publicly verifiable* scheme, to our knowledge this is the first known perfectly secure publicly verifiable secret sharing.

A possible application of our results is as follows as Cramer *et al.* have shown distributed commitment, verifiable secret sharing, commitment transfer protocols, and commitment sharing protocols are the basic building blocks for *commitment multiplication protocol* and *multi-party computation*. As a result of our improvements, the communication complexity of the commitment multiplication protocol and the multi-party computation is improved with a factor of $\Omega(m)$.

Keywords: perfect security, distributed commitments, (publicly) verifiable secret sharing.

1 Introduction

In this paper we consider two related topics, namely perfectly secure *verifiable secret sharing* and *distributed commitments*. In *Secret Sharing Schemes* (SSS) it is assumed that the dealer and the players follow the protocol correctly. When this assumption is dropped we come to the harder problem of *verifiable secret sharing* (VSS) scheme: here, some players (including the dealer) may not follow the protocol and, even worse, may provide incorrect information. Still, the honest players should be able to reconstruct the secret, even without the dealer's help. When such a reconstruction is not possible without the help of the dealer, we get a weaker version, namely *distributed commitments* (DC).

The classical result of Cramer *et al.* [3] is that perfectly secure VSS and DC for general access structure can be efficiently built on top of any *linear secret sharing* (LSS) scheme. This result has been extended by Cramer *et al.* [1] by showing that VSS and DC for general access structure can be efficiently built even on top of any SS scheme. Note that the main goal of the authors of [1] is not only to provide generic constructions, but to make them *efficient* [3]. Cramer *et al.* [3] introduced three auxiliary commitment protocols in order to describe a perfectly secure *multi-party computation* (MPC) solution: *commitment transfer protocol* (CTP), *commitment sharing protocol* (CSP) and *commitment multiplication protocol* (CMP).

Any improvement of the (communication) complexity of the above mentioned protocols (VSS, DC, CTP, CSP) will reduce the overall complexity of many commitment multiplication protocols (CMP) and multi-party computation protocols (MPC), where they are used as basic building blocks. This observation motivates our research. In short, we improve the efficiency of the linear VSS and CSP and the non-linear VSS, CTP and CSP protocols.

Stadler [7] defined the so called *publicly verifiable secret sharing* (PVSS) scheme, in which the definition of VSS is extended such that the sharing should be publicly verifiable. Later Schoenmakers [8] added a requirement that even the reconstruction should also be publicly verifiable. Note that both Stadler and Schoenmakers define and consider public verifiability only for computationally secure schemes. In [6] Gennaro *et al.* have improved the round complexity of VSS. We will show that the protocol of Gennaro *et al.* is in fact the first perfectly secure PVSS.

The outline of this paper is as follows: Sect. 2 introduces the basic notions: access and adversary structure, (linear) secret sharing, monotone span program, verifiable secret sharing, and distributed commitments; it also describes the security model we are working on. In Sect. 3 we review the previous work on perfectly secure DC and auxiliary DC protocols, VSS and PVSS. Our results both on linear and non-linear VSS and DC schemes are presented in Section 4: We suggest a more efficient VSS and CSP in the linear case. Then, continuing the idea from [1] a general treatment of perfectly secure VSS schemes is given. We also show that more efficient VSS, CTP, and CSP could be built even if the underlying SS is non-homomorphic. We pointed out that the VSS protocol with reduced round complexity [6] is in fact PVSS. Then, we conclude in Section 5.

2 Preliminaries

Denote the participants of a *secret sharing* (SS) scheme by $\mathcal{P} = \{P_1, \dots, P_n\}$, and by the *dealer* \mathcal{D} . The dealer possesses a value $s \in \mathbb{F}$ as a secret input for the scheme, and then he shares it among the players. We will consider the general access structure case. The groups which are allowed to reconstruct the secret are called *qualified* (denoted by Γ) and the groups which should not be able to obtain any information about the secret are called *forbidden* (denoted Δ). We will consider the complete case where Δ is equal to Γ^c (the complement of Γ).

Consider an *adversary* \mathcal{A} who may corrupt some of the players. The adversary is characterized by a particular subset $\Delta_{\mathcal{A}}$ of Δ called *adversary structure* while the set Δ is called *privacy structure*. The set of players belonging to Δ are *curious* but execute the protocol correctly, while the set of players belonging to $\Delta_{\mathcal{A}}$ are *corrupt* and may not follow the protocol. In other words, one distinguishes between *passive* and *active* corruption. If $\Delta_{\mathcal{A}} \neq \{\emptyset\}$ the adversary is called *active*, and otherwise *passive*. The adversary can also be *static*, meaning that the set of corrupt players is chosen by the adversary once and for all before the protocol starts, or *adaptive* meaning that the adversary can at any time during the protocol choose to corrupt additional player. Further on we consider the so called *mixed adversary* model.

An adversary \mathcal{A} is called \mathcal{Q}^ℓ if no ℓ sets in $\Delta_{\mathcal{A}}$ cover the full set of players \mathcal{P} . An adversary is called *rushing* if in any round of communication, corrupt players receive messages before the honest players, and then, based on those messages the adversary can decide whom to corrupt next.

A secret sharing scheme based on an access structure Γ is a pair (*Share*, *Reconstruct*) of protocols (phases), namely, the *sharing phase*, where dealer \mathcal{D} shares among the players a secret s from a field \mathbb{F} , and the *reconstruction phase*, where the players try to reconstruct s , such that the following two properties hold:

Privacy - the players of any set $B \in \Delta$ learn nothing about the secret s as a result of the sharing phase;

Correctness - the secret s can be computed by any set of players $A \in \Gamma$.

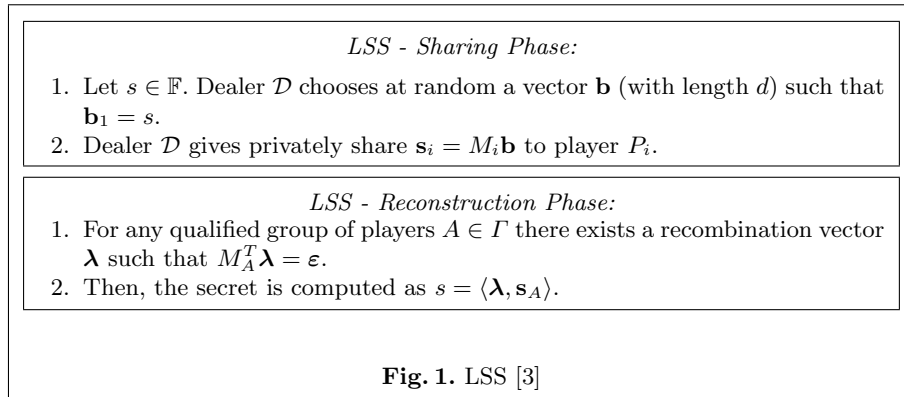
An SS scheme is *linear* (denoted by LSS) if the dealer uses only linear operations to share (to reconstruct) the secret among the participants.

Definition 1. A Monotone Span Program (MSP) \mathcal{M} is a quadruple $(\mathbb{F}, M, \varepsilon, \psi)$, where \mathbb{F} is a finite field, M is a matrix (with m rows and $d \leq m$ columns) over \mathbb{F} , $\psi : \{1, \dots, m\} \rightarrow \{1, \dots, n\}$ is a surjective (labeling) function and $\varepsilon = (1, 0, \dots, 0)^T \in \mathbb{F}^d$ is called target vector. The size of \mathcal{M} is the number m of rows.

As ψ labels each row with a number i from $\{1, \dots, m\}$, corresponding to player $P_{\psi(i)}$, one can imagine that each player owns one or more rows from the matrix. Also, consider a “function” φ from $\{P_1, \dots, P_n\}$ to the sets of subsets of P , $P(\{1, \dots, m\})$ which gives for every player P_i the set of rows he owns (denoted by $\varphi(P_i)$). In some sense φ is the “inverse” function of ψ . For any set of players

$B \subseteq \mathcal{P}$, consider the matrix consisting of the rows these players own in M ; thus, let M_B denote the restriction of M to these rows (i.e. for i with $i \in B$).

An MSP is said *to compute* an access structure Γ when: $\varepsilon \in \text{im}(M_A^T)$ if and only if A is a member of Γ . Hence, when a set A is accepted by \mathcal{M} there exists a so-called *recombination vector* (column) $\lambda \in \mathbb{F}^{|\varphi(A)|}$ such that $M_A^T \lambda = \varepsilon$ (or in other words there exists $\lambda \in \mathbb{F}^m$ such that $M^T \lambda = \varepsilon$ and $\text{supp}_{\mathcal{P}}(\lambda) \subseteq A$). Using the recombination vector λ it is easy to see that for the qualified group A the following relation holds: $\langle \lambda, M_A(s, \rho)^T \rangle = \langle M_A^T \lambda, (s, \rho)^T \rangle = \langle \varepsilon, (s, \rho)^T \rangle = s$ for any secret s and any random vector ρ . Notice that the vector $\varepsilon \notin \text{im}(M_B^T)$ (i.e. B is forbidden group) if and only if there exists a vector $\mathbf{k} \in \mathbb{F}^d$ such that $M_B \mathbf{k} = \mathbf{0}$ and $\mathbf{k}_1 = 1$. When both matrix rows and columns are labeled by a function ψ , we call such a matrix *doubly-labeled*. We present as an example the protocol for LSS in Fig. 1.



A verifiable secret sharing (VSS) scheme consists of two stages. In the first stage, the dealer commits to a unique value s' independent of the action of the corrupt players. Moreover, whenever the dealer is not corrupt s' is equal to s . In the second stage, the already committed value s' will be recovered by all honest players independent of the action of the corrupt players.

When the adversary has unlimited computing power one refers to this as the *information theoretic* model (abbreviated i.t.) and talks about *unconditional* security. In contrast to this, when the parties are limited to probabilistic polynomial time computations, one refers it as the *cryptographic* model and uses the term *computational* security. When in the information theoretic model additionally *zero error probability* is achieved, one talks about *perfect* security. We consider the secure-channels (i.t.) model with broadcast (which may be given as primitive or simulated) and focus on perfect secure protocols. Also we will consider an adversary, which can be *active*, *adaptive* and *rushing*. Now a formal definition of VSS follows.

Definition 2. A verifiable secret sharing scheme secure against a $\Delta_{\mathcal{A}}$ -adversary is a pair (Share, Reconstruct) of protocols (phases). At the beginning of the Share phase the dealer \mathcal{D} inputs to the protocol a secret $s \in \mathbb{F}$, at the end of a Share phase each player P_i is instructed to output either “accept” or “reject”. At the end of the Reconstruct phase each player P_i is instructed to output a value in \mathbb{F} . The protocol is unconditionally secure if the following properties hold:

- Acceptance (Termination): If one honest player “rejects” at the end of Share, then all honest players “reject”. No honest player “rejects” if \mathcal{D} is honest.
- Correctness (Verifiability): Share is either rejected, or \mathcal{D} is committed to a unique value s' . In Reconstruct all honest players will output the same value s' . Moreover if the dealer is not corrupt $s' = s$.
- Privacy (Unpredictability): If \mathcal{D} is honest and shares a secret s , the adversary learns nothing about s from his view of the Share protocol.

The main difference between SS and VSS is that in VSS the players and even the dealer can be corrupt. That is why the *Detection* sub-phase is included in both *Share* and *Reconstruct* phases in order to detect incorrect behavior of the players and/or the dealer. Another difference is that VSS provides *robustness*, i.e., the players can reconstruct the secret without the dealer’s help.

In a commitment scheme, a committer \mathcal{D} commits to a secret value s by publishing a commitment in such a way that the commitment reveals nothing about the secret s . This is the first important characteristic of the scheme, called *hiding*. The player can later open the commitment to reveal s in a verifiable way, in the sense that the player cannot open the commitment to any other value than s . This is the second important characteristic of the scheme, called *binding*.

A typical way to use commitments is to require the committer to prove certain relations among his committed values, without revealing these committed values in the process. A player committed to two secrets can be required to prove a commitment to their sum or product, provided that addition and multiplication of secrets is defined. A commitment scheme is called *homomorphic* when it is additive, which is the case with many known commitment schemes. Obviously homomorphic schemes handle trivially the additive relations, even non-interactively.

Generally speaking a commitment scheme consists of two phases. The *Commit* phase allows \mathcal{D} to commit to a value $s \in \mathbb{F}$, and the *Open* phase allows \mathcal{D} later to open the commitment. Both phases can end by honest players rejecting their outcome, in which case \mathcal{D} is declared corrupt. \mathcal{D} ’s *commitment* to a value s is denoted by $[s]_{\mathcal{D}}$, which means the total information distributed to the players during the protocol *Commit*.

There are known efficient *interactive* zero-knowledge protocols in the two-players setting and there are natural adaptations of these protocols to distributed setting with n players, where some of them are corrupt. A Distributed Commitment (DC) scheme is a pair of protocols, (*Commit*, *Open*), among a possibly corrupt dealer (committer) \mathcal{D} and a set of players P_1, \dots, P_n . Certain distributed commitment protocols do not involve any interaction between players P_1, \dots, P_n and hence, can be described as taking place between the committer and just a

single verifier. Most computationally secure distributed commitments are of this special form, where each of the players independently plays the role of the verifier. But in the information theoretic setting the interaction between the players is crucial.

Definition 3. A distributed commitment scheme secure against a $\Delta_{\mathcal{A}}$ -adversary is a pair (Commit, Open) of protocols (phases). At the beginning of the Commit phase the committer (dealer) \mathcal{D} inputs to the protocol a secret $s \in \mathbb{F}$; at the end of the Commit phase each player P_i is instructed to output either “reject” or “accept”. In the beginning of the Open phase the dealer opens his commitment. At the end of the Open phase each player P_i is instructed to output either “reject” or “accept”. The protocol is secure if the following properties hold:

- **Acceptance:** If one honest player “rejects”, then all honest players “reject”. No honest player “rejects” if \mathcal{D} is honest.
- **Binding:** Commit is either rejected, or \mathcal{D} is committed to a unique value s . Open is either rejected, or the committed value is correctly opened.
- **Hiding:** If \mathcal{D} is honest and commits to a value s , the adversary learns nothing about s from his view of the Commit protocol.

It is well-known that commitments in the two party setting can be either unconditionally hiding or binding, but not both. In contrast, the information theoretic distributed commitments can be both unconditionally hiding and binding.

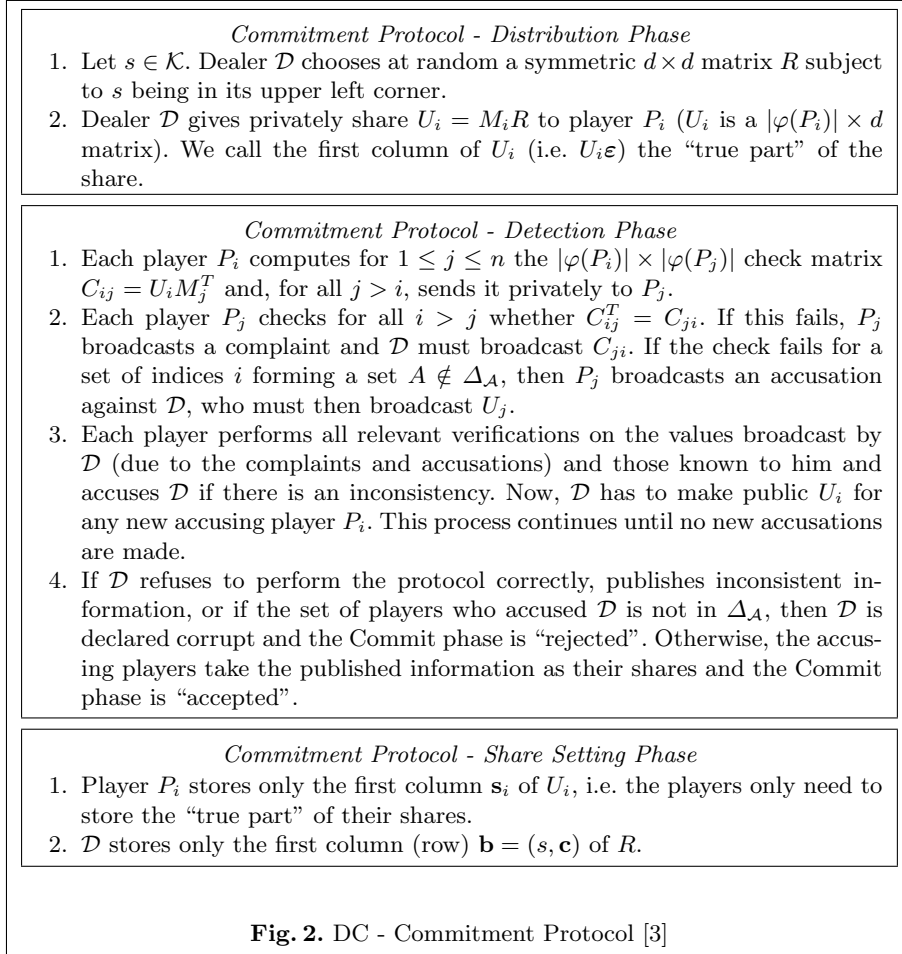
It is important to note that information held by the honest players in DC uniquely determines the committed value. If in a scheme the honest players can efficiently reconstruct the committed value, even in the presence of an adversary, such a scheme is identical to a VSS. Recall that if the scheme is not a VSS, then the honest players cannot efficiently determine the value without \mathcal{D} ’s help, but they should be able efficiently to verify and to accept (or to reject) the value in the Open phase. Thus, DC can be seen as a reduced (weaken) version of VSS.

3 Previous Work

In this section we will give a short description of the main known schemes. From now on let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP, φ be the “inverse” of ψ , and M be an $m \times d$ matrix. Let \mathcal{M} compute an access structure Γ and consider a $\Delta_{\mathcal{A}}$ -adversary which is \mathcal{Q}^3 and which can be *active*, *adaptive* and *rushing*. Recall that we consider the secure-channels (i.t.) model with broadcast (which may be given as primitive or simulated by the participants) and we focus on perfect (i.e. error-free unconditionally) secure protocols.

3.1 Distributed Commitments

We will describe a perfectly secure DC scheme, given by Cramer *et al.* [3]. The idea is to extend the LSS, induced by the MSP, to a commitment scheme by



ensuring that even when the dealer is corrupt, the correct players are guaranteed to receive consistent shares of the secret (see Fig. 2).

A way \mathcal{D} to open the commitment is to broadcast s and the full set of the “true parts” of the players’ shares i.e. $(\mathbf{s}'_1, \dots, \mathbf{s}'_n)$. In this case the “consistency” of the shares should be verified, e.g. by solving the system of linear equations $M\mathbf{b} = \mathbf{s}'$ regarding \mathbf{b} and taking into account that $\mathbf{b}_1 = s$. Since $\mathbf{b} \in \mathbb{F}^d$ the complexity of this approach is $O(d^3)$.

Another (more efficient) way to open the commitment is to broadcast \mathbf{b} , i.e. s and \mathbf{c} . In this case each player P_i accuses \mathcal{D} if his stored value \mathbf{s}_i does not match, i.e. if $M_i \mathbf{b} \neq \mathbf{s}_i$.

We stress here the following differences between i.t. DC and the two-party computationally secure commitments: The commitment of \mathcal{D} to s is $[s]_{\mathcal{D}} = (\mathbf{s}_1, \dots, \mathbf{s}_n)$. The private information of \mathcal{D} is (s, \mathbf{c}) (\mathbf{c} a random vector).

Note that i.t. DC schemes are *interactive*, i.e. in the commit phase the players should interact in order to accept or reject the commitment, while computationally secure commitment schemes could be *non-interactive*. The protocol described in the Detection phase is referred also as *pair-wise checking* [3, 4], but we will refer to it as *pair-wise checking with accusations*.

A sharing is called *one or two-dimensional* when the dealer shares the secret by means of a vector (see Fig. 1) respectively a matrix (see Fig. 2). Using a two-dimensional sharing one is able to perform pair-wise with accusations consistency checking. Notice that the commitments $[s_i]_{\mathcal{D}}$ form an $m \times m$ matrix MRM^T , which is doubly-labeled and the matrix R is symmetric.

3.2 Auxiliary Commitment Protocols

In [3] the authors describe the following auxiliary homomorphic commitment protocols: *Commitment Transfer Protocol* (CTP) and *Commitment Sharing Protocol* (CSP).

A Commitment Transfer Protocol

This is a protocol that allows a dealer \mathcal{D} committed to a value s by a commitment $[s]_{\mathcal{D}}$ to transfer his commitment to another player \mathcal{R} who therefore becomes committed to the same value by a commitment $[s]_{\mathcal{R}}$. The players in \mathcal{P} may or may not be involved in this protocol. Both \mathcal{D} and \mathcal{R} may also be players from \mathcal{P} . A CTP is secure against a $\Delta_{\mathcal{A}}$ -adversary [3] if the following principles hold:

- *Correctness*: After the protocol, either \mathcal{R} is committed to s or \mathcal{D} is identified by all honest players as being corrupt. An honest \mathcal{D} will never be accused of being corrupt.
- *Privacy*: If both \mathcal{D} and \mathcal{R} are honest, then the adversary obtains no information about s from observing the CTP.

If \mathcal{D} is honest, then a corrupt player \mathcal{R} cannot achieve anything by cheating during the CTP (see Fig. 3). We stress here that the complexity of the so described CTP is the same as of a DC.

Notice that CTP can also be used to prove that two values s and s' are equal provided that the corresponding commitments $[s]_{\mathcal{D}}$ and $[s']_{\mathcal{D}}$ are given.

A Commitment Sharing Protocol

The CSP allows a dealer \mathcal{D} already committed to a value s by $[s]_{\mathcal{D}}$ to share s , resulting in shares $(\mathbf{a}_1, \dots, \mathbf{a}_n)$, in such a way that every player is committed to his share \mathbf{a}_i of s by $[\mathbf{a}_i]_{P_i}$. A CSP is secure against $\Delta_{\mathcal{A}}$ -adversary [3] if the following principles hold:

- if \mathcal{D} is honest, then (a) every player is committed to his share of s , and (b) the adversary learns nothing about s .
- If \mathcal{D} is corrupt, then either condition (a) holds or \mathcal{D} is by all honest players detected as being corrupt.

Commitment Transfer Protocol:

The following protocol converts $[s]_{\mathcal{D}}$ into $[s]_{\mathcal{R}}$.

1. Let $[s]_{\mathcal{D}} = (\mathbf{s}_1, \dots, \mathbf{s}_n)$ and the private information of \mathcal{D} be (s, \mathbf{b}) , i.e. $M(s, \mathbf{b}) = (\mathbf{s}_1, \dots, \mathbf{s}_n)$.
2. Dealer \mathcal{D} sends privately to \mathcal{R} the shares determining s , i.e. $(\mathbf{s}_1, \dots, \mathbf{s}_n)$ and his private information (s, \mathbf{b}) . If this information is not consistent then \mathcal{R} broadcasts a complaint, and the protocol continues with the last step.
3. Player \mathcal{R} commits to s (independently), resulting in $[s]_{\mathcal{R}}$. Let $[s]_{\mathcal{R}} = (\tilde{\mathbf{s}}_1, \dots, \tilde{\mathbf{s}}_n)$ and let the corresponding private information be $(s, \tilde{\mathbf{b}})$.
4. Using the linearity of commitments, \mathcal{R} opens the difference $[s]_{\mathcal{D}} - [s]_{\mathcal{R}}$ to reveal 0, using the information from the first step as if he has created $[s]_{\mathcal{D}}$ himself. In other words \mathcal{R} publishes the vector $(0, \mathbf{b} - \tilde{\mathbf{b}})$.
5. Every player P_i is now able to verify that his share $\mathbf{s}_i - \tilde{\mathbf{s}}_i$ corresponds to the public vector $(0, \mathbf{b} - \tilde{\mathbf{b}})$, i.e. that $M_i(0, \mathbf{b} - \tilde{\mathbf{b}}) = \mathbf{s}_i - \tilde{\mathbf{s}}_i$. If this succeeds, the protocol ends. Otherwise go to the next step.
6. If we arrive at this step, it is clear that at least one of \mathcal{D} or \mathcal{R} is corrupt, so \mathcal{D} must then open $[s]_{\mathcal{D}}$ in public, and we either disqualify \mathcal{D} (if he fails) or continue with a default commitment to s assigned to \mathcal{R} .

Fig. 3. CTP [3]

As we will see in the next sections committing to a and then, performing CSP is equivalent to applying a VSS to a and then, proving that this is a commitment to the same secret. The purpose of this protocol is that the commitments to the shares will prevent the corrupt players from contributing false shares when the secret is reconstructed (see Fig. 4). We stress here that the complexity of the so described CSP is equal to the complexity of d DC plus m CTP and hence, in total the complexity of $\Omega(m)$ DC (recall that $d \leq m$).

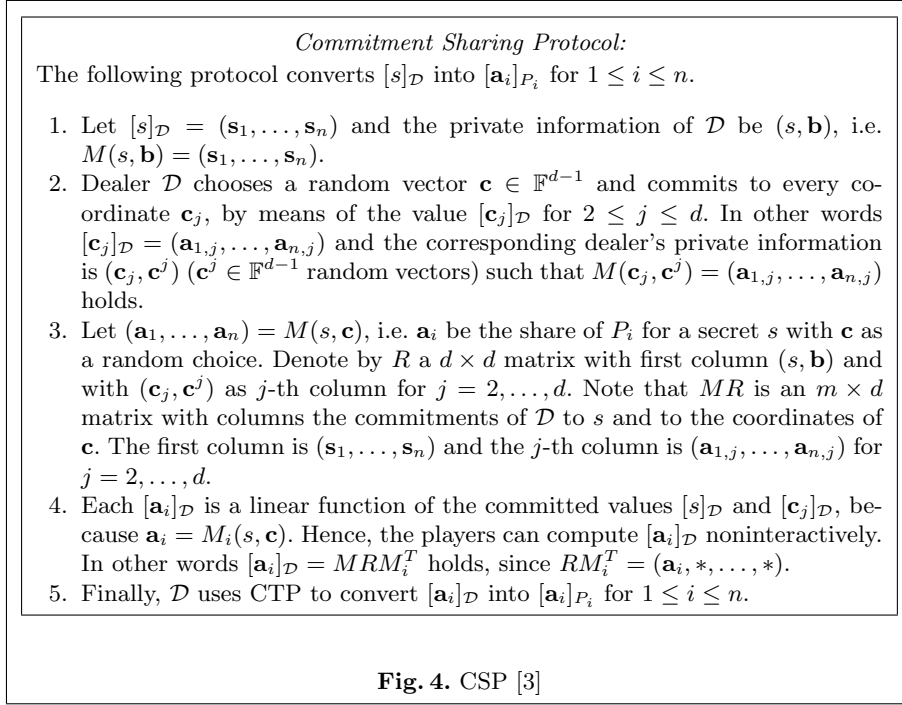
Notice that, as in Section 3.1, the commitments $[\mathbf{a}_i]_{\mathcal{D}}$ form an doubly-labeled $m \times m$ matrix MRM^T , but now the matrix R (used in Fig. 4) is not symmetric.

3.3 Verifiable Secret Sharing

As stated in [3, 1] a VSSs can be considered as DC schemes in which in addition every player is committed to his share.

Theorem 1. [3] *For any field \mathbb{F} and any LSS \mathcal{M} with \mathcal{Q}^3 adversary structure $\Delta_{\mathcal{A}}$, there exists an error-free VSS scheme in the i.t. model, secure against any active and adaptive adversary \mathcal{A} .*

The proof consists of two steps: first the DC scheme (see Fig. 2) is proven to be secure against such an adversary, second committing to s and then, performing CSP (which is secure against such an adversary) is proven to be equivalent to VSS.



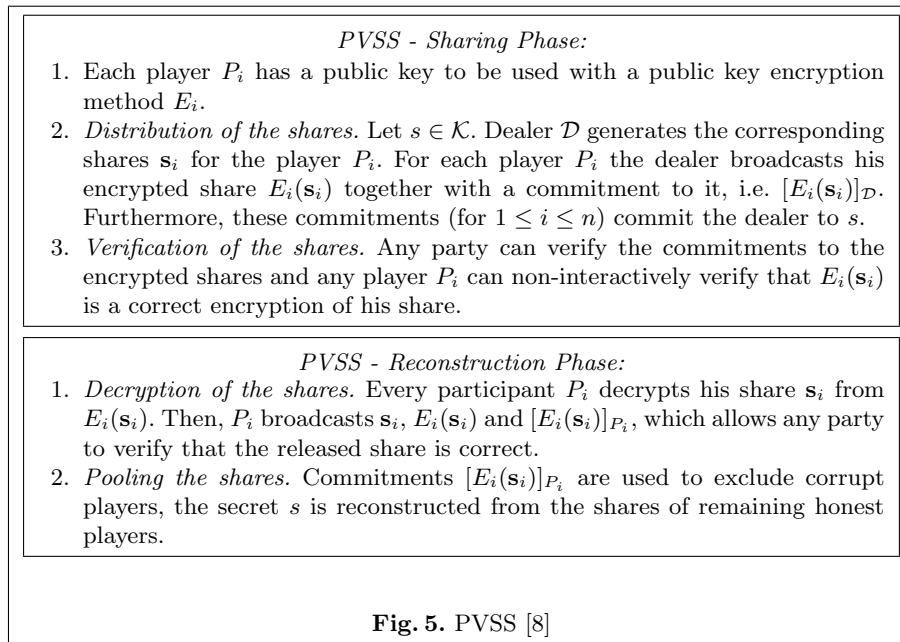
Recall that the complexity of a CSP (see Fig. 4) is $\Omega(m)$ times the complexity of a DC (see Fig. 2). Thus, the complexity of the VSS scheme, described in [3], is also $\Omega(m)$ the complexity of a DC scheme.

In the full version of [3] the authors point out that apart from the threshold case, their DC protocol (i.e. Fig. 2) is a VSS, (i.e. efficient reconstruction without the help of the dealer is possible) only in the case when for each set B whose complement is in $\Delta_{\mathcal{A}}$ the matrix M_B has full rank. In this case, players should store all information received from the dealer to reconstruct efficiently. However, the authors thought that in general, the scheme cannot guarantee efficient reconstruction, so it can only be used as a commitment scheme.

3.4 Publicly Verifiable Secret Sharing Scheme

A *publicly verifiable secret sharing* (PVSS) scheme (as introduced by Stadler in [7]) is a VSS scheme with the property that the validity of the shares distributed by the dealer can be verified by any party; not only by the players. Hence, the verification is not limited to the respective participants of the scheme. More efficient PVSS schemes have been proposed in [5, 8]. Note that all known PVSS schemes are computationally secure and non-interactive. In the PVSS model considered in [7, 8] it is assumed that the secret is computationally hidden, since in a secure channel model the communications between the participants is per-

formed over bilateral private channels and thus, it is not publicly verifiable. It is assumed that there is a broadcast channel and that this communication is publicly verifiable. In [8] the PVSS definition from [7] has been extended with the requirement that the reconstruction should also be publicly verifiable. Here we describe a model of non-interactive PVSS (see Fig. 5).

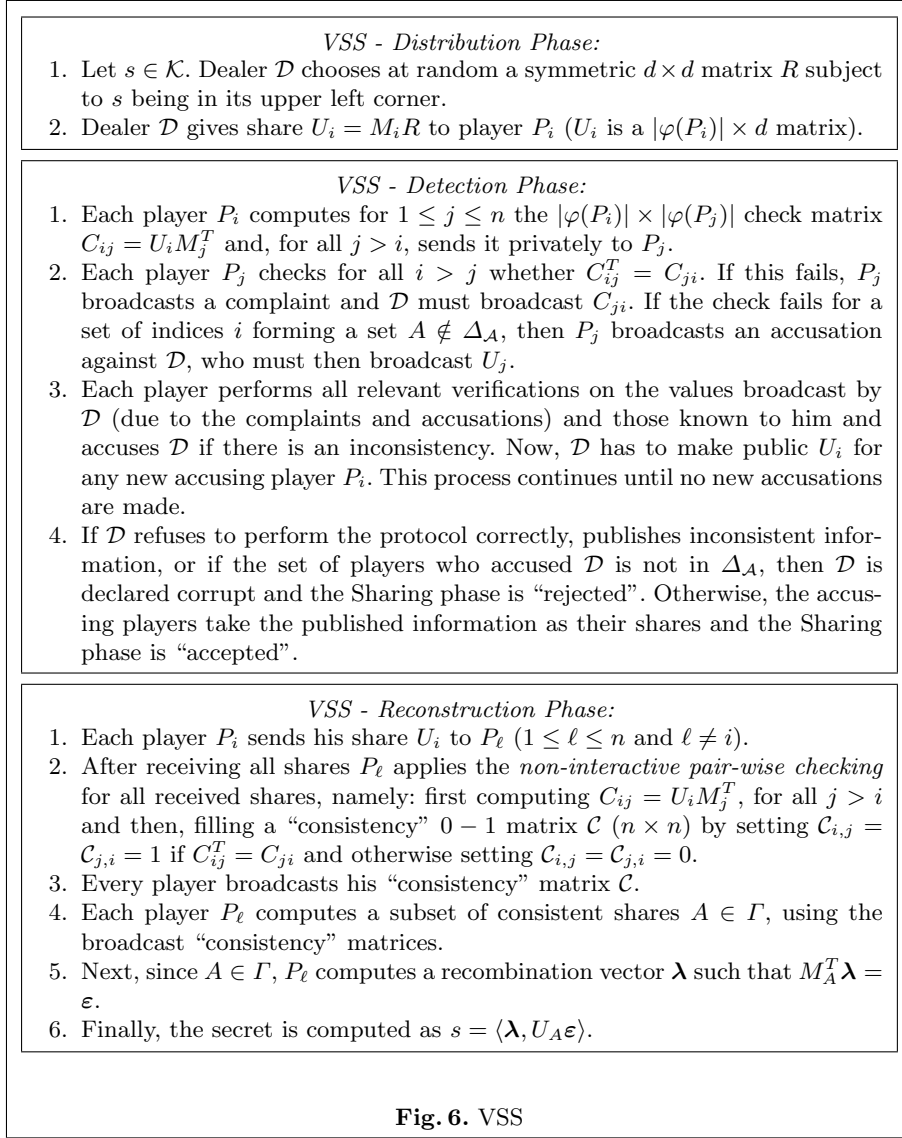


4 Our Results

4.1 An Efficient linear VSS Scheme

In this section we present a new construction, and we will prove that this scheme is more efficient perfectly secure linear VSS (see Fig. 6) comparing to [3]. We revisit the commitment protocol (see Fig. 2), namely the Distribution and the Detection Phases which together will form the Share Phase of the new VSS scheme.

Note that the “pair-wise” checking protocol ensures the consistency of the shares. We distinguish two flavors of this protocol: the first one - “with accusations”, used in the detection phase, is interactive and gives the dealer also the role of judge/verifier; the second one is “non-interactive”, it is used in the reconstruction phase (in VSS schemes) and does not involve the dealer but it is assumed that only certain shares are not consistent with each other in that case.



Therefore in the first variant at the end of the detection phase (in this case the dealer can be corrupt) the goal is to find a set of consistent shares belonging to a group of honest players or the dealer is declared to be corrupt if such a set does not exist.

To prove that the scheme is robust we first point out that all honest players have consistent shares at the end of the share phase (as it was the case for the DC schemes). Recall that we consider an adversary with \mathcal{Q}^3 adversary structure $\Delta_{\mathcal{A}}$.

Thus, having performed the pair-wise checking in the reconstruction phase one can always detect a group of qualified and honest players with consistent shares, which ensures the robustness. An efficient algorithm to find such set of consistent shares works as follows. First, all players which have inconsistent shares with any qualified group are excluded. Note that this could only be corrupt players and no honest player is excluded in this way. Denote the set of remaining players by A . Second, from the \mathcal{Q}^3 property it follows that in the set of remaining players a group of qualified and honest players exists. Denote such a group by G ($G \subseteq A$). From the consistency of the shares it follows that $M_j U_G^T = U_j M_G^T$ for every player $P_j \in A$, otherwise P_j does not belong to A . Hence, $M_j U_G^T \boldsymbol{\lambda} = U_j M_G^T \boldsymbol{\lambda} = U_j \boldsymbol{\varepsilon}$ holds, since $G \in \Gamma$ and $\boldsymbol{\lambda}$ is the recombination vector corresponding to G . Now notice that $U_j \boldsymbol{\varepsilon}$ is just the “true” part of U_j ’s share, and that $M_j U_G^T \boldsymbol{\lambda}$ is uniquely determined by the shares of G . Hence, the equality between them implies that the “true” part of U_j ’s share is uniquely determined by the shares of G . This observation ensures that all players in A have consistent “true” parts of their shares. Finally, one needs only the “true” parts of the shares in order to reconstruct the secret, this observation concludes the proof.

Comparing the VSS scheme described in Fig. 6 with the scheme from [3] (see Theorem 1) it is easy to see that our scheme has the same complexity as the DC scheme described in Fig. 2 and thus, it is $\Omega(m)$ times more efficient (see Section 3.3). Recall that m is in the worst case super-polynomial of n . Also as we have shown it is not necessary M_B to have a full rank in order this protocol to be VSS like it is the case in [3].

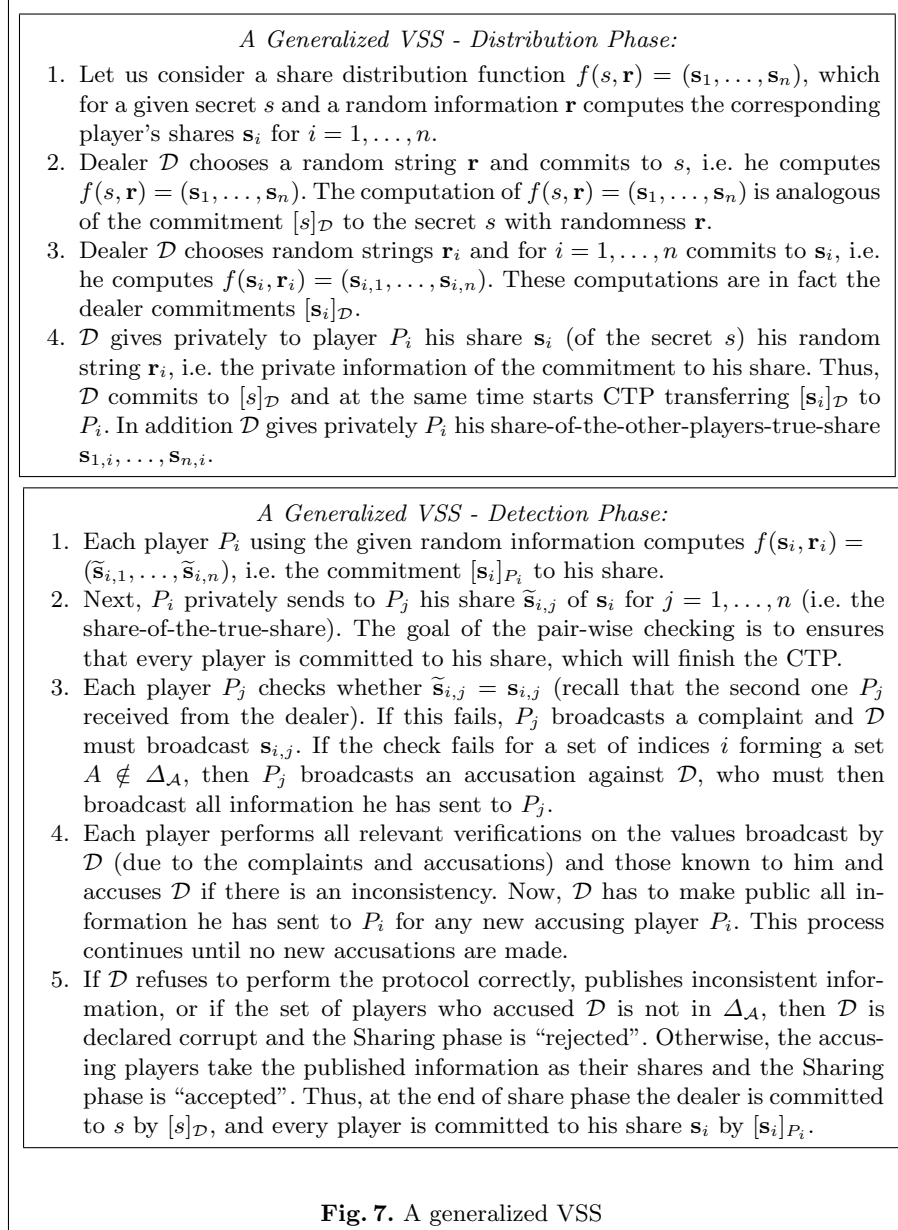
We would like to note that the proposed scheme cannot be directly compared with the scheme from [2] because of the following reasons: we consider perfectly secure, i.e. error free protocols, while the protocols of Cramer *et al.* are with negligible failure probability; we consider the general access structure case, while the focus of [2] is on the threshold case.

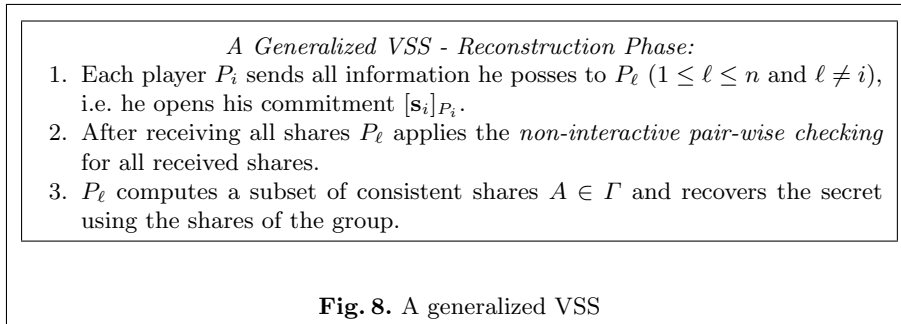
Security proofs of the schemes proposed in this section can be easily derived from the arguments given in [1, 3]. Due to the lack of space here we omit the proofs, but they will be provided in the full version of the paper.

4.2 A Generalized Treatment of VSS Schemes

The perfectly secure VSS schemes can be considered as schemes in which every player has the usual “one-dimensional share” plus shares of the “one-dimensional share” of the other players. We will show that these additional shares-of-the-true-share allow consistency checking. Thus, in this way the sharing phase achieves that the dealer is committed to the secret and that every player is committed to his share. This is achieved in the following way: first the dealer is committed to the secret and to each share, and then for every player the dealer’s commitment to his share is transferred to the player, in such a way that every player becomes committed to his share. Note that for a linear VSS scheme the “one-dimensional share” corresponds to the “true part” of the share (the first coordinate of the vector). Hence, such a protocol could be built on top of any SS, not necessarily

an LSS. As a generalization we propose perfectly secure VSS scheme, which is not necessarily linear, described in Fig. 7 and Fig. 8.





Note that in the linear VSS scheme described in Fig. 6 every player P_i possesses his share \mathbf{s}_i (of the secret s) and his random information (vector) \mathbf{r}_i . Using them he is able to compute his share-of-the-other-players-true-share $\mathbf{s}_{i,1}, \dots, \mathbf{s}_{i,n}$. Thus, comparing to the protocol described in Fig. 7 and Fig. 8 every player possesses n additional values, which makes the general scheme less efficient comparing to Fig. 6. In fact, for the VSS scheme described in Fig. 6 we have $U_i = (\mathbf{s}_i, \mathbf{r}_i)$ and the check value $C_{ij} = \mathbf{s}_{i,j}$.

Note that we performed CTP without requiring the scheme to be homomorphic (obviously the protocol in Fig. 3 is not working if the scheme is non-homomorphic). Namely the receiver reuses the same random string as the committer and the commitment of the receiver is accepted if a set of qualified players gets the same shares. This is, in fact, a generalization of the CTP protocol in the non-homomorphic case.

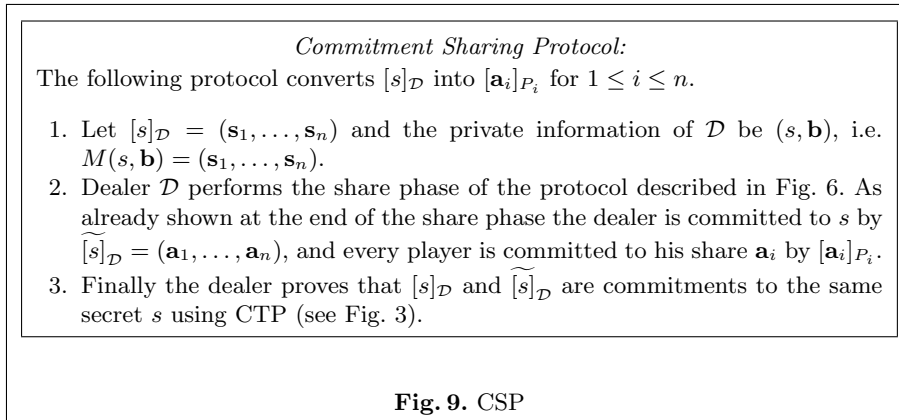
Remark 1. As it has been pointed out VSS schemes provide robustness. But, in fact, another important difference is that in VSS schemes every player stores his share (which is a commitment to its “true part”), while in distribute commitment schemes only the “true part” of the player’s share is kept, i.e. the player commitments are discarded.

Hence, in VSS schemes the dealer is committed to s by $[s]_{\mathcal{D}} = (\mathbf{s}_1, \dots, \mathbf{s}_n)$ and every player P_i is committed to his share \mathbf{s}_i by $[\mathbf{s}_i]_{P_i}$.

4.3 A More Efficient CSP

In this section we will show that a more efficient CSP exists. A refined analysis of the share phase in Fig. 6 shows that it is essentially a CSP. Thus, \mathcal{D} can either simply choose the first column of the matrix R used in the Commitment Protocol which corresponds to the commitment $[s]_{\mathcal{D}}$ and in this case the players just verify that they possess the same shares of s ; or R is chosen at random in the share phase but then, the dealer executes CTP proving that these are two commitments to the same secret.

In Fig. 9 we provide a description of the second case, the security of the CSP scheme follows directly from the security of the proposed VSS protocol. Note



that this approach uses one instead of d executions of DC (see Sect. 3.2) and one instead of m executions of CTP. Hence, the new CSP is $\Omega(m)$ more efficient, and it also has two additional advantages:

- The symmetry of the scheme is preserved since in this case R is symmetric (see Fig. 4), which is essential in case the protocol is a sub-protocol in more complex ones (e.g. MPC);
- The proposed scheme works (implementing the first case) even if the underlying SS is non-homomorphic.

4.4 VSS with Reduced Round Complexity as an example of PVSS

Since the interaction over a computer network is usually a time consuming operation, the *round* complexity of interactive protocols turns out to be an important complexity measure. The problem of finding the minimum number of rounds required to realize VSS has been studied by Gennaro *et al.* [6]. In order to reduce the number of rounds Gennaro *et al.* [6] have modified the expensive pair-wise checking protocol. The next protocol we provide here (we describe only the Detection phase) is the Gennaro *et al.* protocol with reduced round complexity (see Fig. 10). Note that the usage of pads leads to encryption of the share (in step 2.), which allows to broadcast the encrypted share. The same approach can be used to reduce the round complexity of the generalized VSS scheme (see Fig. 7).

After we have considered several perfectly secure VSS (DC) schemes, it is clear that for such schemes the interaction between players is crucial in order to verify a given dealer’s commitment. But for computationally secure VSS (DC) schemes, every player may without interaction with other players verify the dealer’s commitment. Thus, computationally secure schemes may be *non-interactive*, while the unconditionally secure schemes are *interactive*.

All known PVSS schemes are computationally secure. It is an interesting question what would be a PVSS scheme in the information theoretic setting?

PVSS - Detection Phase:

1. Player P_i generates and sends to every P_j a random $|\varphi(P_i)| \times |\varphi(P_j)|$ matrix (pad) V_{ij} through a private channel.
2. Each player P_i computes for $1 \leq j \leq n$ the $|\varphi(P_i)| \times |\varphi(P_j)|$ check matrix $C_{ij} = U_i M_j^T + V_{ij} + V_{ji}^T$ and broadcast them.
3. Each player P_j checks for all $i \neq j$ whether $C_{ij}^T = C_{ji}$. If the check fails for a set of indices i forming a set $A \notin \Delta_{\mathcal{A}}$, then P_j broadcasts an accusation against \mathcal{D} , who must then broadcast U_j . P_j and P_i ($i \neq j$) broadcast all values $V_{ij} + V_{ji}^T$.
4. Each player performs all relevant verifications on the values broadcast by \mathcal{D} and P_j (due to the complaints and accusations) and those known to him and accuses \mathcal{D} if there is an inconsistency. Now, this repeats for every new accusing player P_i . This process continues until no new accusations are made.
5. If \mathcal{D} refuses to perform the protocol correctly, publishes inconsistent information, or if the set of players who accused \mathcal{D} is not in $\Delta_{\mathcal{A}}$, then \mathcal{D} is declared corrupt and the Sharing phase is “rejected”. Otherwise, the accusing players take the published information as their shares and the Sharing phase is “accepted”.

Fig. 10. VSS – with reduced round complexity

Considering both models (information theoretic and cryptographic) we can define a PVSS as a VSS in which every party (not only the players) can verify the validity of the players’ commitments to their shares, and in this way to verify the dealer’s commitments in the distribution phase. Of course, the scheme should be secure in the considered model - information theoretic or computational.

We should point out that the reduced round complexity protocol in Fig. 10 is in fact perfectly secure PVSS scheme. This is true since any verifier (not just the participants), using only the broadcast values, can verify the commitment matrices and fill himself a consistency matrix (performing in fact non-interactive pair-wise checking). We stress here that analogous to VSS the perfectly secure PVSS are interactive, while computationally secure PVSS may be non-interactive. Thus, the participation of all players is crucial for accepting or rejecting the sharing phase in the considered model. We emphasize here that using the pads (steps 1. and 2.) the players encrypt their commitments, which allow to broadcast them. The latter permits everybody to perform pair-wise verification on the broadcast (encrypted) commitments. Hence, any verifier can compute the maximum subset $A \subseteq \mathcal{P}$, such that the players in A have consistent shares. If $A^c \in \Delta_{\mathcal{A}}$ and if the set of players who accused \mathcal{D} is in $\Delta_{\mathcal{A}}$ then, the verifier accepts the sharing phase or otherwise declares the dealer corrupt.

5 Conclusions

In this paper, we propose more efficient linear VSS and CSP schemes, where the improvement is with a factor of $\Omega(m)$. Also we show that more efficient DC, VSS, CTP and CSP schemes could be built even if the underlying SS is non-homomorphic. As a result the communication complexity of the known CMP and MPC schemes [3] can be reduced with a factor of $\Omega(m)$. Continuing the idea from [1] a general treatment of perfectly secure DC and VSS schemes is given.

We also define a perfectly secure PVSS extending the previous results in [7, 8] and finally we show that the VSS protocol with reduced round complexity [6] is in fact a PVSS.

References

1. R. Cramer, I. Damgard, S. Dziembowski. On the complexity of verifiable secret sharing and multi-party computation, *STOC'00*, ACM Press, 2000.
2. R. Cramer, I. Damgard, S. Fehr. On the cost of Reconstructing a Secret, or VSS with optimal Reconstruction Phase, *CRYPTO'01*, LNCS 2139, 2001, pp. 503–523.
3. R. Cramer, I. Damgard, U. Maurer. General Secure Multi-Party Computation from any Linear Secret Sharing Scheme, *EUROCRYPT'00*, LNCS 1807, 2000, pp. 316–334. full version – *Cryptology ePrint Archive: Report 2000/037*.
4. S. Fehr, U. Maurer. Linear VSS and Distributed Commitments Based on Secret Sharing and Pairwise Checks, *CRYPTO'02*, LNCS 2442, 2002, pp. 565–580.
5. E. Fujisaki, T. Okamoto. A practical and provably secure scheme for publicly verifiable secret sharing and its applications, *EUROCRYPT'98*, LNCS 1403, 1998, pp. 32–46.
6. R. Gennaro, Y. Ishai, E. Kushilevitz, T. Rabin. The round complexity of verifiable secret sharing and secure multicast, *STOC'01*, 2001.
7. M. Stadler. Publicly Verifiable Secret Sharing, *EUROCRYPT'96*, LNCS 1070, 1996, pp. 190–199.
8. B. Schoenmakers. A simple publicly verifiable secret sharing scheme and its applications to electronic voting, *CRYPTO'99*, LNCS 1666, 1999, pp. 148–164.